UNIVERSITÉ DE LORRAINE

École doctorale IAEM Lorraine

Sémantique lexicale, compositionnalité et coercions. Fondements théoriques des types sémantiques

THÈSE

présentée et soutenue publiquement le 22 novembre 2022

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

William Babonnaud

Composition du jury

Présidente :	Laurence Danlos	LLF, CNRS, Université Paris Cité, France
Rapporteurs :	Paul-André Melliès	IRIF, CNRS, Université Paris Cité, France
	Christian Retoré	LIRMM, CNRS, Université de Montpellier, France
Examinatrice :	Alda Mari	Institut Jean Nicod, CNRS, Paris, France
Examinateur :	Mathieu Constant	ATILF, CNRS, Université de Lorraine, France
Membre invitée :	Laura Kallmeyer	Heinrich-Heine-Universität, Düsseldorf, Allemagne
Encadrant :	Philippe de Groote	Loria, Inria, Nancy, France

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Sommaire

Remerciements **v**

Avant-propos **vii**

Notations **xi**

Listes de données **xiii**

État de l'art et problématique **1**

1 La question des types sémantiques **3**

1.1 La place des types dans les modèles syntaxico-sémantiques 3

1.2 L'intégration de la sémantique lexicale et ses implications 10

1.2.1 Des limites au système montagovien ? 10

1.2.2 Lexique, types, et la répartition de l'information sémantique 12

1.2.3 Vers une théorie des coercions 15

1.2.4 Des types encore trop abstraits 19

1.3 Des principes-clé pour un système plus efficace 24

1.4 Problématisation de la question des types 30

I Études linguistiques **37**

2 Contributions historiques à la notion de type sémantique **39**

2.1 La théorie des types simples 40

2.2 Les catégories sémantiques 43

2.3 La hiérarchie générativiste 46

2.4 La sémantique lexicale structuraliste et au-delà 50

2.5 L'erreur de catégorie 54

3 Vers une formalisation des principes linguistiques des types 61

3.1 Ce que les types sémantiques ne sont pas 62

3.1.1 Types et référence. 62

3.1.2 Types et prototypes. 65

3.1.3 Types, logique et lexique. 66

3.1.4 Type et sens. 69

3.2 Axiomatique générale des types sémantiques 71

3.2.1 Notes préliminaires sur l'objet d'étude 72

3.2.2 Axiomes généraux 73

3.2.3 Existence et caractéristiques primaires 76

3.3 Fondations structurelles et ontologiques des types 78

3.3.1 La composante structurelle 79

3.3.2 La composante ontologique 82

3.3.3 Extension aux phénomènes sémantiques complexes 86

II Études formelles 91

4 Éléments de théorie des catégories 93

4.1 Premières définitions 94

4.1.1 Catégories 94

4.1.2 Foncteurs 98

4.2 Propriétés universelles, limites et autres constructions 103

4.2.1 Limites et colimites 103

4.2.2 Autres constructions générales 110

4.3 Topos élémentaires 115

5 Modèles catégoriques des théories de types 123

5.1 Principes généraux 123

5.2 Modèles catégoriques pour la théorie des types simples 128

5.3 Vers des modèles adaptés à la sémantique linguistique 132

5.3.1 Logique catégorique du premier ordre 133

5.3.2 Un modèle pour le système montagovien 136

5.4 Inverser la dynamique : des modèles aux théories 139

5.4.1 Reformulations admissibles 139

5.4.2 La construction des ontologies 144

5.4.3 Ontologies sommersiennes et logique intuitionniste 149

6 Construction de théories de types sémantiques **157**
6.1 Le constructeur de types prédicats 158
6.2 Un système de sous-typage covariant 164
6.3 Le traitement des hétérotypes 179
6.4 Coercions libres et principes d'inférence 185
6.5 Transformations structurelles 196

III Études empiriques **203**

7 Détermination empirique de types ontologiques **205**
7.1 Linguistique formelle et données empiriques 206
7.2 De la théorie sommersienne à l'analyse de corpus 208
7.3 Algorithmes et éléments d'implémentation 213
7.3.1 Architecture globale 214
7.3.2 Choix d'implémentation 216
7.3.3 Difficultés techniques 219
7.4 Premiers résultats 221
7.4.1 Résultats des premiers tests 221
7.4.2 Leçons tirées de l'expérience 222

Conclusion **227**

Annexe **233**

A Récapitulatif des règles de CPTT **233**
A.1 Types 233
A.2 Coercions 233
A.3 Termes 234
A.4 Jugements de typage 235
A.5 Théorie équationelle 236

Bibliographie **237**

Remerciements

La présente thèse marque pour moi l'achèvement d'une longue période de découverte et de formation au cours de laquelle je me suis petit à petit avancé dans le monde de la recherche, et ceci n'aurait jamais été possible sans le soutien de Philippe de Groote, qui non seulement a accepté d'encadrer ma thèse, mais avait auparavant suivi mon parcours en amont depuis notre rencontre lors de mon tout premier stage de recherche il y a déjà sept ans ; je lui suis grandement reconnaissant de m'avoir fait confiance au tout long de mon projet d'étude et d'avoir été là dans toutes les démarches et les étapes de ce périple. Je souhaiterais remercier de même Laura Kallmeyer pour avoir cru en mon projet et en mes capacités avant et pendant cette thèse, et pour m'avoir offert l'opportunité de consolider mon travail au cours d'une quatrième année de thèse et plus encore, je suis honoré d'avoir reçu un tel soutien.

Les vicissitudes de la vie, dont tout particulièrement un certain épisode de pandémie et ses tentaculaires conséquences, ont fait que je n'ai pas pu échanger et surtout collaborer autant que je l'aurais souhaité avec mes collègues ; mais je retiens néanmoins de nombreuses discussions intéressantes avec de nombreuses personnes, et tout autant de bons moments qui ont contribué à faire de ma thèse une période agréable. Pour toutes ces raisons, je tiens à remercier ces personnes du monde académique que j'a eu la chance de croiser au cours de ces quatre dernières années. Au Loria, on y comptera Sylvain Pogodalla et Maxime Amblard pour des conversations scientifiques ou sur la vie en thèse qui m'ont été importantes, ainsi que mes collègues doctorantes et doctorants : Maria Boritchev, Pierre Ludmann, Clément Beysson, Laurinne Jeannot et Chuyuan Li (ainsi que leurs familles respectives, que je n'oublie pas : Titouan Carette, Juliette Wolff, Orianne Cabaret, et tous les noms que j'oublie), pour les discussions scientifiques ou non qui ont permis à mon esprit de vagabonder ailleurs que dans mon propre travail, jusqu'à pouvoir l'éclairer parfois d'une perspective nouvelle, et pour les moments de vie qui rendent mes quelques années à Nancy totalement inoubliables. Il en va de même à l'Université de Düsseldorf, où je tiens à remercier toutes les personnes qui m'ont accueilli et accompagné professionnellement et socialement lors de chacune de mes visites : Rainer Osswald, Simon Petitjean, Christian Wurm, Tatiana Bladier, Deniz Yavaş, Younes Samih et Long Chen. J'espère pouvoir collaborer plus étroitement avec toutes ces personnes si l'avenir m'en donne la possibilité.

Une telle suite de remerciements ne saurait être complète sans évoquer les proches qui rendent ma vie plus éclatante, et ne cessent par conséquent de me rendre la force nécessaire pour accomplir mes projets — dont la présente thèse est l'une des réalisations les plus monolithiques. Qu'il me soit donc permis de remercier tout d'abord ces compagnons cachanais, issus de la promotion la plus formidable que le département d'informatique de l'ancienne ENS Cachan ait jamais comptée, et que j'ai toujours un tel plaisir à retrouver malgré les grandes distances qui parfois nous séparent : Mathieu Huot, Emmanuel Arrighi, Guillaume Hocquet, Thomas Dupriez, Clément Beauseigneur et Antonin Penon, il y aurait tant à dire que je préfère ne rien ajouter ici ; vivement notre prochaine aventure ensemble ! Impossible également de ne pas mentionner Clément Ringeade, Jeanne Boîteux et Claire Kloppmann pour nos génialissimes repas théma-

tiques qui ajoutent à la vie les meilleures épices ; j'ai grand hâte de connaître notre prochaine destination. Je tiens par ailleurs à saluer mon vieil ami Ambroise de Izarra, nos retrouvailles ayant été un temps fort de ces dernières années ; que de chemin parcouru depuis le lycée ! Il reste bien des gens encore que j'aurais pu évoquer ici, et ces personnes sauront se reconnaître sans équivoque : ce n'est pas parce que leurs noms ne sont pas mentionnés explicitement dans ce paragraphe que je les oublie pour autant.

Et bien entendu, je n'en serai certainement pas là si je n'avais pas eu le soutien indéfectible de mes parents, qui ont su me porter assez haut pour que je devienne capable de produire des textes si pointus qu'il n'en comprendront pas toute la teneur ; je ne les remercierai jamais assez pour avoir su me transmettre cette *puissance* de vie qui m'anime aujourd'hui. De tendre pensées vont également à ma grand-mère, qui est toujours curieuse de tous les aspects de ma vie et à qui j'ai toujours grand plaisir à répondre. Je ne saurais pas non plus passer sous silence mes irremplaçables frères Damien et Baptiste, dont je sais qu'ils sont fiers de moi autant que je suis fier d'eux.

Les derniers mots de cette section ne peuvent aller qu'à une seule personne, ma rayonnante compagne qui éclaire ma vie plus que ne le ferait jamais aucun soleil, ma bien-aimée Célia Atzeni. Construire une vie de couple n'aura jamais été un tel défi qu'en réalisant tous deux nos thèses respectives, mais nous avons su malgré tous les tracas garder la tête froide et continuer à nous tirer respectivement vers le haut ; je n'aurais aucun mot assez juste pour exprimer le bonheur d'avoir une si magnifique personne à mes côtés. Je crois bien qu'un jour nous nous regarderons tous deux en tant que docteure et docteur, en nous disant : « Te souviens-tu de l'époque où nous faisions nos thèses ? — Oh que oui, ce n'était pas une mince affaire ! », et nous en rirons de bon cœur avec la tendresse que l'on a pour les souvenirs qui se bonifient avec le temps.

Avant-propos

Les thématiques abordées dans la présente thèse m'ont été introduites pour la première fois au cours des deux stages de recherches qui ont respectivement conclu ma dernière année de licence et ma première année de master. En 2015, j'ai été initié à la linguistique informatique par l'étude de la théorie de Richard Montague via son célèbre article « *The Proper Treatment of Quantification...* » [120], me familiarisant peu à peu avec les concepts majeurs de la sémantique formelle, dont le principe de compositionnalité, sous la direction de Philippe de Groote au Loria. L'année suivante, je me suis retrouvé à l'Université de Düsseldorf sous la supervision de Laura Kallmeyer, avec pour objectif de contribuer aux développements d'une interface syntaxe-sémantique à l'aide de la sémantique des cadres et de la logique hybride. Il m'avait alors été proposé deux directions possibles à explorer : la modélisation de la pluralité, ou bien celle de la polysémie. Je ne saurais pas expliquer quelles motivations ont guidé mon choix, si ce n'est peut-être une certaine *intuition* sur ce que je serais capable d'accomplir dans l'une ou l'autre direction, possiblement liée à ce que la lecture de quelques articles m'aura inspiré au sujet de chacun ; toujours est-il que c'est sur la seconde option que mon intérêt s'est porté.

Les types sémantiques n'étaient alors qu'un outil formel globalement indéfini, faisant partie intégrante des formalismes de la sémantique des cadres que j'avais à manipuler. J'ai donc abordé le problème de la polysémie et des coercions sous un angle *technique*, celui de déterminer comment les mécanismes du lexique génératif de Pustejovsky pouvaient être reproduits et intégrés à une interface syntaxe-sémantique formée de formalismes différents ; les types sémantiques considérés étaient alors exactement les mêmes que ceux proposés par Pustejovsky, ainsi d'ailleurs que les exemples reproduits. Le cœur de mon travail au cours de ce stage s'était donc focalisé sur la transposition formelle d'un mécanisme dans un système différent, dans le but de dégager la manière dont ce dernier pouvait être utilisé pour résoudre les problèmes de polysémie ; la question des types était alors globalement secondaire, et l'article rassemblant les résultats de ce stage [11] n'aborde pas véritablement cette question.

Tout ceci aurait pu s'arrêter là, puisque les années suivantes de mes études m'ont vu me promener sur d'autres chemins, que ce soit en explorant des problèmes en logique pure ou en tentant de proposer des modélisations d'autres phénomènes linguistiques à l'aide des cadres sémantiques, dont la manipulation au cours de mon stage à Düsseldorf m'avait laissé suffisamment enthousiaste pour que je m'essaie à les réutiliser ailleurs. Mais d'une manière ou d'une autre, il a bien fallu qu'une partie de ma conscience ait conservé le souvenir de ce stage et se rende compte que la construction que j'avais proposée n'était certainement pas le fin mot de l'histoire, car lorsqu'il m'a fallu construire une proposition de sujet pour ma thèse, je suis naturellement revenu sur la sémantique lexicale et tout particulièrement sur les applications des coercions, dont je préssentais alors un potentiel qui dépassait le simple cadre de la polysémie. Comme je devais revenir sous la direction de Philippe de Groote, le projet s'est adapté aux thématiques de l'équipe qu'il dirigeait, et cette question des coercions s'est donc redirigée vers le lambda-calcul plutôt que vers la sémantique des cadres. Mon nouvel objectif était donc a priori d'intégrer

le mécanisme des coercions, et donc par extention des modèles de la polysémie, à un nouveau formalisme d'orientation beaucoup plus montagovienne.

Ce n'est qu'une fois cette thèse véritablement commencée que le problème majeur a émergé devant mes yeux. En sémantique des cadres, la manipulation des types est très simple dans la mesure où ils ne sont que des propriétés de nœuds dans un graphe : ils sont tous au même niveau, et il n'y a pas de distinction proprement formelle entre les prédicats et les arguments. Mais la situation est tout autre en lambda-calcul simplement typé, où le constructeur de type flèche associé aux lambda-abstractions induit une hiérarchisation des types selon leur ordre. Or, une coercion est fondamentalement une transformation de type, et est donc influencée par la nature et les propriétés de ces types : lorsqu'ils sont tous des éléments atomiques mis sur un même niveau, il n'y a aucune question supplémentaire à poser et la mise en œuvre est immédiate, mais lorsqu'ils arborent une structure plus complexe, certaines décisions doivent être prises pour les paramétrer correctement. C'est sur la base de cette remarque que l'objectif de ma thèse s'est déplacé : avant de pouvoir prétendre raconter quoi que ce soit de pertinent sur les coercions, il m'était nécessaire d'explorer les types sémantiques eux-mêmes, et d'en poser des contours nets qui seraient autant de fondations solides pour la construction des coercions.

La question la plus difficile m'est alors apparue : *qu'est-ce qu'un type sémantique ?* À ma grande surprise, la réponse n'allait pas de soi, ce qui m'a amené à penser qu'il s'agissait d'un concept qu'on utilise beaucoup sans vraiment comprendre ce qu'il représente — ou du moins, que *j*'utilisais sans le comprendre. J'aurais pu m'accrocher à des propositions de réponses formulées par d'autres, mais aucune n'avait su me convaincre entièrement ; là où certaines me paraissaient seulement hasardeuses, d'autres imposaient un certain nombre de contraintes que j'ai jugées déraisonnables — on retrouvera dans la suite diverses précisions sur ces idées et leurs défauts. Mais surtout, j'ai été intringué par l'*arbitraire* dont se paraient ces propositions : les types sémantiques utilisés en pratique, que ce soit dans des exemples de formalismes ou dans des applications plus concrètes, portent toujours sur elles une composante intuitive, mais sans fondements, ce qui ouvre une autre question : comment vérifier qu'un type sémantique est pertinent, qu'il modélise le plus fidèlement possible cette partie du comportement de la langue ? Là encore, impossible de le déterminer sans évaluer d'abord la nature même des types. J'ai alors entrepris un travail de fond : sonder l'histoire de la sémantique formelle et de la sémantique lexicale, et assembler pièce par pièce les idées que le temps et l'usage ont accollées à la notion de type sémantique, dans l'espoir d'en dégager une théorie générale qui, sans forcément apporter une meilleure réponse que celles déjà avancées, pourrait au moins contribuer à clarifier la situation et les problèmes qui se posent.

Néanmoins, une telle étude n'aurait pu être complète sans une composante formelle. Je me suis donc donné pour objectif d'essayer de connecter les impératifs linguistiques et philosophiques propres à la notion de type sémantique avec son utilisation formelle, notamment en lambda-calcul puisqu'il s'agissait bien du cadre initial de mon sujet. Mon but ici n'était alors pas vraiment de proposer un *nouveau* formalisme, mais d'essayer de combler un vide apparent : celui de la théorie de type minimale, celle qui serait capable de proposer un cadre général qui unifierait tous les autres formalismes utilisant les types sémantiques, malgré leurs divergences. Le présent texte rassemble donc l'essentiel de mes trouvailles, de mes interrogations, et de mes constructions autour de ces objectifs. Bien que l'effervescence de mes idées au cours de son élaboration a été moins linéaire que sa structure et que le présent avant-propos ne le laissent à penser, il reproduit l'organisation logique de ces idées, en commençant par la remise en contexte des phénomènes qui ont mené au questionnement initial, puis en examinant les fondements linguistiques des types sémantiques avant d'en proposer un cadre mathématique plus formalisé ; par ailleurs, il inclut des réflexions autour de la définition concrète et pratique d'un système de types sémantiques.

Naturellement, toutes les réponses ne sont pas contenues dans le texte qui suit, mais j'espère y avoir mis suffisamment d'éléments pour convaincre que les types sémantiques posent un problème important pour la sémantique formelle, et pour poser des fondements solides sur lesquels des solutions pourront être trouvées.

Remerciements institutionnels. Les trois premières années de réalisation de la présente thèse ont été financées par le Ministère de l'Enseignement Supérieur et de la Recherche dans le cadre d'un contrat doctoral spécifique normalien (CDSN). La quatrième année a été financée par la Fondation allemande pour la recherche (*Deutsche Forschungsgemeinschaft*) dans le cadre du projet n° 440934416 « *Coercion and Copredication as Flexible Frame Composition* ».

Note relative aux publications. Au cours des quatre années de déroulement de ma thèse, il m'a été donné mainte fois l'opportunité de publier des articles diffusant des idées et des développements relatifs à mon sujet de travail. Il convient cependant de souligner que le contenu de la présente thèse étend très largement le contenu de ces articles, sans toutefois le recouvrir entièrement. Voici cependant les liens à retracer entre mes publications de cette époque et le présent texte : les idées générales exposées dans [7] sont diluées selon leur portée en sections 3.3, 5.3 et 5.4 ; le système introduit dans [10] a inspiré celui présenté en section 6.4, qui l'étend et l'améliore ; le lien entre prédicats et types discuté dans [9] est évoqué plusieurs fois au cours du texte sans être précisément réintroduit quelque part ; et enfin la théorie de types exposée dans [8] a posé des bases qui sont reprises, développées et largement améliorées en sections 6.1 et 6.2. Le reste des idées exposées dans le présent texte, à l'exception de petites mentions çà et là, n'a pas fait l'objet d'une autre publication de ma part.

Prérequis. La thèse défendue dans ces pages relève du domaine de la linguistique informatique et, à ce titre, fait appel à de nombreuses notions issues de chacune des deux disciplines parentes que sont la linguistique d'une part, et l'informatique d'autre part. Ces deux disciplines étant elles-mêmes vastes, il ne sera aucunement surprenant que le présent texte s'appuie sur une base de connaissances qu'il estime suffisamment répandues — du moins à ses possibles lectrices et lecteurs — pour se passer d'en réexpliquer les détails et aller directement au cœur du propos. Néanmoins, afin de fournir un maximum de repères à toutes les personnes pouvant être amenées à le consulter, nous nous proposons en préambule de toute discussion de situer plus précisément ce travail au sein du paysage scientifique en exposant les principaux prérequis nécessaires à sa bonne compréhension. Seules les théories que nous jugeons les plus complexes et les moins connues, comme la théorie des catégories, seront formellement rappelées dans le courant du présent document.

Le travail présenté ici s'intéresse à la modélisation par systèmes symboliques des interactions entre la *syntaxe*, c.-à-d. la manière dont les mots se combinent pour former des phrases, et la *sémantique*, c.-à-d. la signification des phrases ainsi formées. Nous supposerons acquises la distinction des différentes parties du discours existant dans la plupart des langues ainsi que la notion de *syntagme* et ses composantes. Une certaine connaissance du domaine de la grammaire générative est également attendue, notamment dans la distinction entre les structures *surfacique* et *profonde* de la syntaxe ainsi que dans la compréhension des arbres syntaxiques et des ambiguïtés qu'ils peuvent mettre en évidence. Une bonne compréhension des notions chomskyennes de *compétence* et *performance* [41], bien que peu mobilisées dans notre discussion, pourra aider à mieux cerner la problématique de la thèse ; par ailleurs, nous supposerons comprises les notions de *sémantique lexicale* [71] et de *pragmatique*, qui pèsent de manière importante sur ladite problématique.

Du côté mathématique, outre les bases traditionnelles incluant les classes, les ensembles, les fonctions et toutes les propriétés associées, nous utiliserons abondamment des éléments d'algèbre universelle : les notions de *signature*, de *structure* et de *terme*, avec naturellement la définition associée de *morphisme*, forment en effet le noyau dur des modèles que nous explorerons ou définirons dans ces pages. Ces modèles ne se limitent cependant pas à de simples structures : des systèmes formels d'ordre supérieur seront également abordés, avec principalement les *lambda-calculs* [14] qui occuperont une place importante dans nos discussions ; aussi, une connaissance assez large de leurs propriétés générales est nécessaire. On mettra notamment l'accent sur le lambda-calcul simplement typé — et sur les lambda-calculs typés de façon plus large —, et insistera sur les propriétés d'*équivalence* et de *réduction* sur les lambda-termes. De façon encore plus large, des connaissances dans le domaine de la *réécriture* sont conseillées. Par ailleurs, nous aurons recours à plusieurs reprises à des éléments de la théorie des ordres [47].

Représentation sémantique oblige, nous ferons aussi appel à de nombreux éléments de logique dans notre discussion ; c'est pourquoi une maîtrise des notions élémentaires associées au domaine du *calcul des prédicats* [80] se révèlera indispensable. Il conviendra également d'avoir en tête les différences inhérentes aux systèmes logiques *classique* et *intuitionniste* et leurs implications, ainsi que les applications de la correspondance de Curry-Howard. Il sera aussi essentiel de pouvoir lire et comprendre des *règles d'inférence*, dont l'usage est transposé de la logique à la théorie des types comme traditionnellement. Nous ferons de plus appel à de nombreux éléments des logiques *d'ordre supérieur*. Enfin, bien qu'elle ne sera pas intensivement utilisée dans ces pages, une connaissance basique des propriétés de la *logique linéaire* est recommandée.

Terminons cette première mise au point par l'introduction de quelques notations mathématiques et autres conventions typographiques. Les références aux ensembles classiques de nombres utiliseront la notation traditionnelle en gras de tableau noir : $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ et $\mathbb{R}$ désigneront donc respectivement l'ensemble des entiers naturels, des entiers relatifs, des nombres rationnels, et des nombres réels. Par ailleurs, $\mathbb{N}^*$ est l'ensemble des entiers strictement positifs. Nous utiliserons d'autres lettres de cette fonte pour désigner des ensembles fondamentaux dans les théories qui nous intéressent. Pour tout entier naturel n, la notation $[\![1; n]\!]$ désignera l'ensemble des entiers compris entre 1 et n inclus ; un tel ensemble est vide lorsque $n \leq 0$. Distinction sera faite entre la flèche $\to$ pour les types de fonctions et la flèche $\mapsto$ pour représenter des fonctions anonymes : par exemple, $n \mapsto n + 1$ désigne une fonction $\mathbb{N} \to \mathbb{N}$. Pour ce qui est de la typographie, nous mettrons en *italique* les mots qui dénotent des lemmes de la langue naturelle ; de plus, suivant les circonstances, les types dans nos théories pourront être représentés en *italique* ou en `machine_à_écrire`.

Notations

Théorie générale des ensembles

$\varnothing$	Ensemble vide
$\mathbb{N}$	Ensemble des entiers naturels
$\mathbb{N}^*$	Ensemble des entiers strictement positifs
$\mathbb{R}$	Ensemble des réels
$\mathcal{P}(A)$	Ensemble des sous-ensembles de A
$[\![m; n]\!]$	Ensemble des entiers compris entre m et n inclus
$x \mapsto f(x)$	Fonction anonyme qui à x associe $f(x)$

Ontologies et hétérotypes

$\mathbb{B}$	Ensemble des types ontologiques
$\leq$	Ordre sur les types ontologiques
e	Plus grand élément de $\mathbb{B}$
$\mathcal{P}(\mathbb{B})^\square$	Ensemble des sous-ensembles de $\mathbb{B}$ aux éléments incomparables pour $\leq$
$\mathbb{H}$	Ensemble des hétérotypes
$\mathfrak{a}$	Fonction des aspects $\mathbb{H} \to \mathcal{P}(\mathbb{B})^\square$
$\Rrightarrow$	Ordre des injections sur $\mathcal{P}(\mathbb{B})^\square$
$[\![\sigma]\!]$	Interprétation de σ dans le modèle catégorique

Jugements

$\Gamma \vdash u : \sigma$	Jugement de typage
$\Gamma \vdash u = v : \sigma$	Jugement équationnel
$\Gamma \mid \Theta \vdash \phi$	Jugement propositionnel
$\Gamma \vdash c : \sigma \leq \tau$	Jugement de sous-typage
$\Gamma; \Delta \vdash u : (\sigma, C)$	Jugement contraint

Catégories

$\mathsf{obj}(\mathcal{C})$	Classe des objets de $\mathcal{C}$
$\mathcal{C}(A, B)$	Classe des morphismes de $\mathcal{C}$ de A vers B
$g \circ f$	Composée de f suivie de g
id_A	Morphisme identité de A
$f : A \to B$	Abréviation pour $f \in \mathcal{C}(A, B)$
$A \cong B$	A et B isomorphes
$A \rightarrowtail B$	Monomorphisme de A dans B
$A \twoheadrightarrow B$	Épimorphisme de A dans B
$A \rightharpoonup B$	Morphisme de partiel de A dans B
Set	Catégorie des ensembles
$\mathsf{1}$	Catégorie à un objet
$\mathsf{2}$	Catégorie à deux objets et un morphisme

$\mathcal{C}^{\mathrm{op}}$	Catégorie opposée à $\mathcal{C}$
$S \downarrow T$	Catégorie virgule des foncteurs S et T
$\dot{X}$	Foncteur constant vers l'objet X
$\lim D$	Limite du diagramme D
$\operatorname{colim} D$	Colimite du diagramme D
0	Objet initial
0_A	Unique morphisme $0 \to A$
1	Objet terminal
$!_A$	Unique morphisme $A \to 1$
$A \times B$	Produit des objets A et B
$\prod_{i=1}^{n} D_i$	Produit des D_i, $i \in [\![1; n]\!]$
$A + B$	Coproduit des objets A et B
$\coprod_{i=1}^{n} D_i$	Coproduit des D_i, $i \in [\![1; n]\!]$
B^A	Exponentielle de A et B
$\mathrm{ev}_{A,B}$	Morphisme d'évaluation $B^A \times A \to B$
$\mathcal{P}A$	Objet puissante de A
$\mathrm{name}(f)$	Nom du morphisme f
Ω	Classificateur de sous-objet
$\mathrm{Sub}(A)$	Ensemble partiellement ordonné des sous-objets de A
$\leq$	Ordre sur les sous-objets de $\mathrm{Sub}(A)$
χ_m	Caractéristique du monomorphisme m
δ_A	Morphisme diagonal $A \to A \times A$

Listes de données

Table des figures

1.1 Fragment de hiérarchie de types issue de [183] 21
1.2 Fragment de hiérarchie de types issue de [135, §6.2] 22
1.3 Schéma du système d'interface syntaxe-sémantique hypothétique 28

2.1 Exemple des premiers degrés de la hiérarchie chomskyenne 49
2.2 Classification des noms d'après [3] 51

5.1 Règles de déduction naturelle intuitionniste 134

6.1 Définition de up_unify 188
6.2 Définition de down_unify 188
6.3 Règles de la théorie des types avec contraintes 191

7.1 Schéma général de l'algorithme ATHICE 216

Liste des tableaux

1.1 Catégories et types traités ou sous-entendus par Montague 5

2.1 Descriptions des niveaux d'analyse proposés par Sommers 56

Partie préliminaire

État de l'art et problématique

Chapitre 1

La question des types sémantiques

For us, types are not imposed afterwards to constrain the size of the world, but are a precondition of meaning.

Paul Taylor [177]

Au moment de l'écriture des présentes lignes, près de cinquante ans se sont écoulés depuis les balbutiements des modèles formels pour la sémantique des langues naturelles. Les prémisses de ces modèles sont bien entendus plus anciens, reposant d'une part sur les constructions formelles de la syntaxe héritées de l'école polonaise ainsi que des travaux de Chomsky, et d'autre part sur des considérations issues de la philosophie du langage, auxquelles nombre de philosophes ont contribué depuis la fin du XIX^{e} siècle. S'il conviendra de ne pas négliger ces prémisses (cf. chapitre 2), nous nous autorisons à les laisser de côté pour l'instant afin de mieux nous concentrer sur les questions soulevées par les traitements modernes de la sémantique formelle. En effet, malgré ces cinquante ans d'avancées et de modèles toujours plus performants, il subsiste encore des zones d'ombres et des imprécisions dans les multiples rouages qui fondent cette approche de la sémantique, et ce chapitre se donne pour objectif, à travers une revue des progrès de la sémantique formelle sur cette période, de mettre en lumière la petite partie de ces imprécisions sur laquelle la présente thèse s'attardera — et pourquoi elle s'y attardera.

Il s'agit donc ici de délimiter les contours du sujet central de la thèse défendue dans ces pages, tout en dressant un état des lieux de la recherche sur cette thématique au moment de leur rédaction. Nous commencerons par examiner en section 1.1 les fondements de la sémantique formelle moderne afin d'en dégager les principes généraux qui en ont émergé ; puis, nous aborderons les apports de la sémantique lexicale dans ce traitement formel et quels problèmes en découlent en section 1.2. Une observation préliminaire des problèmes en question nous conduira en section 1.3 à l'énoncé de principes destinés à guider la recherche de solutions, et la section 1.4 récapitulera les leçons à tirer de cette première analyse et formulera la problématique générale de la thèse developpée dans la suite de ce document.

1.1 La place des types dans les modèles syntaxico-sémantiques

Les approches modernes de la sémantique formelle, et plus particulièrement de l'interface syntaxe-sémantique, commencent au début des années 70 avec notamment les travaux extrêmement influents de Richard Montague [119, 120]. Dans le souci d'offrir à la sémantique une structure formelle aussi aboutie que ne l'était celle de la syntaxe à l'époque, Montague a pro-

posé un modèle où les expressions syntaxiques, telles que dérivées par une certaine grammaire, peuvent être traduites en expressions sémantiques par l'application d'un homomorphisme. Cette simple idée permet de concrétiser la notion de *compositionnalité* au niveau sémantique : chaque mot d'une expression linguistique apporte son propre fragment de signification, et la signification de l'expression entière est alors dérivée des fragments qui la composent en utilisant notamment l'information de la manière (syntaxique) dont les mots sont combinés entre eux. Dans le modèle de Montague, l'information syntaxique est préservée et utilisée par la définition même d'homomorphisme pour fournir un ordre de combinaison aux fragments de significations associés aux mots, et l'application de ces combinaisons permet alors d'en déduire la signification finale.

Plus précisément, la partie syntaxique de ce modèle est régie par un ensemble de *catégories syntaxiques*, qui regroupent les mots en fonction de leur statut grammatical et de leur comportement dans la phrase : il y a ainsi des catégories pour les noms communs, les verbes intransitifs, les adverbes, et ainsi de suite. La construction de ces catégories obéit à la logique de complétion syntaxique : un verbe transitif, par exemple, a notamment besoin d'un syntagme nominal en guise d'argument objet, et dès lors qu'on lui en fournit un, l'expression résultante a le même comportement qu'un verbe intransitif [1] ; en conséquence, la catégorie des verbes transitifs se construit à partir de celle des syntagmes nominaux et de celle des verbes intransitifs, exprimant que fournir un argument de cette première catégorie résulte en une expression de la seconde. Cela permet alors d'établir une construction récursive des catégories syntaxiques sur la base d'un petit nombre de catégories de bases. En usant de notation plus modernes, la proposition originale de Montague [120] repose ainsi sur trois catégories de base : les noms communs N, les sytagmes verbaux VP, et les phrases complètes S.

La syntaxe concrète s'élabore sur la base de mots regroupés en ensembles indexés par les catégories syntaxiques précedemment construites, à l'aide d'opérateurs abstraits permettant de former des expressions de plus en plus complexes en respectant les exigences des catégories syntaxiques auxquelles elles doivent appartenir, tout en s'occupant des détails grammaticaux — les accords, par exemple. Le tout forme une structure algébrique dont les mots sont les symboles d'arité nulle, et telle que chaque expression appartient à une catégorie syntaxique. Une expression syntaxique correspond ainsi à un arbre dont les nœuds sont les opérateurs abstraits utilisés et portant aux feuilles les mots qui la constitue. L'information de la manière dont les mots sont combinés, si chère au principe de compositionnalité, est exactement incarnée dans cette structure interne d'arbre, et l'utilisation d'un homomorphisme est alors le meilleur moyen d'exploiter cette information pour en dériver la sémantique de l'expression.

Il ne reste plus qu'à expliciter le langage algébrique cible, celui qui permet d'interpréter sémantiquement les expressions syntaxiques. L'objectif ici est naturellement d'obtenir des formules logiques, avec l'idée générale qu'une phrase complète doit correspondre à une ou plusieurs propositions logiques. Tout comme les mots, les autres expressions correspondent donc à des formules incomplètes, des propositions à trous qui attendent d'être complétées par de nouveaux fragments de signification. Les formules en question reposent sur un calcul de prédicats où les individus, que nous appellerons plutôt *entités* dans les présentes pages, représentent des éléments du monde réel — ceux-ci pouvant être immatériels ou abstraits —, où les propositions sont les états de faits, pouvant être vrais ou faux, et où les prédicats décrivent des propriétés imputables aux entités ou à d'autres propriétés moins complexes.

Formellement, ce calcul des prédicats s'exprime parfaitement dans le lambda-calcul simplement typé de Church [42], dont le système de types repose sur deux types de base, [2] celui des

1. On parle alors généralement de *syntagme verbal*.
2. Dans la théorie de Church, ces types sont notés ι pour les entités et o pour les propositions ; dans la suite

Partie du discours	**Catégorie syntaxique**	**Type sémantique**
Noms communs	N	$e \to t$
Syntagmes verbaux	VP	$e \to t$
Proposition	S	t
Syntagmes nominaux	$VP \to S$	$(e \to t) \to t$
Adjectifs	$N \to N$	$(e \to t) \to e \to t$
Adverbes	$VP \to VP$	$(e \to t) \to e \to t$
Verbes à comp. verbal	$VP \to VP$	$(e \to t) \to e \to t$
Déterminants	$N \to VP \to S$	$(e \to t) \to (e \to t) \to t$
Verbes transitifs	$(VP \to S) \to VP$	$((e \to t) \to t) \to e \to t$
Prépositions	$(VP \to S) \to (VP \to VP)$	$((e \to t) \to t) \to (e \to t) \to e \to t$
Pronoms relatifs sujet	$VP \to N \to N$	$(e \to t) \to (e \to t) \to e \to t$
Verbes avec conj. de sub.	$S \to VP$	$t \to e \to t$
Adverbes mod. de prop.	$S \to S$	$t \to t$
Conjonctions I	$S \to S \to S$	$t \to t \to t$
Conjonctions II	$VP \to VP \to VP$	$(e \to t) \to (e \to t) \to e \to t$
Conjonctions III	$(VP \to S) \to (VP \to S)$ $\to (VP \to S)$	$((e \to t) \to t) \to ((e \to t) \to t)$ $\to (e \to t) \to t$

TABLE 1.1 – Catégories et types traités ou sous-entendus par Montague

entités e et celui des propositions t, ainsi que sur le seul constructeur $\to$ pour les fonctions.[3] Du côté des termes, le calcul s'appuie sur un grand nombre de constantes, rassemblant trois sortes d'objets : (i) les prédicats, un prédicat n-aire étant une constante dont le type est de la forme $\alpha_1 \to \cdots \to \alpha_n \to t$, (ii) les entités référencées, de type e, permettant notamment l'interprétation des noms propres, et (iii) les connecteurs logiques, dont la nature et les types sont donnés ci-dessous :

$$\neg : t \to t \qquad \wedge, \vee, \Rightarrow : t \to t \to t \qquad \exists, \forall : (e \to t) \to t$$

Ce lambda-calcul forme le langage cible. L'homomorphisme qui assure la traduction de la syntaxe vers la sémantique se décompose alors en deux parties : la première assigne à chaque catégorie syntaxique un type sémantique, et la seconde envoie chaque expression syntaxique vers le lambda-terme qui représente son interprétation sémantique, en s'assurant d'une part que la structure interne est bien preservée, et d'autre part que le typage est bien respecté : l'image d'une expression doit avoir pour type l'image de la catégorie syntaxique de cette expression. Chaque mot est ainsi associé à un lambda-terme qui représente sa signification, et chaque opérateur abstrait est associé à un lambda-terme qui explicite comment les significations des éléments qu'il combine doivent elles-mêmes se combiner.

Du fait des limites au nombre de catégories syntaxiques possibles (car celles-ci dépendent du langage, et la quantité de formes différentes identifiées par les linguistes au fil du temps est finie), le nombre de types sémantiques utilisés en pratique dans le langage objet est lui-même fini, en dépit de l'infinité de types constructibles dans la théorie des types simples. Dans la table 1.1,

de cette thèse, nous nous appuierons plutôt sur les notations introduites et utilisées par Montague : le type e correspondra à ι, et le type t à o.

3. Montague [120] utilise en réalité un troisième type de base s, représentant les mondes possibles et destiné à l'interprétation de ses formules. Ce type est un artéfact résultant du choix de système logique, en l'occurrence la logique intensionnelle. D'autres systèmes n'ayant pas l'utilité d'un tel type, la plus grande généralité de cette discussion est obtenue en laissant ce troisième type de côté. Le cas particulier de la proposition de Montague peut néanmoins être récupérée par le biais de la procédure d'intensionnalisation décrite par de Groote et Kanazawa [51].

nous listons les catégories syntaxiques utilisées par Montague et donnons leurs types sémantiques équivalents, en suivant naturellement la relation d'homomorphisme de la syntaxe vers la sémantique. Certaines catégories reconnues par les linguistes et sous-entendues dans l'article de Montague ont été ajoutées afin d'avoir une liste un peu plus exhaustive ; il convient cependant de noter que certaines constructions syntaxiques, par exemple l'apposition, sont ignorées par ce modèle. Afin de respecter la convention de Montague, les catégories syntaxiques y sont construites à partir des trois symboles N, VP et S. Notons que certaines structures sont simplifiées dans cette approche : les verbes à complément verbal rassemblent des composés comme *try to* et les verbes avec conjonction de subordination comprennent par exemple *believe that*. Les syntagmes nominaux comprennent tout ce qui peut être sujet d'un syntagme verbal, ce qui inclut les composés déterminant-nom, les noms propres et les pronoms personnels ; les pronoms relatifs sujet couvrent des cas tels que *who* dans « *the woman who loves Bill* » [4] ; les adverbes modificateurs de proposition sont fréquents en anglais, et incluent par exemple *hopefully* ou *necessarily* ; et les trois types de conjonctions différencient la coordination (I) des propositions, comme dans « *John walks and Bill runs* », (II) des syntagmes verbaux comme dans « *John walks and eats* », et (III) des syntagmes nominaux comme dans « *John and Bill run* ».

On s'aperçoit rapidement que différentes catégories syntaxiques peuvent correspondre à un seul et même type sémantique, à commencer par les noms communs et les syntagmes nominaux eux-mêmes, qui sont tous deux associés au type des prédicats unaires du premier ordre $e \to t$. On retrouvera similairement pas moins de trois catégories différentes [5] donnant lieu au type des modificateurs de prédicats $(e \to t) \to e \to t$, regroupant les termes qui prennent un prédicat du premier ordre en argument et en retournent un nouveau. Pour seize catégories syntaxiques, nous obtenons ainsi douze types différents, et il est essentiel de remarquer que ces types sont tous des prédicats d'ordre supérieur : une fois tous les arguments fournis, le type du résultat final est toujours t. Cette omniprésence des prédicats dans l'interprétation sémantique est un aspect important de la sémantique montagovienne, et qui aura notamment valu à cette dernière certaines critiques quant à son interaction avec le sous-typage [34], sur lesquelles nous reviendrons notamment en sections 1.3 et 6.2.

Qu'en est-il maintenant des termes ? En général, à une catégorie syntaxique donnée vont correspondre un ou plusieurs opérateurs qui encadrent la composition de cette catégorie avec l'argument attendu. Un exemple notable du modèle de Montague est la composition du verbe avec son sujet : les variations en temps (et potentiellement en genre et nombre, quoique Montague n'aborde pas directement ces cas) sont chacunes gérées par un opérateur différent. Il n'est cependant pas impossible que plusieurs de ces opérateurs puissent avoir la même image via l'ho-

4. Dans [120], Montague simplifie grandement les propositions relatives par l'utilisation de *such that*, de sorte que les pronoms relatifs sujet et objet sont encodables par la combinaison de *such that* avec une proposition où la position correspondante est occupée par un pronom personnel, qui est ensuite accordé en fonction du nom que la relative complète, l'exemple précédent devenant ainsi « *the woman such that she loves Bill* ». Le cas sujet se prête bien à la généralisation proposée en table 1.1 car le pronom n'a besoin que d'un syntagme nominal comme argument, ce qui donne une catégorie syntaxique relativement simple. Nous omettons dans la table le cas objet, qui dans cette approche est plus difficile à construire sans recourir à des grammaires catégorielles combinatoires ou à la notion syntaxique de *mouvement*. Il pourrait néanmoins être approximé par une catégorie demandant un verbe transitif et un syntagme sujet, et retournant un modificateur de nom, soit une catégorie $((VP \to S) \to VP) \to (VP \to S) \to N \to N$, dont nous nous épargnerons la traduction en type sémantique par souci de lisibilité.

5. On pourra s'étonner que les adverbes et les verbes à complément verbal soient considérés comme différents alors que leur catégorie syntaxique est $VP \to VP$ dans les deux cas. Cette distinction est en réalité plus linguistique que mathématique, et est effacée par l'utilisation de la notation $\to$ au sein des catégories syntaxiques. La notation utilisée par Montague, plus proche des grammaires catégorielles, distinguait deux signes différents B/A et $B//A$, que nous simplifions ici tous deux par $A \to B$.

momorphisme d'interprétation syntaxique, en supprimant par exemple la distinction de personne et de nombre à un temps fixé en considérant que l'information nécessaire est contenue dans le syntagme sujet. Ces opérateurs prennent tous en arguments deux expressions VP et $VP \to S$, et forment une expression de type S. Par homomorphisme, leurs images sont des lambda-termes $((e \to t) \to t) \to (e \to t) \to t$. Le plus simple qu'on puisse imaginer est l'opérateur de composition $\lambda P \lambda u.Pu$, qui applique simplement le premier argument au deuxième. Nombre d'opérateurs sont généralement interprétés de cette manière. Ce n'est cependant pas toujours exactement le cas, à l'image de l'opérateur $\lambda P \lambda u.\mathcal{H}(Pu)$, où $\mathcal{H} : t \to t$ est le connecteur logique temporel exprimant le passé dans la logique intensionnelle de Montague, utilisé pour la composition du verbe au passé avec son sujet.

La contrainte d'homomorphisme donne cependant lieu à une discipline particulière d'interprétation des mots signifiants du langage : si chacun d'eux est associé à une constante (entité ou prédicat) du lambda-calcul sémantique, le type de cette constante diverge fréquemment du type attendu pour l'interprétation du mot auquel elle se rapporte, ce qui conduit généralement à englober ces mots dans une machinerie compositionnelle plus complexe. Un premier exemple révélateur est celui des syntagmes nominaux, dont le type attendu est $(e \to t) \to t$: cette catégorie syntaxique capture entre autres les noms propres, auxquels on associe des constantes de type e ; un plongement de ces constantes dans des termes de type $(e \to t) \to t$ est donc nécessaire pour rendre correctement compte de leur sémantique dans le système de Montague. Si $\mathbf{b} : e$ est la constante associée au nom propre *Bill*, alors l'interprétation de ce mot sera $\lambda u.u\mathbf{b}$. Soit $\mathbf{walk} : e \to t$ l'interprétation du mot *walk* (dans ce cas, le type attendu pour la catégorie VP et la constante associée coïncident) ; notons F l'opérateur syntaxique de composition du syntagme verbal avec son sujet à la troisième personne du singulier du présent, et notons $\mathscr{L}$ l'homomorphisme syntaxico-sémantique. Alors, la phrase « *Bill walks* » s'interprète dans le modèle de Montague de la façon suivante :

$$
\begin{array}{ccc}
\text{SYNTAXE} & & \text{SÉMANTIQUE} \\
F\ Bill\ walk & \xrightarrow{\qquad\mathscr{L}\qquad} & (\mathscr{L}F)\ (\mathscr{L}Bill)\ (\mathscr{L}walk) \\
 & & \quad = \ (\lambda P \lambda u.Pu)\ (\lambda v.v\mathbf{b})\ \mathbf{walk} \\
 & & \quad \to_\beta^* (\lambda v.v\mathbf{b})\ \mathbf{walk} \\
 & & \quad \to_\beta \mathbf{walk}\ \mathbf{b}
\end{array}
$$

Ainsi, la construction du système est telle que plusieurs étapes de β-réduction sont nécessaires pour obtenir la simple application du prédicat à son sujet dans l'interprétation sémantique. Dans ce cas particulier, la complexité des termes utilisés s'explique par la volonté de Montague de donner une interprétation uniforme de tous les syntagmes nominaux, alors même que ces derniers sont grammaticalement variés, comprennant notamment les noms propres, les pronoms personnels, et les composés déterminant-nom, par exemple « *every man* ». Leur interprétation sous le type $(e \to t) \to t$ en fait des *quantificateurs généralisés* (on notera en effet que ce type est le même que celui assigné aux constantes $\forall$ et $\exists$ plus haut), ce qui permet notamment de rendre compte des aspects logiques apportés par les déterminants dans les composés déterminant-nom ; le traitement uniforme de ces syntagmes facilite plusieurs opérations, notamment la coordination comme dans « *John and every man* ». Cette approche des syntagmes nominaux a néanmoins de quoi frustrer, dans la mesure où l'on aurait apprécié d'appliquer directement **walk** à la constante **b** dans l'interprétation plus haut, s'épargnant ainsi plusieurs étapes de calcul. De fait, ce choix de Montague a été longuement débattu, conduisant à des principes de *changements de type* [131, 129] sur lesquels nous aurons l'occasion de revenir en sections 1.3 et 6.5.

Si les syntagmes nominaux n'ont pas l'apanage de la complexité du système, certaines catégories syntaxiques se révèlent complexes justement parce qu'elles dépendent de ces syntagmes. C'est notamment le cas des verbes transitifs : un verbe transitif simple, par exemple *love*, est associé à une constante **love** : $e \to e \to t$ dont le type diverge du type attendu $((e \to t) \to t) \to e \to t$ d'après la table 1.1. La comparaison des deux types permet cependant de se rendre compte que cette divergence résulte du premier argument, qui est passé d'une entité à un quantificateur généralisé. Le lambda-terme correspondant présente en conséquence le schéma si caractéristique où le premier argument est appliqué au reste du terme : pour *love*, on écrit alors $\lambda P \lambda x.P\,(\mathbf{love}\,x)$. Un autre cas de divergence entre constantes et termes qui n'implique pas le cas des syntagmes nominaux est celui des adjectifs, dont les constantes sont aussi des prédicats unaires du premier ordre, par exemple **heavy** : $e \to t$. En cela, les adjectifs sont donc similaires aux noms ; mais, à la différence de ces derniers, leur interprétation sémantique est un modificateur de prédicat $(e \to t) \to e \to t$. Le schéma général pour les termes est donc de compléter le prédicat argument par un connecteur logique lui conjoignant la constante associée à l'adjectif : ainsi, l'interprétation de *heavy* s'écrit $\lambda u \lambda x.\mathbf{heavy}x \wedge ux$. [6]

Depuis les propositions initiales de Montague, habitude a été prise de remplacer dans les catégories syntaxiques la primitive *VP* des syntagmes verbaux par la primitive *NP* des syntagmes nominaux, et de définir les syntagmes verbaux par la catégorie $NP \to S$. Les nouvelles catégories syntaxiques sont obtenues à partir de celles listées en table 1.1 en substituant toutes les occurrences de $VP \to S$ par *NP*, puis toutes les occurrences restantes de *VP* par $NP \to S$. On conserve alors la traduction $\mathscr{L}(NP) = (e \to t) \to t$ assignée aux syntagmes nominaux, qui se propage ensuite à toutes les autres catégories. Il en résulte des types sémantiques encore plus complexes que dans la table précédente, où certains types e en argument se voient élevés au type $(e \to t) \to t$ [7] ; en particulier, le type assigné aux syntagmes verbaux passe de $e \to t$ à $((e \to t) \to t) \to t$, et l'interprétation de verbe intransitifs tels que *walk* passe alors de la simple constante **walk** au terme élevé $\lambda P.P\,\mathbf{walk}$. Une transformation similaire s'opère notamment pour le second argument des verbes transitifs.

De cette présentation détaillée du formalisme de Montague, nous retiendrons trois choses. Tout d'abord, l'organisation du système met en lumière l'unidirectionnalité de l'analyse à l'interface syntaxe-sémantique : pour pouvoir calculer correctement l'interprétation sémantique d'une phrase, il est nécessaire (selon le principe de compositionnalité) d'utiliser l'information syntaxique de l'ordre et la manière dont les mots de la phrase sont combinés — autrement dit, la sémantique dépend de la syntaxe : on dira alors que l'analyse sémantique est *dirigée par la syntaxe*. Nous retrouvons ce fait dans l'homomorphisme de traduction, qui est bien défini dans la direction attendue, et sur lequel aucune hypothèse d'inversibilité n'est possible puisque, comme attesté par la table 1.1, cette application n'est pas même injective. Une conséquence directe de

6. Ce modèle de l'adjectif est néanmoins grandement simplifié, dans la mesure où il ignore les considérations liées à la classification des différentes sortes d'adjectifs, comme décrit par exemple dans [92, 130]. Ces travaux suggèrent en particulier de considérer l'introduction directe pour certains adjectifs de constantes $(e \to t) \to (e \to t)$ afin de traduire une modification du prédicat argument plus profonde que ne le permettent les connecteurs logiques. Ceci est notamment défendu pour les adjectifs *subsectifs*, à l'instar de *good* dans « *a good violonist* » : être un bon violoniste n'implique pas qu'on soit bon tout court, mais seulement qu'on est violoniste. Le cas de *heavy* lui-même est plus complexe qu'il n'y paraît de par sa nature d'adjectif *scalaire* : son interprétation nécessite des lois d'échelles qui dépendent du nom qu'il complète. Ainsi, une fourmi lourde ne fait pas partie des entités qu'on reconnaîtrait généralement comme lourdes : elle n'est lourde que *par rapport* aux autres fourmis, une subtilité que le lambda-terme donné précédemment n'est pas en mesure de capturer. Notons toutefois que toutes ses considérations ne remettent pas en cause le typage assigné aux adjectifs.

7. La transformation e vers $(e \to t) \to t$ apparaît dans la littérature sous le nom d'*élévation de type* (« *type raising* ») et est l'un des axes majeurs du concept de changement de type introduit par Partee [131, 129]. C'est ce principe qu'utilisa implicitement Montague afin d'assurer l'uniformité de type aux syntagmes nominaux.

cette observation est que le pouvoir de combinaison des fragments de signification ne peut reposer sur la sémantique seule, et donc en particulier sur les types sémantiques seuls. Deuxièmement, la plupart des mots signifiants de la langue considérée se voient assigner des interprétations simples dans le langage sémantique, sous forme de prédicats ; mais la nécessité de conserver la propriété homomorphique de l'opération de traduction conduit la plupart du temps à celle de plonger ces prédicats dans des termes de types plus complexes, la complexité en question servant principalement de mécanique compositionnelle qui, après quelques étapes de calculs, aboutit à un résultat dont on peut parfois légitimement penser qu'il aurait pu être plus rapide à calculer à partir des prédicats de départ. Nous reviendrons dans la section 1.3 sur ce « paradoxe de la complexité ». Enfin, les types sémantiques, de par leur fondements en théorie des types simples, sont structurellement hierarchisés par la distinction entre entités et prédicats. [8] Les syntagmes nominaux sont l'unique source d'entités dans les formules logiques : pour les noms propres, ces entités sont référencées sous forme de constantes de type e, et pour les pronoms et les composés déterminant-nom, ces entités sont apportées par des variables quantifiées issues des lambda-termes les interprétant. Toutes les autres expressions sont des prédicats. Ces derniers peuvent être hiérarchisés selon deux critères : d'une part leur arité, c.-à-d. le nombre d'arguments qu'ils acceptent, et d'autre part leur ordre, c.-à-d. la complexité des arguments qu'ils acceptent — les modificateurs de prédicats du premier ordre, comme les adjectifs, sont notamment du second ordre. En revanche, la logique sous-jacente, elle, reste au premier ordre, car les quantificateurs ne lient que des variables d'entités. Cette hiérarchisation reflète naturellement les relations de prédications du côté syntaxique mais, comme le démontre l'exemple des syntagmes nominaux, la correspondance entre catégories syntaxiques et types sémantiques n'est pas aussi rigide qu'elle n'y paraît, et des variations dans l'ordre des prédicats utilisés peuvent être recevables ; sans compter l'élévation de type qui permet d'identifier une entité avec l'ensemble de ses propriétés. L'arité, par constrate, se doit de refléter au minimum le nombre d'arguments obligatoires en syntaxe. [9]

Toutes ces remarques aident à mieux cerner le rôle des types sémantiques au sein de l'analyse syntaxico-sémantique. Si la sémantique dépend bel et bien de la syntaxe pour se construire correctement, la syntaxe seule ne peut suffire à guider entièrement la composition sémantique, parce qu'elle est linéaire : sa structure interne ne repose que sur les mots, et des opérateurs de composition syntaxique simples qui permettent de combiner (généralement) deux expressions pour en former une nouvelle. Cette structure, grâce aux catégories syntaxiques, donne des informations essentielles sur le nombre d'arguments des prédicats, qui est par la suite reflété par la sémantique, mais ne donne pas intrinsèquement d'informations sur la complexité des prédicats logiques sous-jacents. La distinction des différents ordres de prédicats, laquelle guide la contribution de chaque partie dans la composition sémantique finale, est le propre des types sémantiques : différents choix de typage mènent à différentes manières de construire une signification, et c'est ce choix qui permet notamment de varier la théorie logique dans laquelle ces significations sont exprimées. La présentation faite dans cette section permet la construction de formules en logique du premier ordre, a priori classique bien que la logique intuitionniste ne soit pas formellement exclue ; néanmoins, et comme indiqué à plusieurs reprises, des interprétations dans d'autres logiques comme la logique intensionnelle de Montague ou la logique

8. Cette distinction range le type des propositions comme un prédicat d'arité nulle.

9. Rien n'empêche cependant d'aller au-delà de ce nombre. La logique intensionnelle de Montague, avec son type des mondes possibles, en est un premier exemple, quoique son type n'est pas utilisé directement par les constantes de prédicat. C'est en revanche le cas dans certaines implémentations de la sémantique évènementielle (cf. [128]) où le système de types se voit enrichi d'un nouveau type de base v représentant les évènements, qui vient s'ajouter aux arguments du verbe, donnant p. ex. des constantes de la forme **love** : $v \to e \to e \to t$.

néo-davidsonienne sont accessibles par le simple ajout d'un type de base et la révision des types de certaines constantes ainsi que d'images de certaines catégories syntaxiques.

Cette dernière remarque souligne le fait que le choix de la théorie de type utilisée influence également l'interprétation sémantique. Dans les exemples explorés ici, nous sommes restés globalement dans la théorie des types simples, dont les seules briques sont les types de base et le constructeur de fonctions. Que se passe-t-il si l'on ajoute de nouvelles façon de construire des types ? Est-il possible d'adapter le paradigme de traduction syntaxe-sémantique pour utiliser ces types au sein d'un nouveau modèle, et surtout, est-il pertinent de le faire ? Comme nous allons le voir dans la suite de ce chapitre, de nombreuses propositions ont été faites ces dernières décennies pour utiliser des théories de types plus complexes dans le but de gagner toujours plus de précision dans l'interprétation sémantique. Cette nécessité s'explique évidemment par le besoin de modéliser des phénomènes sémantiques de plus en plus subtils, que nous examinons dans la section suivante.

1.2 L'intégration de la sémantique lexicale et ses implications

La sémantique lexicale est une discipline à peu près aussi ancienne que la linguistique elle-même, et comporte une histoire riche en approches et en théories diverses [71]. Dans cette section, nous nous contenterons d'analyser la manière dont certaines des approches issues de ce domaine ont été intégrées dans les modèles d'interface syntaxique-sémantique inspirés de près ou de loin par le modèle de Montague précédemment présenté, quelles nouvelles questions ont alors émergé et quelles sortes de solutions ont été proposées pour y répondre. Toutefois, il apparaît pertinent de commencer par donner ici quelques intuitions sur les motivations qui ont pu pousser à intégrer ces théories dans un modèle déjà complexe. Ces explications seront maintenues à leur degré de détail minimal ici, mais l'on en retrouvera une analyse en profondeur dans le chapitre 2.

1.2.1 Des limites au système montagovien ?

Si le système de Montague permet effectivement, grâce à l'homomorphisme de traduction, de calculer l'interprétation de n'importe quelle expression syntaxique, c'est qu'il sous-tend un principe présentant des limites indésirables : toute expression syntaxique correctement construite, c.-à-d. grammaticale du point de vue des catégories syntaxiques, peut se voir assigner une interprétation sémantique. Les limites de ce principe sont atteintes lorsque l'on s'aperçoit qu'il est très facile de construire dans un tel système des phrases qui sont grammaticalement correctes, mais qui n'ont pas de signification acceptable pour les locuteurs de la langue. Une instance connue de ce phénomène est incarnée par la célèbre phrase de Chomsky [39] « *colorless green ideas sleep furiously* » : bien que parfaitement correcte du point de vue de la syntaxe, avec un syntagme nominal composé d'un nom et de deux adjectifs et un syntagme verbal composé d'un verbe intransitif et d'un adverbe, cette phrase n'a aucun sens dans une conversation normale. [10] Pourtant, replacée dans un modèle montagovien, il est tout à fait possible de construire la représentation logique d'une possible signification de cette phrase. Cet exemple illustre ainsi le phénomène de *surgénération sémantique* du modèle de Montague, c.-à-d. la capacité du modèle à traiter comme acceptables des énoncés qui ne le sont normalement pas sémantiquement parlant. Une extension de ce problème a trait au rejet des énoncés : un tel modèle est en effet incapable de distinguer le faux de l'absurde.

10. Une étude plus détaillée de cette phrase dans le chapitre 2 montrera d'ailleurs que les raisons de cette absence de signification sont multiples et n'appartiennent pas toutes au même degré d'analyse. Cette remarque, importante pour la suite, est cependant facultative dans la présente discussion.

Lutter contre ce phénomène de surgénération implique de poser des limites à la compositionnalité des mots selon des critères sémantiques. Or, il appartient justement à la sémantique lexicale d'étudier la signification des mots, leur évolution et leur classification : s'il faut trouver de telles limites, c'est probablement dans ce domaine qu'il faudra les chercher. L'idée générale serait d'avoir à notre disposition une manière de distinguer les combinaisons de mots qui sont sensées de celles qui ne le sont pas. Plus concrètement, cela suppose d'avoir une classification des mots qui se combinent correctement entre eux, ou alors de disposer pour chaque mot d'une banque d'informations listant notamment les contraintes qui pèsent sur le pouvoir combinatoire du mot. On appelle *lexique* une telle liste d'informations (cf. notamment [139] pour une introduction générale, voir aussi notre discussion en sous-section 3.1.3). Le lexique cristallise l'apport de la sémantique lexicale dans les modélisations formelles de la langue : d'une façon ou d'une autre, les informations qu'il contient doivent être incorporées dans un système similaire à celui de Montague. Mais, si l'on admet avoir un tel lexique sous la main, à quel endroit précis au sein du modèle de Montague devrions-nous incorporer ses informations ? Une première idée serait de raffiner les catégories syntaxiques pour distinguer davantage de cas différents : par exemple, séparer les noms en distinguant les êtres animés, les objets et les concepts abstraits (cf. p. ex. [3, 40]). Une telle approche risquerait cependant d'introduire le phénomène inverse de *sous-génération* (si tant est qu'il ne fût pas déjà présent dans le système initial) : si la classification est trop restrictive, certaines phrases pourtant correctes du point de vue sémantique (voire syntaxique) pourraient ne plus être représentables dans le système (cf. [96]). Une seconde option serait alors d'introduire cette information du côté sémantique, en ajoutant des contraintes supplémentaires à satisfaire lors des applications du lambda-calcul sémantique.

Dans le modèle initial, la seule contrainte qui pèse sur la composition au sein de ce lambda-calcul est celle du type sémantique, dont la structure doit être respectée. Néanmoins, le recours à l'homomorphisme de traduction assure que cette contrainte est toujours respectée par défaut. En cela, l'utilisation des types sémantiques, bien qu'importante d'un point de vue théorique pour la distinction des entités et des différents ordres de prédicats, est presque anecdotique puisqu'elle n'apporte aucune contrainte supplémentaire par rapport à la syntaxe : son seul intérêt a priori est d'assurer que l'homomorphisme de traduction soit cohérent ; mais une fois cet homomorphisme construit, les types sémantiques n'ont plus aucune utilité pratique et peuvent être passés sous silence. Qu'en est-il lorsque l'on cherche à intégrer l'information lexicale ? Il s'agirait de pouvoir vérifier, avant toute opération de β-réduction, si le lexique ne contient aucune contre-indication à la composition des mots utilisés, donnant ainsi la garantie que le résultat réduit sera bien l'interprétation d'une phrase sensée. On voudrait en particulier d'appliquer l'idée que chaque mot comporte un certain nombre de *présuppositions*, certaines intrinsèques, c.-à-d. portant sur sa signification propre, et d'autres portant sur les arguments des prédicats : pour donner une signification correcte, les présuppositions du mot passé en argument doivent s'accorder avec les attentes du prédicat. Par exemple, un verbe comme *marcher* présuppose que son sujet soit capable de se mouvoir (du moins dans l'acception littérale du mot), et un nom comme *homme* satisfait ces présuppositions, rendant ainsi la combinaison « l'homme marche » sémantiquement correcte ; alors qu'un nom comme *table* ne remplit pas ces présuppositions, et la phrase « ?la table marche » est sémantiquement bancale. [11] Au fil du temps, l'idée d'intégrer ces présuppositions au sein du lambda-calcul a pris la forme d'annotations supplémentaires au niveau des types et,

11. Par « sémantiquement bancale », nous cherchons à souligner que l'énoncé, quoique grammatical, n'a pas de sens littéral applicable au monde réel. Il est bien sûr toujours possible de trouver des contextes où la phrase a du sens, que ce soit par métaphore, ou bien dans un monde imaginaire (« dans ce dessin animé, la table marche »). Par défaut, nous utiliserons toujours ce concept, si imprécis soit-il, de *monde réel* pour tenter d'interpréter sémantiquement les énoncés.

plus précisément, de subdivisions du type e des entités en sous-types offrant une classification plus dense.

Cette nouvelle approche rend toute son importance aux types sémantiques dans le modèle montagovien : au lieu d'outils théoriques dont les conditions sont toujours remplies en pratique, on se retrouve avec des contraintes plus fortes qu'il est désormais impossible d'ignorer, et qu'il convient alors de vérifier avant toute opération de β-réduction. Ceci rajoute une étape supplémentaire dans l'analyse sémantique, entre la traduction de la syntaxe par homomorphisme et le calcul de la représentation sémantique finale, au cours de laquelle on procède à la *vérification des types*, c.-à-d. qu'on observe si, pour chaque application dans le lambda-calcul sémantique, le type de l'argument et celui attendu par le prédicat s'accordent. Lorsque ce n'est pas le cas, on parlera d'*erreur* ou de *discordance de types*, et ce phénomène indique en général que l'énoncé originel est sémantiquement bancal. Notons cependant que lors de la traduction la structure des types sémantiques, c.-à-d. leur ordre et leur arité, n'est pas modifiée par rapport au système initial, maintenant la moindre importance de cette contrainte comme observé précédemment. Seuls changent les occurrences des types e, remplacés par divers nouveaux types qui correspondent toujours à des entités, et n'introduisent donc pas de modifications d'ordre et d'arité. Pour ce faire, l'homomorphisme de traduction est enrichi d'un lexique qui associe à chaque mot le ou les types élémentaires attendus en remplacement des types e initiaux. Par ailleurs, il est essentiel de noter que ces nouveaux types sont fréquemment ordonnés, au moins partiellement, pour traduire des implications de contrainte, d'où notamment l'usage du terme *sous-type* pour les désigner. Par exemple, imaginons qu'il y a un type a encodant les contraintes de mobilités nécessaires pour marcher : on aurait pour le mot *walk* une constante $\mathbf{walk} : a \to t$, où le type a remplace effectivement le type e par rapport à l'exemple donné en section 1.1. Supposons de surcroît disposer d'un type h encodant les contraintes qu'il pourrait y avoir à être humain. Ces dernières contraintes contiennent en particulier celles de pouvoir marcher, ce qui signifie qu'être humain est moins général, plus précis, que pouvoir marcher : cette implication est alors encodée par une relation de sous-typage $h \leq a$. Pour un nom propre comme *Bill*, on aurait alors la constante $\mathbf{b} : h$. En dérivant l'interprétation de l'énoncé « *Bill walks* », on espère alors se retrouver dans une situation où le prédicat **walk** s'applique à la constante **b** : en vérifiant les types, on s'aperçoit que le prédicat attend un argument de type a, mais qu'un type h est fourni. Cependant, comme h est un sous-type de a, les contraintes qui pèsent sur le prédicat sont bien satisfaites, et la composition reste valide.

1.2.2 Lexique, types, et la répartition de l'information sémantique

Malheureusement, aussi simple que cette solution puisse paraître de prime abord, elle apporte aussi avec elle un nouveau lot de difficultés à traiter, et non des moindres puisqu'elles ont occupé une grande partie des débats en sémantique formelle au cours des trentes dernières années. L'une des premières intégrations concrètes de la sémantique lexicale au sein d'une approche formelle est due à James Pustejovsky, avec sa théorie du lexique génératif (en anglais *Generative Lexicon*, abrégé en GL, cf. [135]). C'est notamment à cette théorie que l'on doit la notion de lexique introduite au début de cette section. L'objectif de Pustejovsky était de fournir une base théorique capable de rendre compte de la capacité créative du langage : les mots ne peuvent pas être simplement décrits par une liste de sens comme dans un dictionnaire, mais possèdent une base de sens qui leur est propre et qui est régie par une structure interne permettant de dériver des sens différents en fonction du contexte. Pustejovsky [135, chap. 4] cite entre autres l'exemple du mot *good*, dont le sens évolue en fonction du nom qu'il complète : un violoniste n'est pas bon de la même manière qu'un repas est bon, ou qu'une voiture est bonne, etc. Par conséquent, l'entrée

d'un mot dans un lexique rassemble une certaine quantité d'informations de base, sur lesquelles il est possible de s'appuyer pour former une grande variété de significations. Et l'une des premières choses à constater à cet égard est que Pustejovsky y place les informations sur la structure argumentale des mots, c.-à-d. les arguments qu'il attend, soient-ils syntaxiquement réalisés ou non. [12] En particulier, l'information lexicale capture les spécifications qui décrivent les constantes dans le système de Montague. À travers cette approche, il est possible de concevoir le lexique non pas comme un ajout à l'homomorphisme de traduction, mais comme un remplacement de cet homomorphisme par un nouveau paradigme de traduction vers le langage sémantique.

La structure argumentale de Pustejovsky fait usage de types sémantiques élémentaires pour caractériser les présuppositions associées à chacun des arguments du mot décrit par l'entrée lexicale. Ces types sont complétés par une structure d'information supplémentaire, les *qualia*, qui précisent certaines caractéristiques intrinsèques à la signification du mot qui ne sont pas directement encodables dans les types eux-mêmes. Ceci fait alors émerger la première grande difficulté de cette approche : quelles sont les présuppositions encodables dans les types sémantiques, et quelles sont les informations qui échappent à cet encodage ? Considérons par exemple les mots *homme* et *femme* : tous deux désignent plus généralement des êtres humains, distingués sur la base de leur genre. Doit-on les distinguer sémantiquement par deux types élémentaires différents *man* et *woman*, ou peut-on se contenter d'utiliser le type h des humains et caractériser la nuance à l'aide de prédicats supplémentaires ? Dans un système prédicatif comme celui de Montague, un mauvais choix de caractérisation peut aboutir à une représentation redondante, comme par exemple un prédicat $\mathbf{man} : \mathit{man} \to t$. Il y aurait alors un type et un prédicat pour représenter la même chose d'un point de vue compositionnel et d'un point de vue logique ! Dans un souci d'efficacité et de simplicité, il serait souhaitable de pouvoir éviter ce type de redondances si elles ne sont pas nécessaires, ce qui nous ramène à l'idée d'être capable de distinguer quelles propriétés intrinsèques d'un mot constituent son type et quelles autres propriétés ne doivent pas y figurer. Pustejovsky fait par exemple le choix d'utiliser le type h des humains et de placer le prédicat de genre dans un quale qu'il nomme *constitutif* ; dans le système de Montague, cela revient par simplification à définir un prédicat $\mathbf{man} : h \to t$ enrichi possiblement d'une implication $\forall x.\mathbf{man}(x) \Rightarrow \mathbf{male}(x)$ pour récupérer l'information de genre.

On retrouve ce premier problème dans la gestion d'un phénomène sémantique très important dans le présent travail : la *polysémie*. Il s'agit du phénomène par lequel un même mot, placé dans des contextes différents, exhibe des significations différentes mais liées entre elles, qu'on appelle fréquemment des *aspects*. On en trouvera plusieurs exemples dans [123], parmi lesquels les bien connus *newspaper* et *book* : en fonction du contexte, le premier peut désigner soit le journal comme publication physique, soit l'institution qui écrit, édite et publie ces copies physiques ; et le second peut désigner soit le livre comme objet, soit son contenu, commun à toutes les copies du livre en question. [13] Des exemples impliquant chacun de ces aspects, issus de [123], sont reproduits ci-dessous.

(1) a. The newspaper weighs five pounds.
b. The newspaper fired John.

12. Cette dernière catégorie comprend ce que Pustejovsky appelle les *arguments par défaut*, c.-à-d. des arguments qui sont sous-entendus en l'absence de complément pour occuper sa place (par exemple, « *John built the house (out of bricks)* »), ainsi que les *arguments fantômes*, qui sont incorporés dans la sémantique du mot et ne sont réalisés que s'ils apportent des précisions dessus, comme p. ex. *beurrer* qui correspond à « tartiner avec du beurre », mais autorise des précisions du type « beurrer avec du beurre salé » ; cf. [135, §5.2].

13. Pustejovsky [135], contrairement à Nunberg [123] qui n'évoque pas cette possibilité, reconnaît pour *newspaper* un troisième aspect similaire au second aspect de *book* pour le contenu informationnel. Par souci de simplicité, nous conservons dans la présente discussion l'approche de Nunberg.

(2) a. The book weighs five pounds.
b. The book has been refuted.

Les mots polysémiques se distinguent des pures homonymies par la relation que leurs aspects peuvent avoir entre eux. Pour *newspaper*, il y a un objet physique d'un côté et une institution de l'autre, mais l'institution produit l'objet ; de même, le premier aspect de *book* est un objet physique, et son second est un concept informationnel, auquel on accède à travers l'objet, par la lecture en particulier. Par contraste, le mot *bank* a deux significations pouvant se traduire en français par *banque* et *berge* respectivement, et il est clair qu'aucune relation logique ne peut unire ces deux significations de manière satisfaisante, celles-ci étant trop éloignées l'une de l'autre : par conséquent, *bank* est un cas d'homonymie et pas de polysémie. Une autre manière de reconnaître la polysémie est de vérifier sa propension à la *coprédication*, c.-à-d. la possibilité d'accéder à plusieurs de ses aspects à partir d'une seule occurrence du mot, comme illustré par les exemples ci-après (le premier étant issu de [123] et le second de [4, §1.2]). Dans le premier exemple, le mot *newspaper* se rapporte à la fois aux deux aspects précédemment évoqués, l'un mobilisé par *decide*, et l'autre par *its format* ; et dans le second, *lunch* se rapporte à la fois à la nourriture (*delicious*), et à l'évènement au cours duquel elle est consommée (*take forever*).

(3) a. The newspaper has decided to change its format.
b. The lunch was delicious but took forever.

L'une des idées majeures introduites par Pustejovsky [135] au sujet des mots polysémiques est l'expression des différents aspects au travers d'un constructeur de type spécifique $\bullet$, les types utilisant ce constructeur étant alors appelés *types pointés*. Chaque aspect d'un mot polysémique a ses propres contraintes et ses propres présuppositions, et ces différents aspects, quoique liés entre eux par des relations logiques, demeurent suffisamment différents de ce point de vue pour se voir assignés des types différents, et surtout *incompatibles*, c.-à-d. qu'aucun des types impliqués n'est le sous-type d'un autre. Un type pointé exprime alors le type d'un mot polysémique comme le « produit » des types de ses différents aspects. Ainsi, si les types *phys*, *institution* et *info* représentent respectivement les entités physiques, les institutions et les concepts informationnels, alors le mot *newspaper* sera associé au type $\mathit{phys} \bullet \mathit{institution}$, et le mot *book* sera associé au type $\mathit{phys} \bullet \mathit{info}$.

Dans GL, les mots polysémiques (souvent appelés *objets pointés*) sont représentés par une collection d'arguments modélisant les différents aspects du mot, enrichie par des qualia qui décrivent de façon plus détaillée les relations entre eux. À cet égard, il est important de souligner qu'un objet pointé n'est pas défini uniquement par son type, mais également par la relation que ses aspects entretiennent entre eux. Des mots différents sont tout à fait susceptibles d'exhiber des aspects de mêmes types, mais reliées par des relations différentes. Une note de Pustejovsky [137] liste un grand nombre de cas de polysémie, chaque complexe de types regroupant un ou plusieurs mots, p. ex. *book* et *CD* pour le type $\mathit{phys} \bullet \mathit{info}$. Pourtant, si dans les deux cas l'information est contenue dans l'objet physique, la nature de ce stockage et de la manière dont on accède à l'information diffère : pour l'un, cela se fait par la lecture, et pour l'autre, par l'écoute. [14] Mais la composition de ces structures à l'échelle d'une phrase requiert une discipline de manipulation des types plus complexe que celle de la théorie des types simples utilisée par Montague. Plusieurs

14. Il est plus difficile de se rendre compte de cette subtilité en français du fait que le mot *lecture* s'applique aussi bien aux livres qu'aux disques. En anglais, la distinction est plus claire entre *reading* pour les livres et *playing* pour les disques ; quoique l'utilisation de *reading* est attestée pour les disques dans le cas de l'extraction de données informatiques, par opposition à la musique. Il est possible que cette distinction tienne plus précisément à la nature de l'information elle-même.

implémentations et théories pour la manipulation des types pointés ont ainsi été proposées et discutées, notamment à travers la logique de composition de types d'Asher et Pustejovsky (en anglais *Type Composition Logic*, abrégé en TCL) [5, 4], ou bien comme extension des théories modernes des types (*Modern Type Theories*, MTT) défendues notamment par Luo [107]. La question du traitement de la polysémie et des cas de coprédication est désormais essentielle dans les systèmes de sémantique formelle, et est intrinsèquement liée à la question des types sémantiques. En particulier, toute réponse apportée à notre premier problème de déterminer quelles propriétés sont encodables sous forme de types sémantiques élémentaires devra être capable de reproduire le comportement des objets pointés.

1.2.3 Vers une théorie des coercions

Dans le cadre d'une théorie sémantique formelle, la polysémie telle qu'introduite précédemment implique l'accès à un certain nombre d'opérations capable d'adapter la correspondance des types au moment de la composition, afin d'exploiter la relation logique qui unit les différents aspects de l'argument entre eux. Ainsi, si un prédicat attend un argument de type *phys* et est appliqué à un argument de *phys* • *info*, on s'attend à disposer d'un mécanisme théorique capable de sélectionner l'aspect correspondant. Ce mécanisme n'est pas sans rappeler celui du sous-typage décrit quelques paragraphes plus haut : si l'on approxime (à des fins d'exposition uniquement) les types comme un ensemble de contraintes, il s'agit alors de sélectionner un sous-ensemble de contraintes qui correspond à celles attendues par le prédicat ; tandis que pour les types pointés, il s'agit de d'abord choisir un ensemble de contraintes parmi plusieurs possibles, représentant les aspects. Les deux cas consistent ainsi en des opérations capables de projeter un type élémentaire vers un autre. De ce point de vue, ils forment deux instances d'un phénomène encore plus général appelé *coercion*.[15]

Cette notion de coercion désigne de façon générale une opération de remplacement d'un type par un autre. La définition qu'en donne Pustejovsky est la suivante :

> a semantic operation that converts an argument to the type which is expected by a function, where it would otherwise result in a type error. [135, §4.6]

Le phénomène de coercion intervient donc spécifiquement dans un cadre compositionnel, pour corriger un « mauvais » argument lors de l'application d'un prédicat. Sémantiquement, il s'agit de convertir l'information contenue dans la représentation de l'argument en une information utilisable par le prédicat auquel il est passé. Pour mieux comprendre en quoi une telle opération consiste, il peut être utile — et nous le ferons à plusieurs reprises dans la présente thèse — de dresser une analogie entre la linguistique et une autre forme de système de signes interprétés : les langages de programmation. Ce n'est en effet pas un hasard si l'interprétation de ces langages partage avec la linguistique le terme de « sémantique » : dans les deux cas, il s'agit de convertir un système de mots (les *signes*) en expressions mathématiques décrivant leur signification. Les langages de programmation cependant, étant des constructions humaines ad hoc, sont généralement bien plus formalisés et par là même plus facile à interpréter que les langues naturelles. Pourtant, le principe général de leur interprétation reste le même : tout repose sur une

15. Nous privilégierons dans la suite le terme de *coercion* plutôt que celui de *coercition*, tout d'abord parce que différents dictionnaires modernes reconnaissent *coercion* comme une variante recevable de *coercition*, et que *coercion* est par ailleurs attesté en français comme une forme vieillie qu'on retrouvera notamment dans le Littré, ce qui valide l'usage de ce terme autrement que comme un anglicisme ; et ensuite parce que le terme de *coercion* dans son usage technique en linguistique est déjà suffisamment attesté en français pour être référencé comme tel dans certains dictionnaires (cf. p. ex. sur le Wiktionnaire : `https://fr.wiktionary.org/wiki/coercion`) : par conséquent, l'utiliser comme tel dans la présente thèse revient donc simplement à maintenir voire propager un nouvel usage, fruit de l'évolution continuelle de la langue française.

application qui traduit la syntaxe, c.-à-d. une collection organisée de signes, vers la sémantique, c.-à-d. une interprétation mathématique plus abstraite (cf. p. ex. [178]). De surcroît, les langages de programmation utilisent également des types qui décrivent les propriétés intrinsèques et contraintes des valeurs manipulées.

Considérons donc pour l'exemple un langage typé comprennant le type `string` des chaînes de caractères, et le type `int` des nombres entiers. Une fonction comme `print` prend en argument une valeur de type `string` et l'affiche à l'écran ; que se passe-t-il alors si l'on souhaite lui passer la valeur `3`, de type `int` ? Intuitivement, tout nombre entier a une manière canonique de s'écrire comme une chaîne, grâce aux caractères de chiffres et au tiret pour les nombres négatifs. En exploitant cette propriété, on peut définir une opération de conversion qui produit une valeur de type `string` à partir d'une de type `int`, tant et si bien que l'application `print(3)` peut effectivement afficher quelque chose malgré la discordance de types évidente. [16] Cette opération de conversion décrit exactement l'idée induite par la notion de coercion : en cas de discordance de types, on cherche à exploiter les informations de l'argument initial pour obtenir une valeur annexe d'un autre type à utiliser comme argument de substitution. Ce nouvel argument doit naturellement dépendre de l'argument initial et, dans les langages de programmation tout du moins, doit être calculé de manière relativement uniforme : il serait peu opportun de convertir `3` en `"3"` ou en `"three"` selon la fonction utilisée sans possibilité de contrôle sur le type d'opération menée. Ceci nous donne une indication supplémentaire sur le fonctionnement général d'une coercion : il s'agit d'une opération qui doit être suffisamment « naturelle » pour s'appliquer dans un maximum de contextes possibles.

On retrouve évidemment ces propriétés dans les langues naturelles, où une coercion signifie qu'on substitue implicitement une signification pour une autre signification qui lui est logiquement reliée. Ces transformations peuvent prendre de très nombreuses formes (Pustejovsky [138] recense pas moins de cinq opérations de coercion différentes) et servent de base à la richesse des langues naturelles. [17] Dans la suite, nous donnons une série d'exemples destinés à illustrer au mieux la nature multiple de ce phénomène, à commencer par les énoncés (4) où le lien logique entre le mot initial et ses dérivés coercés demeure assez évident.

(4) a. John left the party.
b. Mary taught before the party.

Ces premiers exemples, issus de [138], reposent sur l'idée que *party* décrit un évènement, et a en conséquence un type *event* associé. Or, le verbe *leave* exige en argument un lieu de type *location*, tandis que *before* doit être suivi d'un référent temporel de type *time* : dans les deux cas, la signification initiale est coercée vers une valeur d'un nouveau type. Mais ces nouvelles valeurs sont intuitivement simples à déduire de la signification de départ : un évènement, quel qu'il soit, est toujours associé à un lieu (celui où il se déroule) et à une période (le moment où il se déroule) ; de sorte que la coercion dans ces exemples est naturelle, exploitant les informations sous-entendues par la notion même d'évènement. [18] Il est d'ailleurs intéressant de remarquer que

16. Relevons tout de même que ce résultat dépend pragmatiquement de l'importance des types au sein du langage de programmation. Un langage à typage statique fort comme OCaml refusera à tous les coups l'expression `print_endline 3` à cause de la discordance de types, là où l'expression équivalente en Python ou en C fonctionnera sans problème. En OCaml, il est nécessaire d'ajouter explicitement la conversion sous la forme d'une nouvelle fonction comme dans `print_endline (string_of_int 3)`. Cette manière de fonctionner préfigure la notion de coercion explicite que nous introduisons un peu plus loin.

17. Bien que nous fournissons dans ces pages des exemples principalement en anglais, précisons au passage que ce phénomène se retrouve dans d'autres langues, comme en français [141], en néerlandais [6], et même dans des langues non-indo-européennes comme le mandarin [193].

18. Pustejovsky et Rumshisky [143] caractérisent ce comportement par le double rôle des types *location* et *time*

les phrases en (4) fonctionnent également avec l'objet pointé *lunch*, dont l'un des aspects est bel et bien de type *event*. Cela souligne que la sélection d'aspect peut elle-même être vue comme une forme de coercion qui s'accorde parfaitement avec les autres, appuyant ainsi la propriété fonctionnelle des coercions dans leur capacité à se composer entre elles. [19]

Une autre forme de coercion, plutôt que d'exploiter les informations sous-entendues par le type initial comme pour les objets pointés ou les attributs, impose au contraire des informations supplémentaires à un type initialement plus simple. Outre l'introduction de qualia qui constitue une part importante des travaux de Pustejovsky, on y retrouve aussi l'introduction de types pointés. Considérons les exemples suivants :

(5) a. Mary read the rumor.
b. Mary read the screen.

Le mot *rumor* désigne normalement une information pure, un fragment de savoir qu'on peut recevoir et diffuser de multiples manières ; il se voit donc atrribuer le type *info*. Par contraste, le mot *screen* désigne une installation physique sans présupposition particulière ; il reçoit donc le type *phys*. Cependant, le verbe *read* est présenté par Pustejovsky comme un verbe plus complexe, mêlant d'une part un aspect informationel (la réception de connaissance externe) et d'autre part un aspect de perception physique (lire suppose un support sur lequel l'information est écrite) : par conséquent, le type attendu pour l'argument objet de *read* est le type pointé $phys \bullet info$. Que se passe-t-il alors dans les énoncés en (5) ? Dans le premier cas, *rumor* est coercé du type *info* au type $phys \bullet info$: intuitivement, on introduit un support physique sous-spécifié sur lequel Mary a été capable de lire la rumeur mise sous forme écrite. Dans le second cas, la coercion appliquée a converti *screen* de son type initial au même type pointé, introduisant ainsi une information sous-spécifiée qui serait affichée — et donc lisible — à l'écran.

Un dernier cas plus connu de ce phénomène, que Pustejovsky appelle *coercion par introduction*, est celui de la coercion aspectuelle. Il s'agit de situations dans lesquelles un nom est converti en une activité qui l'implique en tant que participant, initiées généralement par un verbe dit *aspectuel*. Nous reproduisons ci-dessous les exemples donnés dans [138, §3.4].

(6) a. Mary began the book.
b. Mary enjoyed her coffee.

Dans les deux cas, un syntagme nominal est fourni là où une activité est attendue : à chaque fois, il existe un évènement sous-spécifié, impliquant le syntagme initial, qui est commencé ou apprécié. La théorie des qualia permet de résoudre en partie le problème de savoir à quels évènements renvoient ces coercions : en effet, cette théorie implique que le lexique comporte pour les entrées *book* et *coffee* des informations supplémentaires sur les activités communes qui

qui peuvent se réaliser comme des nominaux de leur propre chef, mais aussi comme des *attributs* d'autres entités.

19. À la lumière de ces exemples, il pourrait être légitime de se demander si les évènements eux-mêmes ne pourraient pas se rapprocher d'objets pointés de la forme $location \bullet time$, jettant ainsi le doute sur le bien fondé des phrases en (4) pour illustrer un genre différent de coercion. La notion d'évènement a en effet été traitée par de nombreux philosophes qui semblent s'accorder sur sa structure complexe et son aspect spatiotemporel [98, 48, 104, 16]. Curieusement, Pustejovsky ne semble pas — à ma connaissance — avoir abordé cette question dans ses travaux. Une manière de défendre ces exemples serait de remarquer que les évènements obtenus par formation des gérondifs des verbes s'accordent moins bien avec cette approche, notamment d'un point de vue spatial, comme dans « *?John left the reading* ». Si l'on accepte que ces gérondifs sont des évènements au même titre que *party*, alors l'absence de systématicité dans la sélection des aspects suggèrerait qu'on puisse écarter prudemment la modélisation des évènements comme des objets pointés dans le cadre de la présente discussion. Quoi qu'il en soit, d'autres exemples de ce type de coercion existent, par exemple ceux impliquant des verbes de perception comme *hear* (cf. [138, §3.5]).

les utilisent (quale télique), ainsi que sur les activités qui les créent le cas échéant (quale agentif). Pour *book*, par exemple, l'activité associée par défaut est la lecture : le premier énoncé ci-dessus peut donc, hors de tout contexte, se comprendre comme « Mary a commencé *à lire* le livre ». De même, le café est utilisé par défaut comme étant bu : le second énoncé peut donc se comprendre comme « Mary a apprécié *boire* son café ». La part contextuelle — et donc pragmatique — de ce type d'interprétation est cependant significativement plus grande, dans la mesure où de telles significations par défaut peuvent très vite être écrasées par de nouvelles informations. Considérons par exemple l'énoncé suivant :

(7) Mary is a writer. She began a new book yesterday.

Le fait que Mary soit écrivaine laisse entendre une interprétation différente de celle donnée par défaut pour l'énoncé en (6a), où l'activité télique de lecture est remplacée par le quale agentif : « Mary a commencé *à écrire* un nouveau livre ». Seule reste, néanmoins, l'idée générale d'une coercion de l'objet vers un évènement qui l'utilise, et c'est à cette partie que se limitera la question des types sémantiques ; la détermination de l'évènement précis impliqué est repoussé au niveau pragmatique de l'analyse linguistique, et sera donc exclue du présent travail.

Comme souligné précédemment, la sélection des aspects d'un mot polysémique est aussi une forme de coercion. Il en va de même pour le sous-typage, qui est par définition une transformation de types « standardisée » selon des critères de naturalité qu'il restera à préciser (cf. chapitre 2). On pourra d'ailleurs rajouter à ces exemples d'autres formes de changement de sens basés sur le contexte, comme les *transferts de signification* décrits par Nunberg [124] : il s'agit de situations où une entitée est désignée par un attribut qui lui est relié de façon purement pragmatique, comme par exemple dans l'énoncé ci-dessous, que l'on pourrait attribuer au serveur d'un restaurant qui, pour des raisons de praticité, désigne un client par le plat qu'il a commandé :

(8) The omelette is sitting at table 20.

Tous ces exemples font de la coercion un aspect central de l'intégration de la sémantique lexicale dans les systèmes formels. En effet, il a été dit plus haut que l'une des motivations majeures de cette intégration était le risque de surgénération du système de Montague ; or, une approche trop stricte de cette discipline de typage aurait vite fait de soulever le problème opposé de sous-génération. Pour trouver un juste milieu, il est nécessaire d'apporter plus de souplesse à la manipulation des types, et c'est précisément là que le phénomène de coercion intervient comme un moyen de contraindre la composition des mots tout en offrant la base théorique pour rétablir la capacité de production des langues naturelles. L'approche générale substituerait ainsi à la composition non-restreinte des grammaires de Montague une composition tout aussi permissive — car la pragmatique, et notamment le recours aux figures de styles, peuvent donner du sens à beaucoup d'énoncés qui paraîtraient sémantiquement inacceptables de prime abord — mais plus lourdement annotée, gardant trace de tous les changements de types qui ont été nécessaires à la construction de son interprétation. Cette annotation agit alors comme une source d'information utilisable ensuite par les niveaux suivants d'analyse sémantique pour tenter de résoudre complètement la signification de la phrase. Nous reviendrons en sous-section 3.3.3 sur cette idée d'annotation comme étape dans l'analyse sémantique.

Cette discussion du phénomène de coercion et des différentes formes qu'il peut prendre nous mène naturellement vers le second grand problème de l'intégration de la sémantique lexicale : comment rendre compte de manière uniforme et exhaustive des changements de types observés ? L'objectif serait ici de construire une théorie formelle de la coercion, idéalement adaptable aux différents modèles sémantiques proposés au fil du temps. En dépit de l'indéniable contribution

des travaux de Pustejovsky sur cette question, sa théorie GL n'en reste pas moins qu'un cas particulier de formalisme sémantique, utilisant des règles et des idées qui lui sont propres. [20] Les idées sous-jacentes ont cependant fait leur chemin jusqu'à être accomodées sous diverses formes dans d'autres théories. On se rappellera notamment de TCL et MTT qui, en plus des types pointés abordés précédemment, intègrent également des modèles généraux pour la coercion (cf. notamment [110] pour MTT) ; il en va de même pour le lexique génératif montagovien (*Montagovian Generative Lexicon*, abrégé en MGL), une intégration de GL dans un système formel très proche de celui de Montague et notamment developpé par Bassac, Mery et Retoré [15, 149] qui, s'il n'aborde pas directement les objets pointés et le sous-typage, repose aussi sur l'utilisation de coercions.

Quelles formes concrètes prennent ces coercions dans ces différents modèles ? Historiquement, les premiers cas de coercion étudiés ont été ceux liés à l'héritage, initié en théorie des langages de programmation [22]. Sur ce sujet, Luo [107] discute notamment de deux approches distinctes : le sous-typage subsomptif et le sous-typage coercitif. Le premier repose sur l'idée que le sous-typage peut être apparenté à l'inclusion d'ensembles, dans lequel le passage d'un élément d'un ensemble à un autre est naturel et ne requiert aucune action supplémentaire. C'est par exemple ce qu'on utilise en mathématiques : si l'on a une fonction $f : \mathbb{R} \to \mathbb{R}$ et un entier $n \in \mathbb{N}$, l'écriture de l'élément $f(n)$ est autorisée en vertu de l'inclusion $\mathbb{N} \subset \mathbb{R}$ sans autre forme de transformation. En théorie des types, le schéma général des règles de typage associées à cette approche est donné à gauche ci-dessous, et dit simplement qu'on peut changer le type d'un terme le long de la relation de sous-typage. Ceci s'oppose alors à l'approche coercitive, où la transformation de type invoque une fonction explicite : plutôt que de changer le type associé à un terme, on transforme le terme en même temps que le type par l'application d'une telle fonction ; et plus précisément, on invoque de telles fonctions au moment de la composition, menant ainsi au schéma général à droite.

$$\frac{u : a \qquad a \leq b}{u : b} \qquad\qquad \frac{f : b \to c \qquad u : a \qquad a \leq b}{f(\kappa(u)) : c} \quad (\kappa : a \to b)$$

Luo argue efficacement en faveur de cette seconde option pour modéliser le sous-typage et la généralise dans son modèle UTT [110]. Retoré fait également le choix de cette représentation explicite des transformations de types : en MGL, chaque entrée lexicale est associé à une liste de lambda-termes optionnels possibles qui jouent ce rôle de coercions explicites [149]. [21] On notera aussi que MGL fait usage de polymorphisme pour gérer certaines autres transformations. De son côté, Asher [4] fait le choix d'un type de base supplémentaire, appelé type présuppositionnel, qui contient les présuppositions de type des prédicats et sert de niveau intermédiaire d'analyse où les coercions sont appliquées. [22]

1.2.4 Des types encore trop abstraits

Ceci conduit donc à vouloir considérer de manière assez générale une théorie des coercions où ces dernières sont incarnées par des termes-fonctions qui sont insérées entre un prédicat et son argument durant la composition. Idéalement, une telle théorie devrait être capable de distinguer

20. En particulier, le présent texte questionnera au chapitre 3 la pertinence du choix de Pustejovsky (cf. notamment [138]) d'intégrer les qualia au système de types sous la forme d'un groupe indéfini de types distincts des types de bases, et ajoutés à ces derniers via de nouveaux constructeurs.

21. Cette approche permet à Retoré de se passer des types pointés, un choix qu'il défend par l'idée que la polysémie ne devrait pas être attachée à des types, mais directement aux entrées lexicales. Nous aurons l'occasion de revenir en section 6.3 sur cette nuance et les implications qu'elle peut avoir.

22. Cette approche, plus complexe que les autres, ne sera pas détaillée dans ces pages. Le lecteur désireux d'en savoir plus pourra se référer à [4].

les différentes formes de coercion, en ayant potentiellement pour chacune d'elles un mécanisme dédié, de sorte qu'on puisse relier a posteriori les coercions observées à ces mécanismes. On pourrait y voir un problème purement technique, celui de produire un ensemble de règles de composition adéquates pour remplacer l'application directe des lambda-termes dans le système de Montague ; mais la nécessité de précision d'un tel modèle, où la sémantique lexicale est partiellement ou totalement intégrée, au regard des subtilités linguistiques pose un troisième et dernier problème majeur : quels sont les types sémantiques concrets nécessaires et suffisants pour obtenir une telle précision ? Dans un sens, il a été mis en évidence plus haut que subdiviser le domaine des entités en une collection de sous-types était une étape obligatoire pour rendre compte de certains phénomènes sémantiques, et nous ne remettrons pas en cause cette hypothèse. Mais dans l'autre sens survient la question des limites à poser à la quantité de types à introduire : il s'agirait de répondre au principe d'économie sur le nombre de types utilisés et la finesse de la classification qu'ils sous-tendent, afin notamment d'éviter d'obtenir des représentations logiques surchargées de coercions qui n'auraient pas de réel intérêt sémantique, dans le sens où elles n'indiqueraient pas une transformation profonde de la signification initiale.

Les mêmes qui ont proposé des modèles suffisamment riches pour rendre compte de la coercion se sont également essayés à proposer des réponses à cette question complexe ; à plus forte raison que la problématique de représentation lexicale optimale, qui inclut donc cette question des types, s'est posée très tôt dans le développement des approches lexicales pour répondre aux critiques qui qualifiaient les premières représentations d'arbitraires [23]. Les premières esquisses d'une liste de types sémantiques élémentaires, organisées en treillis sur la base d'une relation d'*héritage*, se retrouvent dans les travaux de Briscoe, Boguraev, Copestake et Pustejovsky [23, 46, 140, 135]. Malheureusement, ces travaux ne proposent jamais de liste exhaustive des types utilisés, mais offrent néanmoins un aperçu via quelques fragments qui peuvent aider à se faire une idée des organisations possibles. La figure 1.1 en est un premier exemple, découpant a priori les entités en trois catégories naturelle, physique, et artificielle. Même en gardant à l'esprit que cette hiérarchie de types n'est pas complète (et donc que certains mots n'y ont a priori pas de place), ce découpage soulève plusieurs questions, comme le genre de mots qui pourraient avoir à utiliser les types *natural* et *artifact* : en quoi consistent-ils et quels propriétés capturent-ils ? De même, quelle preuve avons-nous que le type *comestible* est réellement essentiel dans un système de type ? On peut penser à des verbes comme *manger* qui sélectionneraient ce type d'argument, mais l'existence de la géophagie, illustrée par l'énoncé (9) ci-dessous, remet en cause cette idée :

(9) Le guérisseur mange de la terre.

Il y a peu de chance qu'on associe volontairement le type *comestible* au mot *terre*, dans la mesure où ce mot renvoie à quelque chose d'intuitivement très différent des significations de *pomme* ou de *lasagnes*, par exemple. Pourtant, la signification de « manger de la terre » dans l'énoncé (9) est littérale : s'il y a coercion, de quel forme de coercion s'agit-il ? Son aspect d'aliment n'est pas un attribut de *terre*, et n'est pas non plus une instance de sous-typage. Au contraire même, cette transformation change le type initial [23] de *terre* pour l'un de ces sous-types : on pourrait voir cela comme une forme de coercion par introduction, puisqu'on ajoute des contraintes nouvelles au type initial. Et ce raisonnement n'est pas un cas isolé, puisque d'autres instances substituant

23. Le type de *terre* n'est pas précisé ici car la hiérarchie de la figure 1.1 ne donne qu'une intuition relative sur où le placer. La présente discussion s'appuie sur l'idée que le type le plus raisonnable a priori pourrait être *natural_physical*, mais cette affirmation peut être sujette à débat puisque la signification de *natural* n'est pas explicitée.

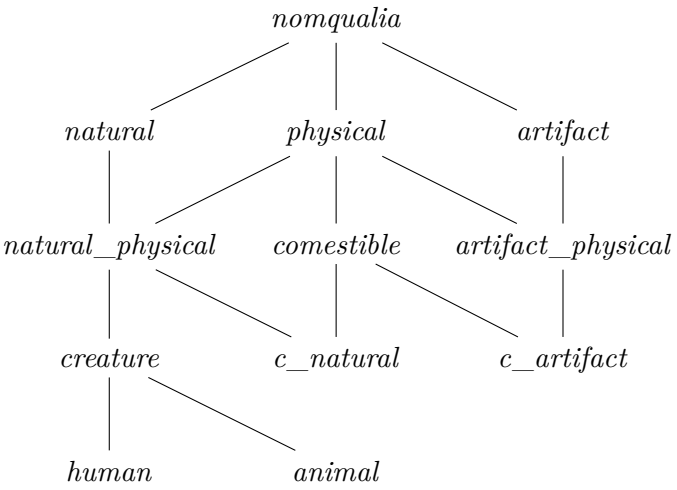


FIGURE 1.1 – Fragment de hiérarchie de types issue de [183]

terre pour des mots du même type sont tous aussi valides et littéraux : « manger des cailloux » se dit des poules, « manger du bois » se dit des termites, etc. À la lumière de ces exemples, on peut alors douter de l'utilité du type *comestible* pour l'interprétation du verbe *manger* : autant utiliser un type plus général, qui couvre les termes *terre*, *bois*, *caillou*, par simple principe d'économie. Il se pourrait que d'autres mots aient effectivement besoin de ce type, mais cette discussion permet au moins d'illustrer dans quelles mesures de telles propositions peuvent être remises en cause.

La figure 1.2 compile plusieurs fragments de la hiérarchie de types imaginée par Pustejovsky. Il est intéressant de relever que cette hiérarchie semble distinguer trois catégories principales de natures différentes de celles issues de l'exemple précédent : les entités, les propositions, et les évènements, une approche qui tranche aussi avec l'idée d'une simple subdivision des entités esquissée précédemment. La nature de ces types *proposition* et *event* n'est pas détaillée dans [135] ; le type *event* comme distinct des entités peut renvoyer sans équivoque à la sémantique néo-davidsonienne, mais le rôle de *proposition* et son lien potentiel avec le type t n'est pas explicité. Par ailleurs, les artéfacts comme introduits dans l'exemple précédent se retrouvent dans cette théorie sous une autre forme : ils sont construits comme des complexes à l'aide des types de qualia et des constructeurs de types associés. On notera en revanche la présence des types pointés ici : le type *print_matter* peut exactement se définir comme $phys\_obj \bullet information$, sous l'hypothèse de considérer la bonne relation entre le support physique et l'information. Un mot comme *book*, par exemple, pourra alors se voir assigner ce type. De même, *newspaper*, entendu cette fois dans sa triple acception d'organisation, d'objet physique et de contenu informationnel, se verra assigner le type $organisation \bullet print\_matter$.

Pustejovsky étoffe par la suite son approche des types sémantiques dans [136]. Le système qu'il y décrit est à nouveau séparé en trois domaines comme en figure 1.2, mais le type des propositions disparaît au profit de ce qu'il nomme qualité, un domaine qui se rapporte aux adjectifs ; alors que le domaine des entités se rapporte aux noms et que le domaine des évènements se rapporte plutôt aux verbes. Puis, chaque domaine se découpe en trois rangs : les types naturels, qui recouvrent les types élémentaires tels qu'on a pu initialement les concevoir dans notre discussion ; les types fonctionnels, qui sont des types naturels enrichis de qualia (et recouvrent donc la notion d'artéfact) ; et les types complexes, c.-à-d. les types pointés. Cette approche lui permet au passage de reclasser les coercions en fonction de s'ils préservent ou non le domaine

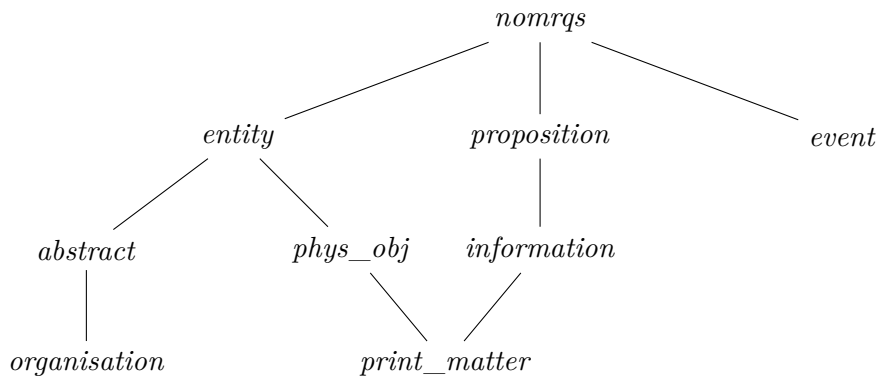


FIGURE 1.2 – Fragment de hiérarchie de types issue de [135, §6.2]

d'une part, et le rang d'autre part. Le cœur de ce travail est néanmoins plus méthodologique que pratique, et si nous aurons l'occasion de revenir sur ce premier aspect (cf. chapitre 3), les exemples de types qui y sont présentés n'apportent pas de grande nouveauté au regard des types que l'on a déjà vu.

Il semble en effet que la répartition générale des types suive une poignée d'heuristiques universellement reconnues, comme la distinction entre entités physiques et abstraites qui ressort de la figure précédente. On retiendra aussi une autre distinction fréquemment rencontrée entre entités animées et inanimées. Asher [4, §2.2] présente plusieurs autres cas similaires dans les langues naturelles : on y retrouve la distinction objet–évènement également vue plus haut, ainsi que d'autres plus rares et parfois plus subtiles comme objet–lieu, masse–dénombrable, sorte–individu, ou encore conteneur–contenu, sans oublier les pluriels collectifs et distributifs ; toutes ces distinctions pouvant servir de charpente autour de laquelle construire un système de types sémantiques. Malheureusement, Asher ne s'aventure pas lui non plus à donner un aperçu des types qu'il préconise concrètement. Retoré s'est également intéressé à cette problématique, et a judicieusement listé dans [149] les diverses réponses apportées. Pour lui, le point de vue d'Asher — et probablement par extension de Pustejovsky et ses collègues — consiste donc à postuler une collection de types restreinte et guidée notamment par les heuristiques précedemment décrites. Élaborant sur cette idée, Mery et Retoré [117] suggèrent de s'inspirer des classificateurs, dont l'usage est très répandu dans les langues asiatiques, pour construire un système de types correspondant, soulignant qu'une telle approche serait naturelle d'un point de vue linguistique et d'un point de vue cognitif. Par ailleurs, les systèmes de classificateurs recouvrent souvent une grande partie des distinctions évoquées au début de ce paragraphe, et sont parfois même plus précis, formant une hiérarchie qui leur est propre et qu'il serait facile d'importer dans un système de types. Les auteurs sont néanmoins conscients que cette proposition a elle-même ses limites, et qu'elle n'apporte pas de réponse définitive à la question des types.

Une dernière approche des types sémantiques, également relevée par Retoré, est celle défendue par Luo et Chatzikyriakidis [109, 34, 35] pour UTT : leur proposition consiste à associer chaque nom commun à son propre type. La raison de ce choix réside dans les spécificités du modèle UTT, qui s'écarte largement de la sémantique montagovienne en remplaçant l'utilisation généralisée de prédicats par des interprétations davantage portées par la théorie des types elle-même. Ainsi, plutôt que d'interpréter une phrase comme « *John is a man* » par une prédication $\mathbf{man}(\mathbf{j}) : t$, UTT recourra plutôt à un jugement de typage $\mathbf{j} : \mathit{man}$; de telle sorte que les prédicats interprétant les noms sont tout bonnement remplacés par ces nouveaux types (évitant de fait la redondance prédicat-type soulignée au début de la sous-section 1.2.2), et que l'interprétation

associée n'est plus une proposition mais une preuve de typage correct. Ce parti pris, qui ancre la sémantique formelle dans une théorie des types plus complexe — avec entre autres l'accès aux types dépendants —, remplit donc le système avec un très grand nombre de types élémentaires. [24] On pourrait confronter ce nombre de types aux nombre d'entrées nécessaires dans un lexique : puisqu'il y a de toute façon des informations lexicales dont on ne peut se passer, il ne paraît pas si dommageable d'introduire tous les noms communs comme des types. Le seul bémol de cette approche est d'ignorer le principe d'économie : en effet, les mots qui demeurent interprétés comme des prédicats, à l'exemple des adjectifs et des verbes, gardent une capacité combinatoire qui englobent de nombreux noms différents, et donc autant de sous-types du type assigné à leur argument. Même en reconnaissant des prédicats à l'utilisation très spécifique, avec donc un type plus restreint (cf. [4, §2.2.7]), l'ensemble des types utilisés pour les arguments des prédicats n'est qu'un petit sous-ensemble des types initialement construits sur les noms communs. Et quand bien même UTT introduit un peu de sucre syntaxique pour effacer les coercions de sous-typage [35, §2.4], il reste théoriquement un grand nombre de coercions implicites qui ne semblent pas strictement nécessaires pour une interprétation sémantique précise.

Il apparaît ainsi qu'un certain flou perdure sur la question de la population concrète des hiérarchies de types sémantiques élémentaires. Comme nous avons pu le voir, certaines heuristiques distinguant des catégories particulières semblent faire consensus, mais celles-ci ne suffisent pas — ou n'ont pas été assez développées — pour obtenir une collection de types satisfaisante. Par souci d'exhaustivité, on pourra également mentionner le projet WordNet [118, 58], dont les *synsets* maximaux sont souvent comparés à un exemple de système de types possibles. Cette approche, cependant, est purement lexicographique : on ne peut donc pas s'attendre à ce que ces synsets reflètent exactement la capacité combinatoire de la langue. Les arguments des prédicats, par exemple, ne sont pas indiqués, et Pustejovsky [136] remarque que l'ambiguïté contrastive et la polysémie y sont gérées de la même manière, là où un système de types sémantiques les distinguerait. [25] En conséquence, il ne semble pas à l'heure actuelle exister de solution définitive à ce troisième problème. Bon nombre des auteurs cités dans la discussion précédente ne donnent d'ailleurs pas de listes de types qu'ils jugent pertinents, mais proposent des méthodologies qui permettraient de les déterminer. Néanmoins, les procédures et suggestions décrites ne sont souvent pas applicables à grande échelle, et requièrent donc un investissement trop important pour pouvoir mener à un résultat exploitable.

Pour résumer. Au fil de cette section, nous avons mis en lumière les limites des grammaires de Montague, et pourquoi l'intégration d'informations issues de la sémantique lexicale semble y apporter une réponse pertinente. Cette intégration s'opère à deux niveaux : d'une part, il s'agit d'enrichir l'homomorphisme de traduction d'une base de données, le lexique, associant à chaque mot les informations sémantiques qui lui sont propres, et d'autre part, il s'agit d'étoffer le système de types sémantiques en subdivisant le type e des entités en une hiérarchie de types variés, encodant en leur sein des contraintes de sens et de compositionnalité. Ce développement

24. À titre d'exemple, la page de statistiques du projet de dictionnaire en ligne collaboratif *Wiktionnaire* (`fr.wiktionary.org/wiki/Wiktionnaire:Statistiques`) recense à la date du 20 janvier 2022 pas moins de 188 020 noms communs sur 397 678 lemmes parmi ses entrées pour la langue française, ce qui ferait donc théoriquement tout autant de types sémantiques élémentaires dans le formalisme UTT associé, là où les propositions précédentes n'en compteraient a priori que quelques dizaines.

25. Il convient de noter que WordNet ne prétend en aucune manière proposer une modélisation des entrées lexicales en termes formels. Aussi, si les données que le projet collecte peuvent être utiles sur bien des tâches (WordNet est notamment très utilisé pour des applications en annotation, p. ex. pour la désambiguïsation des sens [105, 182]), il n'est pas étonnant que son intérêt pour les systèmes formels ne soit que modéré ; même s'il n'est pas exclu qu'une partie de ces données puisse aider à la construction d'un lexique formel.

du paradigme montagovien n'est cependant pas sans apporter de nouvelles difficultés à surmonter pour les sémanticiens. Dans notre discussion, nous avons dégagé trois problèmes majeurs qu'une telle modification implique :

(i) une entrée lexicale dispose d'une grande quantité d'information sémantique associée, reflétant la grande capacité de production de la langue, mais il est donc nécessaire de déterminer quelle partie de l'information doit être enregistrée dans le lexique, et quelle autre partie doit être encodée dans les types sémantiques ;

(ii) en supposant entendue une certaine compréhension des types, la langue démontre la capacité à pouvoir appliquer des coercions, c.-à-d. utiliser de manière compréhensible des mots d'un certain type dans des contextes où un autre type est attendu, résultant en général en la modification de la signification des mots initiaux vers des significations logiquement voire pragmatiquement proches : il faut donc trouver le moyen d'incorporer ces coercions dans le cadre compositionnel considéré ;

(iii) afin d'équilibrer le système entre sur- et sous-génération, il est important d'utiliser une hiérarchie de types qui reflète au mieux le comportement de la langue, ce qui implique de déterminer comment obtenir la collection de types la plus pertinente pour ce faire, et d'en déduire quels sont les types sémantiques à utiliser en pratique.

Il est probable que la réponse au problème (i) fournisse des éléments pour aider à résoudre le problème (iii), ce qui le cas échéant nous ramènerait à deux question primordiales : comment construire les types sémantiques en eux-mêmes, et comment construire la théorie des types qui les utilise pour rendre compte de la compositionnalité. Or, les propositions apportées au fil du temps offrent déjà plusieurs formes de réponses à la seconde question, mais laissent la première dans des eaux troubles : sans solution définitive et complète, mais avec quelques indications parcimonieuses qui permettraient d'y répondre. Il y a donc ici la place pour une revue plus importante de cette première question, et potentiellement pour des comparaisons, voire des généralisations, entre les théories coercitives actuelles.

1.3 Des principes-clé pour un système plus efficace

Nous savons désormais que pour construire des interprétations sémantiques précises et fiables, tenant compte de toutes les subtilités des langues naturelles, il est nécessaire d'enrichir les systèmes syntaxico-sémantiques tel celui de Montague par des informations lexicales, intégrées d'une part dans un lexique, et d'autre part dans une théorie de types plus complexe et plus expressive. L'introduction des coercions est un aspect important de cette nouvelle approche : celles-ci cristallisent en effet toute l'étendue de la variabilité sémantique inhérente à la langue, mais requièrent une discipline de compositionnalité moins stricte et moins directe que celle observable chez Montague, tout en augmentant significativement la taille des lambda-termes apparaissant dans les interprétations finales. Cet aspect est d'autant plus important qu'il se heurte à ce que nous avons appelé en section 1.1 le paradoxe de la complexité, c.-à-d. le fait qu'à chaque mot était associé une constante de prédicat simple mais que les exigences de l'homomorphisme de traduction forçaient à plonger ces constantes dans des lambda-termes encore plus complexes : l'association de cette complexité à l'introduction des coercions peut-elle aboutir à un système facilement utilisable ?

Le problème est d'autant plus subtil que l'interaction entre les types prédicats dans une théorie inspirée des types simples d'une part, et la relation de sous-typage intégrée aux hiérarchies de types issues de la sémantique lexicale d'autre part, ouvre la porte à des difficultés

supplémentaires. Dans la théorie des langages de programmation, il a rapidement été reconnu que la relation de sous-typage, une fois établie pour les types élémentaires, pouvait se propager aux types de fonctions à condition de respecter la règle de *covariance* pour le type de l'image, mais de *contravariance* pour le type de l'argument [1]. Autrement dit, si a, b et c sont trois types tels que $a \leq b$, on a par covariance à droite la relation $c \to a \leq c \to b$, mais du côté opposé, la contravariance à gauche donne $b \to c \leq a \to c$: la relation de sous-typage est inversée lors du passage à la position argument d'une fonction. Cette propriété, qui peut paraître contre-intuitive de prime abord, est nécessaire pour assurer la *sûreté* du système, c.-à-d. la garantie que lorsqu'un terme est bien typable, l'exécution du programme ne conduira pas à un calcul incohérent qui risquerait de produire une erreur. De ce point de vue, être un terme de type a sous-type de b signifie qu'on peut utiliser ce terme à n'importe quel endroit où un terme de type b est attendu sans risque d'incohérence. Si l'on propage ce raisonnement aux fonctions, supposons que l'on ait un emplacement de programme où un type $a \to c$ est attendu : tout argument passé à la fonction devant occuper cet emplacement étant alors au plus du type a, il reste sûr de placer à cet endroit un terme de type $b \to c$ puisqu'il acceptera sans problème l'argument. Par conséquent, utiliser un type $b \to c$ là où un type $a \to c$ fonctionne, et on retrouve la propriété précédente caractérisant le sous-typage, d'où la contravariance.

Cette garantie de stabilité offerte par la contravariance du sous-typage est une propriété si importante que cette règle s'est rapidement imposée comme standard dans toutes les théories qui utilisent du sous-typage, et les théories de types pour la sémantique formelle n'y ont pas échappé. Malheureusement, comme relevé par Luo [107], la complexité de typage apportée par le système de Montague s'accorde mal avec cette règle. En effet, si l'on se rappelle ici de la table 1.1, on remarque par exemple que les noms sont interprétés comme des prédicats du premier ordre, de la forme $e \to t$, et tandis que les adjectifs sont des modificateurs de tels prédicats, soit $(e \to t) \to e \to t$. Que se passe-t-il alors si l'on remplace les types e par des sous-types élémentaires ? On pourrait supposer avoir un adjectif qui s'applique aux objets physiques, par exemple *heavy*[26], porté par une constante $\mathbf{heavy} : p \to t$, et la constante $\mathbf{man} : h \to t$ bien connue pour *man*, avec en hypothèse de fond $h \leq p$. Dans un système montagovien, l'adjectif sera nécessairement plongé dans un terme de type $(p \to t) \to p \to t$ et, pour la dérivation d'une expression comme « *heavy man* », l'homomorphisme de traduction doit appliquer ce terme à la constante **man**. Seulement, une telle application ne peut aboutir que si l'on a une coercion de sous-typage qui satisfait $h \to t \leq p \to t$, soit l'inverse de ce qui est possible par la règle de contravariance !

Il y a donc bien une incompatibilité entre le sous-typage et les grammaires de Montague qu'il nous faut résoudre. La solution apportée par Luo est d'écarter la discipline montagovienne de l'interprétation par prédicats : dans UTT, il choisit comme on l'a vu d'interpréter les noms communs comme des types ; et les modificateurs de ces prédicats, comme les adjectifs, sont traités comme des prédicats du premier ordre encapsulés dans des types dépendants. Ainsi, « *heavy man* » ne sera pas un terme de la forme $\mathbf{heavy}(\mathbf{man}) : t$ comme habituellement, mais plutôt un type $\Sigma x : h.\mathbf{heavy}(c(x))$ (impliquant la coercion $c : h \to p$). Le cas similaire des adverbes est traité différemment, rompant encore davantage avec la tradition montagovienne où adverbes et adjectifs ont le même type : cette fois, l'interprétation dans UTT implique du polymorphisme (cf. [35, §3.1]). Cette solution, bien que développant un formalisme sémantique très puissant et capable de rendre compte de nombreux détails, a cependant l'inconvénient de complexifier la traduction à l'interface syntaxico-sémantique : en effet, avec certaines catégories syntaxiques

26. Cet exemple étant uniquement choisi pour des fins d'exposition conceptuelle, on ignorera ici les considérations de scalarité inhérents à un tel adjectif (cf. note 6 p. 6).

se traduisant comme des types, d'autres comme des termes, et sachant qu'en plus, à l'instar des adjectifs, certains types se construisent en utilisant des termes, l'existence d'une application capable de générer automatiquement l'interprétation sémantique d'une expression donnée est moins évidente. Chatzikyriakidis et Luo [35, §7.3] esquissent une réponse à cette remarque en proposant une grammaire catégorielle avec types dépendants. Cette dernière solution, bien que prometteuse, ne capture cependant pas la simplicité conceptuelle du morphisme de traduction inhérent à l'approche montagovienne, et semble par là même s'écarter légèrement de la question de la compositionnalité.

Naturellement, le choix de s'écarter du paradigme instauré par Montague ou non est propre à chacun, et il n'est a priori pas de formalisme supérieur aux autres. UTT compte nombre de caractéristiques très intéressantes pour la modélisation sémantique, qui viennent au prix de davantage de complexité pour la traduction syntaxe-sémantique, tandis que le système montagovien, comme nous venons de le voir, se revèle conceptuellement simple mais implique d'une part le paradoxe de la complexité précédemment évoqué, et d'autre part une mauvaise interaction avec la discipline traditionnelle du sous-typage. Dans la mesure où notre souci premier serait de conserver la simplicité du paradigme de traduction uniforme d'un langage source, la syntaxe, vers un langage cible, la sémantique, afin de refléter au mieux le fonctionnement des langues naturelles, il serait précipité dans la présente discussion d'écarter immédiatement l'approche montagovienne. Cela conduit évidemment à interroger l'autre partie du problème : peut-on faire fonctionner le sous-typage dans un tel système ? Que faut-il modifier pour y parvenir sans toucher au paradigme de traduction ? Les extensions de cet héritage de Montague, quelles qu'elles soient, semblent en effet converger vers une seule réponse globale, que nous appellerons dans ces pages la *flexibilité*.

Cette idée s'oppose à la rigidité propre à la grammaire de Montague telle que discutée dans la section 1.1, où deux éléments de types différents ne peuvent pas se composer. Désormais, il y a deux types de discordances : une discordance structurelle, où le type de l'argument pêche par son ordre et son arité, et une discordance lexicale, où le type de l'argument contient un type élémentaire différent de celui attendu au même emplacement. La première, comme expliqué au début de la section 1.2, est d'importance moindre puisqu'une grande partie des conditions dérive des catégories syntaxiques initiales, mais divers exemples linguistiques ont poussé à des révisions sur la manipulation des types sans intégration de la sémantique lexicale. Un exemple est la discussion de Chierchia [36] sur la distinction entre fonction (type prédicat) et propriété (type élémentaire ?) comme dans l'énoncé suivant :

(10) Being wise is crazy.

Ici, une nominalisation par le biais d'un gérondif permet de convertir un prédicat du premier ordre, *be wise*, en un argument acceptable pour un autre prédicat du premier ordre *be crazy*, suggérant par conséquent une transformation du lambda-terme associé à *be wise* — si tant est qu'on ne traite pas le gérondif comme une entrée lexicale séparée. Si l'on peut naturellement entrevoir une influence de la syntaxe sur la manière dont cette distinction peut être perçue, l'exemple suffit néanmoins à montrer en quoi des questions d'ordre structurel sur les types sémantiques peuvent être mises en cause.

Un autre cas, plus célèbre et déjà évoqué en section 1.1, est celui des changements de types. Partee et Rooth [131] observent que pour optimiser certaines lourdeurs du système de Montague, [27] il vaut mieux se débarrasser de la contrainte d'homomorphisme pour réduire le type des

27. Le cas traité dans [131] est celui des conjonctions : on se souvient en effet que Montague postule trois entrées différentes, une pour chaque catégorie syntaxique appropriée, alors qu'elles ont la même forme d'interprétation

interprétations, privilégiant notamment pour *NP* le type e plutôt que le type élevé $(e \to t) \to t$, et fournir a posteriori des règles permettant de mobiliser à nouveau les termes sémantiques complexes lorsque cela est nécessaire. Cette idée d'autant plus pertinente que, comme évoqué plus haut, le passage des types simples aux types complexes suit un schéma général uniforme : à titre d'exemple, les constantes de types e qui interprètent notamment les noms propres peuvent être converties en leur terme élevé par la simple application sur celles-là du terme général $\lambda x \lambda u.u\, x : e \to (e \to t) \to t$; tandis que les constantes pour les verbes transitifs, de type $e \to e \to t$, peuvent être converties en leur type élevé via le lambda-terme :

$$\lambda \mathfrak{V} \lambda P \lambda x.\, P(\lambda y.\mathfrak{V}(y)(x)) : (e \to e \to t) \to ((e \to t) \to t) \to e \to t$$

L'exploration de ces remarques est developpée dans [129], où Partee propose d'assigner aux catégories syntaxiques non pas un seul type sémantique, mais un ensemble de plusieurs types possibles, en particulier pour la catégorie *NP* qui se verrait assigner les types e (référentiel), $e \to t$ (prédicatif), et $(e \to t) \to t$ (quantificationnel). Cette proposition est soutenue par différentes observations linguistiques sur le comportement des syntagmes nominaux : la distinction entre les lectures réferentielles et quantificationnelles est par exemple liée à des questions de pluralité, de telle sorte que des syntagmes comme *John*, *a man*, *the man* autorisent la première, alors que *every man* ne supporte que la seconde ; et la lecture prédicative est quant à elle observable pour certains verbes qui semblent utiliser des arguments adjectivaux (réduits au type minimal $e \to t$) à l'image de *consider*. Les énoncés en (11) utilisent la même structure dans les deux cas, mais le second voit le syntagme adjectival (*AP* ci-dessous) remplacé par un syntagme nominal, qui devrait donc recevoir le même type sémantique :

(11) a. Mary considers [$_{\text{NP}}$ John] [$_{\text{AP}}$ competent in semantics].
b. Mary considers [$_{\text{NP}}$ that place] [$_{\text{NP}}$ an island].

Intuitivement, l'effet du changement de type en (11b) est de convertir le référent *an island* en un prédicat *is an island* qui s'appliquerait à *that place*, ce qui conduirait d'un moins de vue sémantique à mobiliser le prédicat **island** : $e \to t$ issu de l'interprétation du nom commun au cœur du syntagme.

Pour gérer cette variété de types, Partee propose comme évoqué précédemment la stratégie de d'abord interpréter les entrées lexicales par leur type le plus petit en terme d'ordre, puis d'augmenter l'ordre si le contexte l'exige. Mais les outils théoriques disponibles sont en réalité plus larges : les changements de types sont incarnés par des applications de conversion d'un type à l'autre. Ces applications incluent les lambda-termes $\lambda x \lambda u.ux : e \to ((e \to t) \to t)$ évoqué plus haut, et $\lambda x \lambda y.(y = x) : e \to (e \to t)$ pour la conversion prédicative, ainsi que les constructions proposées par Chierchia [36] et d'autres opérateurs comme iota [28]. De manière générale, de nombreuses opérations sont disponibles pour les changements de types, bien que certaines — comme iota — ne soient que des opérations partielles. Partee remarque tout de même que les applications qui augmentent l'ordre de leurs arguments sont toujours totales et injectives. Il est alors frappant de constater que la théorie complète de ces applications, sur laquelle nous reviendrons en section 6.5, a de nombreux points communs avec les coercions lexicales introduites en section 1.2 : il s'agit en effet dans les deux cas d'autoriser des transformations au niveau des types sémantiques pour adapter en aval la signification des mots au contexte de la phrase.

sémantique, reposant sur les connecteurs logiques correspondants.

28. Ce subnecteur, introduit par Russell [154], prend en argument une propriété du premier ordre qui n'est satisfaite que par une seule entité, et renvoie l'entité qui la satisfait. Il agit en particulier comme l'inverse du lambda-terme de conversion prédicative ci-dessus : si *is John* est rendu comme le lambda-terme $\mathbf{is\_john} := \lambda x.(x = \mathbf{j}) : e \to t$, alors $\iota x.\mathbf{is\_john}(x) = \mathbf{j} : e$.

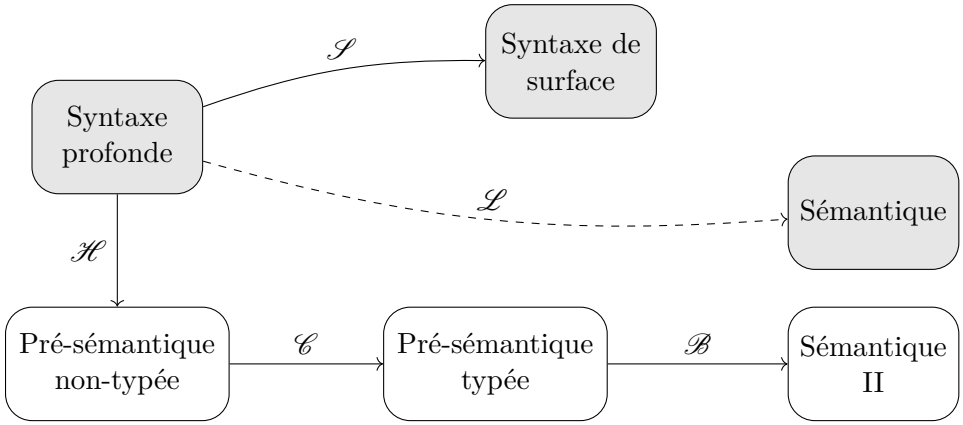


FIGURE 1.3 – Schéma du système d'interface syntaxe-sémantique hypothétique

Cette proposition de changements de types rend donc à la discordance structurelle une plus grande importance, en la remettant à un niveau similaire à celui de la discordance lexicale. Il y a ainsi une base commune à ces deux aspects de la compositionnalité qui appelle, comme on l'a évoqué, à davantage de flexibilité. Il semble par ailleurs difficile d'imaginer pouvoir résoudre les problèmes posés par l'un de ces deux aspects sans toucher à l'autre : à la lumière de la discussion ci-dessus, on peut aisément deviner pourquoi la stratégie de Partee peut également bénéficier aux coercions lexicales, puisque composer avec des types sémantiques plus simple devrait permettre d'éviter par exemple le problème du sous-typage pour les adjectifs observé par Luo. Le nouveau défi à relever serait alors de proposer un système qui introduise de la souplesse pour opérer des changements de types de toutes sortes tout en restant suffisamment cadré pour éviter la surgénération d'interprétations indésirables ou à contresens. Et comme ces transformations au niveau des types sémantiques conduisent à des modifications dans la signification des entrées, il nous paraît essentiel de conserver une trace des différentes opérations appliquées lors de la composition, afin que cette information puisse être prise en compte dans les étapes suivantes de l'analyse sémantique ou pragmatique.

Il va sans dire que cet objectif amène à une reconceptualisation globale de la modélisation à l'interface syntaxe-sémantique, car quelques hypothèses sous-jacentes dans le système de Montague sont remises en cause, en particulier l'uniformité de la traduction entre syntaxe et sémantique : si un tel morphisme de traduction existe, du fait des stratégies de changements de types et de coercions lexicales il ne pourra plus être traité comme un homomorphisme, mais comme la composition de plusieurs opérations non-homomorphiques. Nous schématisons en figure 1.3 une esquisse de la forme que pourrait prendre un formalisme syntaxico-sémantique réévalué au regard de nos objectifs. Ce schéma n'est en rien définitif à ce stade du présent exposé, mais permet de clarifier les grandes lignes des modifications à apporter au système précédent pour y implémenter les composantes nécessaires à une plus grande flexibilité. Par ailleurs, il permet de souligner l'aspect modulaire de l'interprétation sémantique : nous nous fondons sur l'idée que chaque étape de l'analyse est exprimable sous la forme d'une algèbre bien choisie, et que les différentes opérations qui s'y appliquent, si elles ne sont pas des homomorphismes *stricto sensu*, traitent l'information de façon à préserver les isomorphismes de structure.

La figure 1.3 représente en grisé les trois modules initiaux du système de Montague [120] : la syntaxe profonde est le langage où les mots sont contrôlés par leur catégorie sémantique et combinés à l'aide d'opérateurs abstraits ; la syntaxe de surface est la phrase finale, obtenue par le morphisme $\mathscr{S}$ qui « efface » les opérateurs abstraits pour ne conserver que les mots (effaçant

au passage toute ambiguïté syntaxique) ; et la sémantique est le lambda-calcul logique final, obtenu initialement par traduction avec l'homomorphisme $\mathscr{L}$ évoqué en section 1.1 puis par β-réduction du terme ainsi obtenu. Nos objectifs impliquent alors d'abandonner l'homomorphisme en question (d'où son tracé en pointillés) au profit d'une décomposition en plusieurs modules intermédiaires. La *pré-sémantique non-typée* est obtenue par un homomorphisme $\mathscr{H}$ qui préserve la structure de la syntaxe profonde et remplace chaque mot par le lambda-terme représentant son interprétation sémantique minimale, sans tenir compte des discordances de type éventuelles. Il s'agit en cela d'un morphisme moins contraint que l'homomorphisme de Montague. Puis, l'opération de *complétion* $\mathscr{C}$ insère dans la structure pré-sémantique toutes les coercions et changements de types nécessaires pour former un terme qui soit bien typé, formant alors le langage cible de *pré-sémantique typée*. Enfin, l'opération $\mathscr{B}$ applique toutes les β-réductions possibles pour obtenir le lambda-terme final dans un langage sémantique légèrement différent de celui de Montague, car enrichi notamment de coercions élémentaires irréductibles, exploitables ensuite par d'autres opérations pour une analyse sémantique plus poussée.

Ce découpage modulaire de l'interprétation sémantique permet ainsi de commencer par des constructions plus simples, qui par la suite peuvent être converties en des structures plus complexes si nécessaire. On retrouve cette idée chez de Groote [50], qui remarque que les lambda-termes de Montague correspondent à la réécriture de leur interprétation minimale dans le style de la programmation par continuation, ouvrant ainsi la voie à de nouvelles possibilités d'interprétation. De même, la prodécure d'intensionnalisation décrite par de Groote et Kanazawa [51] est une transformation modulaire qui permet de passer d'un langage sémantique simplement typé à sa version enrichie du type des mondes possibles, utilisable dans la logique intensionnelle originale de Montague [120]. Nous retiendrons de ces exemples le principe suivant : l'interprétation sémantique que nous cherchons à produire doit être la plus simple possible, dans le sens où elle doit non seulement être économe sur les types élémentaires utilisés, comme vu en section 1.2, mais aussi sur l'ordre de types sémantiques utilisés qui doivent être maintenus à leur niveau minimal autant que possible.

Pour résumer, la discussion menée dans la présente section, mettant en évidence les problèmes inhérents à l'interaction entre le système montagovien traditionnel et l'intégration du sous-typage d'une part, et la nécessité d'assouplir les contraintes pesants sur l'interprétation des syntagmes nominaux d'autre part, nous enseigne qu'il nous faut un modèle d'interface syntaxe-sémantique repensé en profondeur, que nous pouvons caractériser par les trois principes-clé suivants :

(i) *simplicité* : le noyau dur de la représentation sémantique doit pouvoir être exprimable de façon minimaliste, tant du point de vue des types élémentaires utilisés que de l'ordre des types prédicats impliqués ;

(ii) *flexibilité* : la représentation sémantique doit pouvoir adapter son typage à des phénomènes linguistiques imprévisibles par l'insertion de transformations de types adéquates ;

(iii) *modularité* : la représentation sémantique doit pouvoir être enrichie de dimensions supplémentaires par la simple application de morphismes bien construits.

Le système final devra donc tenir compte de ces caractéristiques, et s'approcher autant que possible de l'organisation schématique donnée en figure 1.3. Il est donc probable que ce nouveau système ait à s'éloigner de la conception traditionnelle du modèle montagovien, tout en conservant certaines de ses idées initiales. De plus, quand bien même l'objectif principal de la présente de thèse ne sera de proposer qu'une théorie sémantique qui tienne compte de ces principes, il nous faudra veiller à ce que ladite théorie puisse s'inscrire parfaitement dans un tel schéma d'interface syntaxe-sémantique.

1.4 Problématisation de la question des types

Il est maintenant temps pour nous d'introduire plus formellement le cadre théorique et les objectifs que nous nous fixons dans la présente thèse. Comme les sections précédentes ont pu le dépeindre, le stade actuel de la recherche sur l'intégration de la sémantique lexicale dans des systèmes formels n'a pas encore résolu toutes les difficultés que cette approche soulève. Les propositions les plus abouties à ce jour sont certainement le modèle UTT de Luo et Chatzikyriakidis [35] et le modèle GL de Pustejovsky et Batiukova [139], offrant toutes deux une vision différente de la sémantique et de son rapport au lexique et à la syntaxe. L'une des principales critiques que l'on pourrait poser à ces formalismes est le manque d'adaptation à un système d'interface syntaxe-sémantique, ce qui rend leur utilisation pour le traitement automatique des langues plus difficile à concevoir. Par ailleurs, nous avons vu que les questions de ce que représentent les types sémantiques, et de quelle collection de types serait optimale pour la modélisation de la langue, ont rarement été approfondies au-delà d'un cadre théorique formel.

Cadre théorique. Comme esquissé en section 1.3, nous nous basons dans le présent exposé sur l'idée que chaque niveau d'analyse linguistique (à partir de la syntaxe) est représentable sous forme d'une algèbre, et que la traduction entre syntaxe et sémantique est une série d'applications qui utilisent les informations algébriques du langage source pour construire les images dans le langage cible. Nous nous rapprochons en cela de la vision proposée par la théorie des grammaires catégorielles abstraites [49], à ceci près que nous écarterons certainement la contrainte d'homomorphisme sur le typage entre les deux langages afin de laisser le champ libre à la procédure de complétion par coercions et changements de types. La plupart des langages utilisés seront assimilés à des lambda-calculs typés dont la théorie des types sous-jacente pourra varier d'un langage à l'autre. Certains langages seront probablement moins expressifs (p. ex. dépourvus de lambda-abstractions), mais pourront toujours être étendus en des lambda-calculs complets.

À l'instar des formalismes présentés dans ce chapitre, nous supposerons l'existence d'un *lexique* dont on attendra au minimum les caractéristiques suivantes :

— chaque unité lexicale a une entrée correspondante dans le lexique ;

— chaque entrée dans le lexique fournit une interprétation sémantique minimale sous la forme d'un lambda-terme typé ;

— lorsqu'une entrée correspond à un objet pointé, le lexique fournit la liste des aspects et des relations logiques qui lient ces aspects entre eux.

À ce stade du présent texte, nous estimons que ces caractéristiques suffiront pour les applications à venir. Néanmoins, il paraît important de souligner que nous ne réduisons par le lexique à ces caractéristiques, et qu'il peut donc être étendu librement, comme par exemple le font Pustejovsky et Batiukova [139]. Au cours de notre étude, il n'est pas exclu que nous en venions à formuler d'autres caractéristiques à ajouter en fonction des besoins.

Nous allons être amenés à étudier diverses théories de types et, comme nous avons pu le voir, les types sémantiques peuvent être déclinés en plusieurs niveaux. Nous avons en particulier appuyé la distinction entre types d'entités et types prédicats dans le système de Montague, et avec l'intégration de la sémantique lexicale, le type des entités s'est vu subdivisé en plusieurs sous-types auxquels nous nous sommes référés indifféremment par des termes tels que « types élémentaires » ou « types de base ». Si ces termes ont été suffisants pour les propos introductifs des sections précédentes, il nous semble essentiel de fixer à partir de maintenant une terminologie plus précise, afin de lever toute possibilité d'ambiguïté, et donner par là-même un cadre formel

sous lequel considérer les théories de types. Nous donnons donc ci-dessous quelques définitions formelles à cet effet.

Définition 1. *Une* ontologie de types *est un triplet* $(\mathbb{B}, e, \leq)$ *tel que* $\mathbb{B}$ *est un ensemble fini,* e *est un élément distingué de* $\mathbb{B}$*, et* $\leq \subset \mathbb{B} \times \mathbb{B}$ *est une relation d'ordre partielle sur* $\mathbb{B}$ *dont* e *est l'unique plus grand élément.*

On comprendra donc que l'ontologie de types désigne la collection des types issus de la sémantique lexicale, c.-à-d. les types classifiant les entités. On y retrouve bien entendu le type général e hérité de la théorie des types simples ; la relation $\leq$ est la relation de sous-typage entre ces types, et la condition de maximalité de e pour cet ordre rappelle donc que tous ces types forment bien une subdivision des entités. Les éléments d'une ontologie de types seront alors logiquement appelés des *types ontologiques*, terme qui remplacera désormais les appellations « types élémentaires » ou « types de base ».

Définition 2. *Un* système de constructeurs *est une paire* $(\mathbb{S}, a)$ *telle que* $\mathbb{S}$ *est un ensemble fini de symboles formels appelés* constructeurs de types*, et* $a : \mathbb{S} \to \mathbb{N}$ *est une fonction d'arité.*

Définition 3. *Un* système de types *est un ensemble de termes récursivement défini à partir d'un système de constructeurs, et d'une ontologie de types dont les éléments sont traités comme des constantes d'arité nulle.*

La définition des systèmes de types implique de possibles restrictions sur les compositions de symboles possibles, de sorte que certaines combinaisons soient interdites. On parlera de système de types *librement engendré* ou plus simplement *libre* lorsqu'aucune restriction n'est faite, c.-à-d. que tout terme généré peut figurer en n'importe quel position d'un constructeur d'arité suffisante. La subtilité qui distingue la conception d'un système de types par rapport à l'approche traditionnelle de la discipline sur ce sujet tient au statut de ce qu'on appelle généralement les types de base. En particulier, nous redéfinissons le statut du type des valeurs de vérité t à travers notre approche : on trouvera ci-dessous à gauche l'organisation traditionnelle des types simples, que nous comparons à droite avec l'organisation que nous proposons (les nombres entre parenthèses indiquent l'arité des constructeurs).

Organisation traditionnelle :

Types de base : $\{e, t\}$
Constructeurs : $\{\rightarrow (2)\}$

Notre approche :

Ontologie de types : $\{e\}$
Constructeurs : $\{t\ (0), \rightarrow (2)\}$

En effet, le type t ne désigne pas une catégorie d'entité, mais une proposition logique, que nous avions assimilé au début de ce chapitre à un prédicat d'arité nulle : par conséquent, il est exclu de l'ontologie et relegué au rang des constructeurs. Il en irait a priori de même pour des types additionnels comme le type des intensions s propre à Montague.

L'organisation des types simples telle que donnée à droite ci-dessus servira de base à l'élaboration de nos autres systèmes de types : on supposera généralement disposer au minimum du type e ainsi que des deux constructeurs t et $\rightarrow$ dans tout système considéré ici, sauf mention contraire explicite. [29] Soulignons néanmoins que la structure combinant ontologie et constructeurs est minimale mais pas exhaustive : il n'est pas exclu que l'on ait besoin d'ajouter d'autres

29. Notons que la plus petite ontologie de types possible est $(\{e\}, e, =)$; par conséquent, il n'est pas étonnant d'avoir toujours e dans notre système. Certaines théories des types, comme UTT, rejettent l'idée d'utiliser un tel type ontologique maximal. Le choix que nous faisons ici est donc un parti pris assez fort, que nous justifierons dans le chapitre 3.

ensembles de symboles à un système de type donné, comme p. ex. des variables de type pour du polymorphisme. De tels ajouts ne remettent pas en cause la définition 3 ci-dessus. Notons enfin que ce qu'on appellera ici *théorie des types* consiste formellement en un système de types accompagné d'une collection symbolique de *termes*, ainsi que d'un ensemble de *règles de typage* permettant d'associer certains termes à des types particuliers.

Cette base théorique nous permet donc d'exprimer les types sémantiques comme des arbres étiquetés, selon un principe bien connu qui place les constructeurs en nœuds ayant autant d'enfants que leur arité, et tel que les enfants d'un même nœud sont totalement ordonnés. Étant donné un type α quelconque d'un système de types fondé sur une ontologie $\mathbb{B}$ et un système de constructeurs $\mathbb{S}$, notons $\theta(\alpha)$ l'arbre associé. Une définition récursive classique nous permet d'écrire un tel arbre comme une paire $(s, [t_1, \ldots, t_n])$ où $s \in \mathbb{B} \cup \mathbb{S}$ est le symbole en racine de l'arbre, d'arité n, et $[t_1, \ldots, t_n]$ est la séquence ordonnée des sous-arbres de cette racine ; le cas de base (c.-à-d. les feuilles) étant bien sûr constitué d'un symbole d'arité nulle et d'une séquence vide. On remarquera alors que les types ontologiques sont toujours en feuilles de ces arbres. Cette vision des types comme des arbres nous permet alors de définir plus fondamentalement la notion de discordance. En préambule, nous aurons besoin de la définition récursive suivante.

Définition 4. *Deux arbres* $(s, [t_1, \ldots, t_n])$ *et* $(s', [t'_1, \ldots, t'_m])$ *formés à partir des mêmes symboles sont* fortement isomorphes *lorsque l'une des conditions suivantes est remplie :*

- $s, s' \in \mathbb{B}$ *(ce qui entraîne* $n = m = 0$*), ou*
- $s, s' \in \mathbb{S}$ *et* $s = s'$ *(ce qui entraîne* $n = m$*) ;*

et qu'en plus, pour tout $i \in [\![1, n]\!]$*,* t_i *et* t'_i *sont fortement isomorphes.*

Nous qualifions cette propriété de forte pour souligner les contraintes supplémentaires qui y sont imposées par rapport à la définition classique d'isomorphisme d'arbre, qui ne tient compte que de l'arité des nœuds : ici, elle tient compte de l'ordre des sous-arbres, et impose en plus une égalité des constructeurs rencontrés en même position dans les deux arbres, ce qui permet notamment de veiller à marquer la différence entre deux constructeurs distincts de même arité. En revanche, elle n'impose pas l'égalité sur les types ontologiques, ce qui la distingue donc de l'égalité d'arbres. Et naturellement, cette propriété s'étend aux types eux-mêmes : on dira que α et β sont fortement isomorphes si $\theta(\alpha)$ et $\theta(\beta)$ le sont. Ainsi, $a \to b$ est fortement isomorphe à $b \to c$, mais pas à $a \times b$. Grâce à cette propriété, nous pouvons enfin définir la discordance :

Définition 5. *Soient deux types* $\alpha \to \gamma$ *et* β*. On dit qu'il y a* discordance structurelle *entre ces deux types si* α *et* β *ne sont pas fortement isomorphes. Par ailleurs, on dit qu'il y a* discordance ontologique *lorsque* α *et* β *sont fortement isomorphes mais pas égaux, c.-à-d. qu'il existe deux feuilles d'étiquettes* $a \in \mathbb{B}$ *et* $b \in \mathbb{B}$ *à la même position dans* $\theta(\alpha)$ *et* $\theta(\beta)$ *respectivement tels que* $a \neq b$*.*

Nous terminons l'introduction de notre base théorique par une définition plus légère qui se rattache aux considérations de discordance vues précédemment, mises cette fois en perspective de la relation de sous-typage sous-tendant l'ontologie de type :

Définition 6. *Deux types ontologiques* a *et* b *sont* incompatibles *lorsque* $a \not\leq b$ *et* $b \not\leq a$*.*

Autrement dit, l'incompatibilité se définit exactement comme l'incomparabilité dans le cadre d'une telle ontologie.

Objectifs. Toutes les définitions fournies ci-dessus permettent donc de poser le cadre dans lequel nous allons développer notre étude. Comme nous avons pu l'évoquer, certaines des hypothèses qui régissent cette base — et nous pensons en particulier au type e comme élément maximal d'une ontologie — relèvent d'un parti pris assez fort, qu'il conviendra donc de justifier adéquatement. Nous pouvons donc considérer la justification de nos premières positions théoriques comme un objectif préliminaire de la présente thèse.

Néanmoins, avant de préciser nos objectifs principaux, il convient de rappeler le contexte dans lequel s'inscrit notre travail. Le domaine de la linguistique informatique peut se définir de manière assez large comme la science étudiant les représentations mathématiques de la langue naturelle et leurs applications, et l'un des axes principaux de cette discipline est justement la production de modèles de langues ; modèles parmi lesquels figure la proposition de Montague. Dans les sections précédentes, nous avons eu l'occasion de cerner les caractéristiques de cette proposition (dépendance de la sémantique aux informations syntaxiques, compositionnalité sous forme d'un morphisme, interprétations des unités lexicales par des lambda-termes élémentaires, distinction entre entités et prédicats dans le système de types) ainsi que ces limites (composition non-restreinte, surgénération), démontrant l'inadéquation des formalismes basés directement sur la sémantique montagovienne comme modèles de la langue naturelle. Le présent exposé s'inscrit alors dans la dynamique d'amélioration de cette catégorie de modèles, dont l'objectif ultime est la production d'un modèle symbolique reprennant les principales caractéristiques du système montagovien.

Plus particulièrement, la classe générale des modèles que nous cherchons à produire se veut au minimum compositionnelle, dirigée par la syntaxe, et basée sur une logique prédicative. De plus, nous inscrivons notre travail dans la droite lignée des formalismes intégrant des éléments de la sémantique lexicale. Comme nous l'avons vu dans la section 1.2, cet impératif d'intégration d'informations lexicales répond aux limites initiales du système montagovien, mais apporte avec lui de nouvelles questions à résoudre : la répartition de l'information sémantique entre types et lexique, la formalisation générale des coercions, et la définition concrète des types ontologiques. Ces questions ne sont pas d'ordre purement pratique : y apporter une réponse claire est fondamental pour limiter la sous-génération d'un tel formalisme, de sorte que le modèle puisse rendre compte de la capacité de la langue à produire des usages créatifs des mots dans dans contextes nouveaux. Par ailleurs, nous avons vu en section 1.3 que d'autres phénomènes sémantiques, qui ne sont pas basés sur des considérations lexicales, nécessitaient néanmoins des transformations de types que nous avons rapproché des coercions : cette similitude entre les deux phénomènes nous encourage à intégrer également ces transformations au sein de nos modèles, ce qui ouvre également la voie à d'autres modifications du système montagovien qui contribueront elles aussi à repousser les limites de celui-ci. À la lumière de ces discussions, nous avons alors identifié trois propriétés supplémentaires qui se doivent de caractériser les modèles que nous cherchons à produire : la simplicité, la flexibilité, et la modularité.

Mises bout-à-bout, toutes ces caractéristiques permettent ainsi de mieux cerner notre objectif : produire une classe de modèles répondant à toutes les propriétés évoquées au paragraphe précédent, capturant autant que possible l'expressivité des propositions précédentes (GL, TCL, UTT, MGL) dans une interface syntaxe-sémantique basée sur la sémantique montagovienne. Nous souhaitons bien sûr que ce modèle soit aussi précis que possible au regard de la langue elle-même, mesurant donc intuitivement son adéquation à sa capacité à faire varier ses interprétations de manière proportionnée par rapport aux variations des énoncés eux-mêmes. Pour cela, nous séparons notre problématique en deux axes principaux : un axe lexical et un axe formel.

L'axe formel peut être considéré comme le cœur du présent exposé, puisque c'est celui qui contribue le plus à l'ancrer dans le domaine de l'informatique. Son objectif majeur est de définir

les contours mathématiques de la mécanique compositionnelle dont nous avons besoin pour notre modèle. Il s'agit en particulier de fournir une réponse à la question (Q1) suivante :

(Q1) *Étant donné une ontologie de types et un système de constructeurs, de quel système et de quelle théorie des types avons nous besoin pour définir un formalisme autorisant des transformations de types cohérentes avec les observations linguistiques ?*

On relèvera dans cette question l'hypothèse d'avoir la base formelle du système de type déjà fournie : comme nous le verrons juste après, la détermination de cette base sera l'apanage de l'axe lexical. L'impératif de cohérence avec les observations linguistiques signifie que les opérations non abstraites, qui modifient directement la forme des lambda-termes, doivent respecter au mieux le comportement de la langue : lorsque de multiples manières de transformer un terme en un autre sont possibles, il conviendra alors de sélectionner les plus adaptées à notre objectif de modélisation. Cet axe consistera donc principalement à formuler une théorie des types capable de rendre compte des propriétés désirées pour un modèle de sémantique montagovienne avec des informations lexicales.

L'axe lexical, quant à lui, va regrouper plusieurs questions qui reposeront davantage sur une base linguistique et philosophique. En effet, quand bien même le présent exposé s'inscrit dans le domaine de l'informatique, il n'en reste pas moins que son sujet traite d'éléments linguistiques, et ignorer les apports de ce domaine dans un tel contexte serait une erreur que nous nous gardons bien de commettre. Il ressort notamment des discussions menées dans ce chapitre qu'il subsiste un certain flou entourant la nature des types sémantiques tels qu'utilisés dans les modèles formels, et s'il est bien beau de construire des théories de types pour la modélisation linguistique, c'est encore mieux si ces théories sont fondées directement sur des observations linguistiques, car elles offriront ainsi une meilleure précision. Comme sous-entendu dans la présentation de l'axe formel, notre objectif ici sera de fournir les bases linguistiques qui permettront de compléter nos propositions de théories des types, en répondant aux questions suivantes :

(Q2) *Quelles sont les propriétés à attendre des types ontologiques, et comment ces derniers sont-ils structurés ?*

(Q3) *Quels constructeurs de types sont indispensables pour rendre compte des observations linguistiques ?*

On retrouvera dans la question (Q2) la nécessité de justifier les bases posées dans notre cadre théorique, tout en explicitant quelle partie de l'information sémantique est représentée par les types ontologiques ; y répondre permettra donc de consolider notre cadre et de poser d'éventuelles constraintes sur notre théorie des types. Quant à la question (Q3), elle a pour but d'expliciter le système de constructeurs strictement nécessaire à notre modélisation.

Nous ajoutons à cet axe lexical une dernière question (Q4) ci-dessous, destinée à fournir des éléments sur la population concrète de l'ontologie de types :

(Q4) *Quelles méthodes permettraient de déterminer une ontologie de types optimale, et à quoi pourrait ressembler une telle ontologie ?*

À ce stade, nous ne prétendons pas résoudre complètement la question des types ontologiques : produire une ontologie adéquate pourrait prendre beaucoup plus que de temps que ne le permet la thèse présentée ici, voire faire l'objet d'un projet de recherche à part entière. Néanmoins, nous espérons ici pouvoir ouvrir de nouvelles perspectives sur ce problème resté jusqu'à présent très abstrait. Notre exploration de ce problème mobilisera des éléments de linguistique de corpus et des outils informatiques plus appliqués que dans l'axe formel, c'est pourquoi nous la distinguons des deux autres questions de l'axe lexical.

Plan général. Le présent exposé ordonne les discussions sur les différentes questions posées ci-dessus en trois parties. Dans un premier temps, les chapitres de la partie I aborderont les questions (Q2) et (Q3) d'un point de vue linguistique et philosophique, afin de poser les bases de l'ontologie de types et du système de constructeur. Puis, la partie II implémentera le cœur de l'exposé en développant la théorie des types destinée à répondre à (Q1), en utilisant les résultats de la partie précédente. Enfin, la partie III sera consacrée à l'étude de la question (Q4) et à la formulation de pistes de réponses.

Première partie

Études linguistiques

Chapitre 2

Contributions historiques à la notion de type sémantique

> *The* class *of things which do not exist belong to the* category *of existence.*
>
> Fred Sommers [171]

L'objectif du présent chapitre est de montrer que la notion de type sémantique telle que nous la comprenons aujourd'hui est le résultat composite de plusieurs décennies de réflexions logiques, philosophiques et ontologiques autour du langage, sa manière de le construire et son rapport au monde qui nous entoure. Il y a plus d'un siècle, des disciplines qui nous paraissent maintenant très éloignées les unes des autres l'étaient alors beaucoup moins : la toute jeune logique mathématique était en effet très liée à la langue, puisque c'est au travers de celle-ci que sont formulées les propositions logiques, ce qui amena de nombreuses discussions sur le statut de la langue au regard de la pensée, au regard de la logique et au regard de notre expérience empirique du monde. Sur ce terreau fertile ont emergé de nombreuses idées qui ont contribué à forger la logique moderne d'une part, et la sémantique formelle d'autre part. J'estime, et tâcherai de défendre ici, que la notion de type sémantique a également fait partie de ces discussions du XX^e^ siècle, du moins sous des formes approchées qui, d'une manière ou d'une autre, ont influencé la conception moderne de cette notion. [30]

Le présent chapitre a ainsi pour vocation d'introduire les différentes idées qui nous semblent importantes pour une meilleure compréhension des principes et de l'utilité des types sémantiques, avec les noms des actrices et acteurs associés au développement de ces idées. Nous n'avons nullement la prétention d'être exhaustif ici, mais espèrons dresser un panorama suffisamment

30. On retrouve en effet dans la littérature une grande diversité d'appellations pour des concepts plus ou moins proches, principalement désignés en anglais par les mots suivants : *genus*, *class*, *type*, *category*, *sort* et *kind*. Il est frappant de voir que ces termes, dont les usages apparaissent plus hésitants au début du XX^e^ siècle (une situation en partie imputable à des divergences de traduction depuis les sources allemandes ou polonaises vers l'anglais), se sont aujourd'hui standardisés dans des connotations beaucoup plus spécifiques. Afin de ne pas ajouter plus de confusion à ce tableau déjà complexe, nous réservons à chaque section de ce chapitre l'usage de termes spécifiques pour rendre compte des notions historiques, mais userons exclusivement du mot *type* pour parler des types sémantiques ; le terme *classe* sera reservé à son usage mathématique courant, et le terme de *catégorie* sera restreint dans le présent chapitre au concept philosophique initié par Aristote et qui a abouti à l'expression « erreur de catégorie » (cf. section 2.5). En dehors de cet usage philosophique et outre son usage commun dans les contextes non ambigus, ce dernier terme désignera principalement dans la suite les objets mathématiques du même nom (cf. chapitre 4).

vaste pour permettre une théorisation des types sémantiques aussi complète que possible, tenant compte d'un maximum d'aspects au regard de la syntaxe et de la sémantique. Cette théorisation, dont nous posons seulement les bases, fera l'objet du prochain chapitre. Dans la suite du chapitre actuel, nous répartissons les contributions en question dans cinq groupes : tout ce qui a trait à l'origine de la théorie des types simples sera traité en section 2.1 ; les idées se référant à la notion historique de *catégorie sémantique* sera abordé en section 2.2 ; les influences de la grammaire générativiste et les autres liens entre types et syntaxe feront l'objet de la section 2.3 ; les apports de la sémantique lexicale traditionnelle à ce tableau seront décrits en section 2.4 ; et la notion aristotélicienne de catégorie et son héritage dans la philosophie du XX^{e} siècle seront présentés en section 2.5.

2.1 La théorie des types simples

Notre histoire, comme beaucoup d'histoires ayant trait à la logique et à la sémantique formelle, commence avec Frege. Dans [63], ce dernier introduit notamment la distinction entre fonctions et objets au sein du langage, soulignant que certains termes du langage pouvaient être vus comme « non-saturés », c.-à-d. qu'il est nécessaire de les combiner avec d'autres mots pour obtenir un sens qui soit complet : par exemple, une séquence comme « la capitale de » contient une place vide, qui invite à être complétée par une autre séquence, p. ex. « la France », afin d'obtenir un sens final satisfaisant. Cette intuition d'une place vide, d'un élément manquant pour compléter le sens de « la capitale de », a mené Frege à faire l'analogie entre de telles séquences et les fonctions mathématiques, permettant ainsi une analyse du langage en termes de fonctions et d'arguments qui a ouvert la voie au principe de compositionnalité que nous connaissons aujourd'hui. La vision très mathématique de Frege sur le langage l'amène rapidement à conclure qu'il est nécessaire, au nom de la rigueur scientifique, d'imposer des délimitations fortes sur les arguments effectivement substituables aux variables mathématiques, de sorte qu'une combinaison de la forme $\odot + 1$, où $\odot$ est un symbole désignant le soleil, ne puisse pas être exprimée, puisqu'elle ne désigne aucun objet (mathématique ou concret) possible : elle est sans référence.

L'attachement de Frege à l'impératif de référence se comprend aisément d'un point de vue logique : il s'agit de s'assurer que tous les objets mathématiques que l'on manipule sont toujours bien définis, et que les formules qui les utilisent ont toujours une valeur de verité, afin d'éviter la construction de raisonnements erronés sur la base d'objets qui n'existent pas. Mais cette analyse s'étend également aux concepts exprimés par le langage : Frege pose comme conditions indispensables à une analyse rigoureuse que l'on puisse toujours déterminer si un objet appartient à un concept ou non, et qu'une fonction ait toujours une valeur sur chacun de ses arguments. Cette délimitation de la portée des fonctions a bien entendu fait son chemin en mathématiques, où toute fonction est normalement présentée avec son ensemble de définition afin d'éviter la production de valeurs impossibles, telles que la racine carrée ou le logarithme d'un nombre strictement négatif. D'un point de vue linguistique, nous pouvons aussi voir dans ces conditions une première intuition des types sémantiques : délimiter le domaine d'application des fonctions du langage revient en effet à imposer une restriction sur les expressions qui peuvent être passées en argument d'une expression donnée. Ainsi, la fonction « la capitale de » n'a intuitivement de valeur que si l'argument désigne une région ou un pays, excluant de fait de nombreux syntagmes qui pourraient théoriquement (c.-à-d. d'après les règles de la grammaire) occuper cette place d'argument.

Cependant, cet aspect développé par Frege s'est retrouvé progressivement occulté dans ses propres travaux, notamment dans sa discussion sur les notions bien connues de sens et de réfé-

rence dans [64], où il se focalise davantage sur la question des noms propres et des propositions subordonnées, reléguant le problème plus général de la combinaison des expressions au second plan. Il en ressort alors un principe bien plus simple : si une combinaison de mots est grammaticale, elle peut avoir une référence ou pas ; et si elle n'en a pas, toutes les expressions dans lesquelles cette combinaison apparaît n'en auront pas non plus — considérant que la référence d'une phrase complète correspond à sa valeur de vérité. Enfin, les expressions possèdent un sens, aussi appelé « mode de présentation », permettant de caractériser la manière dont les expressions désignent des objets : l'exemple célèbre associée à cette notion sont les expressions « étoile du matin » et « étoile du soir », qui ont toutes les deux pour référence la planète Vénus, mais qui ont pour Frege des sens différents. Les discussions et critiques autour de ces notions sont nombreuses et nous aurons l'occasion de revenir sur certaines d'entre elles au prochain chapitre ; notons toutefois que nous ne nous étendrons pas davantage sur la philosophie de Frege dans ces pages, car elle n'apporte plus vraiment à ce stade de prémisses pertinentes à notre notion de types sémantiques.

La pensée de Frege a cependant eu une certaine influence sur Bertrand Russell, qui cherchait tout comme lui à poser des fondements solides à la logique et aux mathématiques. Si les deux sont souvent rapprochés en dépit de certains désaccords philosophiques (cf. p. ex. [185]), Russell est néanmoins reconnu pour avoir consolidé encore plus les formalismes de Frege par l'introduction de la théorie des types [155], destinée à prémunir la logique contre les paradoxes ensemblistes qui ont été découverts, par Russell et par d'autres, durant les premières années du XX^{e} siècle. [31] Cette prévention des paradoxes est pour Russell la motivation principale à l'introduction de sa théorie des types, mais celle-ci formalise également — bien que Russell ne semble pas le mentionner explicitement dans ses écrits — deux principes de sens commun déjà appliqués en mathématiques à son époque : le fait qu'une variable a toujours une plage de valeurs bien délimitée, et le fait qu'une distinction doit être faite entre une fonction et ses arguments [65]. Dans la théorie de Russell, asserter une propriété sur un objet donné revient alors à instancier une variable dans une fonction propositionnelle, et Russell définit alors sa notion de *type* comme « *the range of signifiance of a propositional function, i.e., [...] the collection of arguments for which the said function has values* » [155, p.236]. On retrouve bien dans cette formulation l'intuition de Frege exposée plus haut.

Cette définition permet en particulier à Russell de définir une hiérarchie d'*ordres*, imposant à toute fonction d'avoir un ordre strictement supérieur à celui de ses arguments, de sorte qu'elle ne puisse pas s'appliquer à elle-même, et assertant l'existence d'objets d'ordre 0, qui ne contiennent aucune variable : les *individus*, qui forment une classe spécifique. Mais sa théorie contient une seconde hiérarchisation, appelée *niveau*, et ayant trait aux variables liées dans ces mêmes fonctions. Il en résulte une théorie trop complexe, et reposant sur un axiome dit de réductibilité qui pose de nombreux problèmes de cohérence. Comme expliqué par plusieurs revues historiques [65, 57], cet axiome a été attaqué indépendamment par Chwistek [43] et Ramsey [147], menant à son abandon et à l'effondrement de cette seconde hiérarchie au sein d'une nouvelle théorie, plus simple d'utilisation, que l'on connaît désormais sous le nom de *théorie des types simples*. Cette dernière a connu au cours de la décennie suivante de nombreuses formulations détaillées, notamment par Carnap [27], Gödel [74] et Tarski [176], mais on retiendra surtout la formulation de Church [42], qui est le premier à introduire un type spécifique pour les propositions elles-mêmes et à systématiser la construction de types fonctionnels à partir de ce nouveau type (cf. [85]

31. On citera à titre d'exemple le paradoxe de Russell lui-même, montrant que si l'on peut définir l'ensemble U des ensembles qui se ne contiennent pas eux-mêmes, alors $U \in U$ et $U \notin U$ sont logiquement équivalents, résultant en une théorie incohérente. Russell énumère de nombreux autres exemples dans [155, §1].

pour plus de détails sur ces différentes formulations). Bien entendu, les applications de cette théorie sont a priori mathématiques avant tout, mais souvenons-nous que pour les philosophes de cette époque le langage n'est jamais loin : les expressions non-saturées sont ainsi assimilées aux fonctions, et les expressions saturées aux individus, posant ainsi les bases de la mécanique compositionnelle que l'on retrouve notamment dans les travaux de Montague.

Pourtant, quelque chose manque à ce tableau, comme nous avons pu le voir au chapitre 1 : la possibilité d'exprimer des restrictions de sélection plus précises pour les fonctions, allant au-delà de la notion d'ordre. Pour reprendre notre exemple précédent, il est certain que tous les arguments de « la capitale de » sont des individus et non des fonctions d'ordre supérieur, mais la théorie des types simples ne semble pas fournir initialement d'outils pour exprimer clairement qu'une classe particulière d'individus est attendue. Des réflexions allant dans ce sens ont cependant été proposées par Alan Turing, qui avait une attitude bien plus pragmatique à l'égard de la théorie des types que les philosophes précedemment cités. Il est intéressant de remarquer que ses remarques s'appuient une fois de plus sur la langue naturelle ; il dit notamment :

> The type principle is effectively taken care of in ordinary language by the fact that there are nouns as well as adjectives. We can make the statement 'All horses are four-legged', which can be verified by examination of every horse, at any rate if there only a finite number of them. If however we try to use words like 'thing' or 'thing whatever' trouble begins. Suppose we understand 'thing' to include everything whatever, books, cats, men, women, thoughts, functions of men with cats as values, numbers, matrices, classes of classes, procedures, propositions, ... Under these circumstances what can we make of the statement 'All things are not prime multiples of 6'[?] We are of course inclined to maintain that it is true, but that is merely a form of prejudice. What do we mean by it? [180]

Turing reconnaît ainsi le rôle particulier des noms comme caractérisant des individus, les distinguant notamment des adjectifs en tant qu'expressions qui réfèrent. Sa discussion de *thing* comme terme problématique souligne en quoi l'obtention d'une signification logiquement expressible nécessite des quantifications qui soient correctement restreintes. Une fonction propositionnelle comme « *is not a prime multiple of 6* » est restreinte aux individus par le principe même de la théorie des types simples, mais tous les individus ne sont pas aptes à produire une valeur de vérité si on leur applique une telle fonction, du moins pas sans appliquer « a form of prejudice », comme l'énonce Turing.

Intuitivement, les individus concernés par cette fonction sont principalement les nombres entiers, ce qui exclut la plupart des individus, et le rôle des noms est de fournir la classe de base sur laquelle évaluer cette fonction. Turing argue en faveur de l'usage de gardes pour les quantifications afin de respecter cet impératif, préconisant l'usage de formules logiques de la forme $\forall x \in S.\phi(x)$ plutôt que $\forall x.x \in S \Rightarrow \phi(x)$, où les ensembles S acceptables sont justement présentés sous le nom de *noun-classes*, et construits comme des sous-ensembles des éléments d'un ordre donné dans la théorie des types. Les *noun-classes* ont notamment une forte ressemblance avec ce que l'on connaît aujourd'hui comme les types compréhensions ou types sous-ensembles (cf. p. ex. [161]), qui vus en tant que classes possèdent une forme primitive de sous-typage. Par leur usage, l'utilisation des fonctions est alors restreinte comme attendu, assurant ainsi que l'expression correspondante est signifiante. Il paraît raisonnable de voir dans cette construction une première base intuitive du découpage de la classe des individus en sous-classes, qui préfigure la division du type des entités en sous-types dans nos formalismes modernes.

La comparaison est d'autant plus frappante que, selon Gandy [66], Turing avait conçu préalablement à l'élaboration de sa théorie des *noun-classes* une méthode dite des types virtuels, permettant d'introduire dans le système de Church de nouveaux types pour caractériser certains sous-ensembles de types déjà existants — l'utilité de cette méthode s'étant notamment

affirmée pour le sous-ensemble des entiers de Church.[32] Les *noun-classes*, héritiers directs de cette notion de type virtuel, sont donc à la base des systèmes de types modernes où les individus sont répartis dans différents types en fonction de leur propriétés — en particulier, nous pouvons raisonnablement supposer que ce concept a contribué à l'émergence des types en langages de programmation. En sémantique formelle, la nécessité de restreindre les quantifications préfigure de nombreuses discussions autour du statut du nom commun et de sa représentation sous forme de type ou de prédicat.

La suite, nous la connaissons, puisque la théorie des types simples de Church est au cœur du système montagovien : il ne nous est pas utile de la répéter ici. On doit à la théorie des types l'introduction du terme même de *type*, sans lequel la présente thèse n'aurait pas lieu d'être. L'aspect primordial de cette théorie est de reconnaître une distinction formelle entre les fonctions et les arguments, qui par l'analogie de Frege s'appliquent également aux expressions du langage, selon qu'elles soient saturées ou non. La vision frégéenne des expressions incomplètes comme des fonctions est indispensable lorsque que l'on souhaite traiter la sémantique à l'aune du principe de compositionnalité, et délimiter proprement la plage des valeurs pouvant instancier les variables de ces fonctions s'impose rapidement comme une étape incontournable de la modélisation logique du langage. Il convient alors de souligner que la théorie des types telle qu'on l'a décrite dans les pages précédentes est imparfaite à ce niveau, car elle ne retient, au travers de la notion d'ordre, qu'une partie incomplète des restrictions réellement nécessaires. On retrouve néanmoins dans les premiers travaux de Frege ainsi que dans ceux de Turing l'intuition que des subdivisions au sein même des types d'un ordre donné sont nécessaires pour rendre compte de la portée réelle des fonctions qui acceptent des arguments de cet ordre ; cette intuition préfigure d'une certaine manière la création de types lexicaux comme observés en section 1.2, bien que la notion de sémantique lexicale soit totalement étrangère aux travaux en question.

2.2 Les catégories sémantiques

En parallèle du développement de la théorie des types s'est developpée une autre forme de logique, dite *catégorielle*, reprenant la base des idées exprimées par les catégories d'Aristote (cf. section 2.5) et initiée par le philosophe Edmund Husserl dans son second tome des *Recherches logiques*. En partant du constat intuitif que certaines suites de mots avaient du sens et d'autres non, Husserl a tenté de formaliser les règles qui contraignent le langage pour déterminer quelles suites de mots pouvaient effectivement avoir du sens [19]. C'est cette recherche de principes formels qui l'a conduit à introduire sa notion de *catégorie sémantique*, qui permet de distinguer facilement les combinaisons de mots qui ont du sens de celles qui n'en ont pas. Étant donné un contexte significatif quelconque, c.-à-d. un énoncé qui a du sens, la catégorie sémantique d'un terme de ce contexte est l'ensemble des termes qui lui sont substituables tout en maintenant ce que Husserl appelle l'unité de signification du contexte, autrement dit la capacité de l'énoncé à conserver du sens.

Mais Husserl, pris dans son programme apodictique et son rejet du psychologisme, souhaitait avoir une notion d'unité de signification qui soit essentielle et indépendante des jugements subjectifs, au rang desquels figure l'organisation mentale elle-même. De cette vision radicale de la signification naît une distinction importante sur les conditions qui contraignent la bonne

32. Les entiers de Church sont un encodage des entiers en lambda-calcul simplement typé, formant un sous-ensemble des termes fermés de type $(e \to e) \to e \to e$. Le principe est de définir la représentation d'un entier n par le lambda-terme $\lambda f \lambda x.f^n x$, où $f^n x$ représente n applications successives de f à x. Le nombre 0 est alors défini comme $\lambda f \lambda x.x$, et la fonction successeur par $\lambda n \lambda f \lambda x.(nf)(fx)$; les autres opérations comme l'addition, la multiplication ou encore la fonction prédecesseur sont également définissables.

formation des phrases : une phrase peut échouer à avoir du sens selon des critères objectifs et absolus, ou au contraire selon des critères purement subjectifs et dépendants de l'organisation mentale. Le premier cas est celui de l'*Unsinn*, ou « non-sens », caractérisant les expressions qui n'ont aucune signification intrinsèque, et qui s'oppose au *Widersinn*, ou « contresens », caractérisant les expressions signifiantes qui échouent à être vraies, cohérentes ou référentes [24]. Les règles objectives de Husserl, et donc ses catégories sémantiques, sont construites pour rendre compte uniquement de l'*Unsinn*, ce qui amène à des considérations qui peuvent se révéler contre-intuitives de prime abord : en effet, pour Husserl l'unité de signification d'un énoncé comme « cet arbre est vert » s'abstrait dans sa forme « pure » par le schéma « ce S est P », et son unité de signification est maintenu par toute subsitution de S par une « matière nominale » et de P par une « matière adjectivale » [19]. Il en résulte notamment que la phrase en (1a) ci-dessous possède une unité de sens, alors que la phrase en (1b) n'en possède pas parce que le prédicat *égal* est diadique, et non pas monadique comme *vert* dans le contexte initial.

(1) a. Cet arbre est divisible par 2.
b. Cet arbre est égal.

Par conséquent, *vert* et *divisible par 2* sont substituables l'un par l'autre et appartiennent donc à la même catégorie sémantique.

À ce critère de substituabilité s'ajoute une considération supplémentaire au travers de la distinction entre catégories de bases et catégories fonctorielles : les premières consituent des significations intrinsèques et indépendantes, tandis que les secondes sont des significations qui ont un besoin essentiel de complément, et ne peuvent exister qu'en tant que parties de significations plus vastes. Cette distinction est très frégéenne dans l'esprit puisque ces catégories fonctorielles remplissent sensiblement les mêmes usages que les expressions insaturées, attendant un argument obligatoire provenant d'une catégorie de base : au sein de formalismes plus modernes, telle la théorie de Church, ces catégories sont assimilées à des fonctions. L'objectif d'Husserl était alors d'identifier les catégories sémantiques du langage et de dégager les lois qui régissent les combinaisons des éléments de ces différentes catégories. Il en a résulté une esquisse assez informelle et incomplète de règles, qui avait néanmoins l'intérêt de présenter une méthode de construction récursive des catégories sémantiques (cf. [19]). Les bases ainsi posées par Husserl ont cependant eut une certaine influence sur l'école philosophique dite de Lvov-Varsovie (abrégé dans la suite en « École polonaise »), dont des membres éminents ont repris à leur compte l'idée husserlienne des catégories sémantiques en renforçant la théorie formelle régissant leur construction [33, 19, 189].

Il apparaît que l'École polonaise des années 20 eut une réception plutôt mitigée de la théorie des types de Russell. Comme évoqué dans la section précédente, Chwistek fut l'un des premiers à remettre en cause la ramification et la présence de l'axiome de réductibilité dans la théorie des types originelle, et ce sentiment de perplexité face à cette théorie était partagée par de nombreux autres membres de l'École polonaise. C'était notamment le cas de Leśniewski (cf. [19]) qui, bien que considérant le paradoxe de Russell comme l'un des problèmes centraux de la logique, reprochait à la théorie des types de ne pas rendre correctement compte de l'intuition sémantique : selon lui, le paradoxe était en effet un problème d'intuition plus que de formalisme, et devait donc être résolu en conséquence. C'est pourquoi il préféra construire un système alternatif qui permettrait de formaliser correctement ses intuitions, en s'appuyant sur la notion de catégorie sémantique.

Le système logique de Leśniewski repose sur trois théories axiomatiques que nous ne détaillerons pas ici, et permet la construction d'une hiérarchie de catégories sémantiques fonctorielles, précisant la catégorie des arguments et la catégorie de la fonction, à partir des catégories de

bases que sont les noms et les propositions.[33] Les principes régissant le langage logique résultant affirment notamment que chaque expression doit disposer d'une seule et unique catégorie associée et que les expressions de même catégorie sont substituables les unes aux autres, d'une façon similaire à la définition d'Husserl. De fait, cette approche s'apparente formellement à la théorie des types de Russell, mais en diffère selon un critère intuitif, pour lequel elle s'apparente davantage à la vision husserlienne justement, et par extension à la tradition aristotélicienne. L'unicité de catégorie pour chaque symbole ou expression permet alors de résoudre le paradoxe de Russell, puisque qu'une catégorie fonctorielle ne peut jamais avoir la même catégorie que son argument. À l'époque de son introduction, cette théorie se démarquait de la théorie des types notamment par sa richesse, puisqu'elle permettait de construire récursivement n'importe quelle catégorie possible. La théorie des types simples ne rattrapera ce retard qu'une dizaine d'années plus tard grâce à la formulation de Church et l'introduction du type des propositions.

Le formalisme de Leśniewski est cependant trop formel dans sa construction pour être employé tel quel dans l'interprétation du langage ordinaire, bien qu'il soit très efficace pour des langages plus mathématiques. C'est alors qu'intervient Ajdukiewicz (cf. [19, 33]), qui approfondit les travaux de son collègue en développant une théorie plus adaptée au langage. Ses catégories sémantiques sont là encore définies comme des classes d'équivalence d'expressions et de mots pour la relation de substitution au sein d'une phrase sensée. Au travers de son analyse, on doit notamment à Ajdukiewicz la reconnaissance que l'ordre des mots dans une phrase ne reflète pas l'ordre de composition de leurs significations respectives, ouvrant alors la voie vers ce qui deviendra plus tard l'analyse des structures syntaxiques. En outre, il préfigure le développement des grammaires catégorielles par l'introduction de notations encore en usage de nos jours : les catégories des noms et des propositions sont notées `n` et `s` respectivement, et les catégories fonctorielles sont notées à l'aide de la barre fractionnelle « / ».

Il est cependant frappant de constater que les théories d'Husserl, de Leśniewski et d'Ajdukiewicz qualifient leurs catégories de sémantiques alors que, comme nous avons pu le sous-entendre au fil des paragraphes précédents, ces catégories ne sont pas si sémantiques que ça. En réalité, la construction des catégories sémantiques se rapproche plus d'un formalisme grammatical, et donc syntaxique ; c'est notamment ce que relève Bar-Hillel [13], à qui l'on doit l'introduction des grammaires catégorielles modernes. Ce dernier rejette l'approche apodictique d'Husserl, qu'il trouve difficilement justifiable et qui ne semble contenir selon lui guère plus qu'une sorte d'intuition grammaticale peu sophistiquée : cela saute en effet aux yeux lorsqu'on rappelle les critères d'unité de sens d'Husserl, qui considère notamment la phrase (1a) comme sensée — à contrepied de nos intuitions modernes. Cette caractéristique se retrouve également dans les formalismes de l'École polonaise, qui apparaissent pour leurs auteurs comme plus intuitifs que la théorie des types justement parce qu'ils sont plus proches de la structure observable du langage, qui est syntaxique. Les critères de sens d'Husserl s'y retrouvent également : pour Ajdukiewicz, il n'y a ainsi pas de différence entre les deux phrases suivantes au regard de la cohérence du sens [72] :

(2) a. Le soleil brille.
b. Le soleil siffle.

Cela tient au fait qu'il considère que « brille » et « siffle » appartiennent à la même catégorie, qui s'avère correspondre à celle des verbes intransitifs ; pourtant, la similarité sémantique entre ces deux verbes est peu raisonnable à nos yeux, et notre intuition sémantique moderne nous inclinerait davantage à considérer la phrase (2b) comme dénuée de signification.

33. Malgré l'insatisfaction de Leśniewski avec la théorie de Russell, les deux s'accordaient au moins sur la nécessité d'une forme de hiérarchie formelle pour le fondement de systèmes cohérents [33, §1.2].

Il n'en reste pas moins que ces catégories sémantiques ont une importance historique certaine, et se retrouvent jusque dans les formalismes de Montague, comme expliqué dans [73]. Tout comme Bar-Hillel, Montague reconnaît cependant que sous l'influence des idées husserliennes ces catégories se rapprochent de la grammaire, et utilise dans [120] des catégories très similaires qu'il qualifie cette fois explicitement de syntaxiques. Par ailleurs, il n'hésite pas à revenir dans son traitement de la sémantique aux idées de Frege et Tarski, introduisant notamment une approche dénotationnelle caractérisée par l'interprétation des propositions comme des valeurs de vérités, rompant avec la vision d'Ajdukiewicz. L'homorphisme de Montague parvient notamment à dissocier formellement quelque chose qui, chez Ajdukiewicz et ses prédécesseurs, ne faisait qu'un : la syntaxe d'une part, et la sémantique d'autre part. Mais cet homomorphisme montre aussi une certaine correspondance entre les notions de catégorie sémantique et de type logique tel que défini dans la théorie des types. Ainsi, en dépit de la qualification sémantique de ces catégories, elles n'apportent finalement que peu d'informations sémantiques sur la conception moderne des types ; en revanche, il convient de reconnaître à travers cette idée de catégorie sémantique le lien fort entre syntaxe et sémantique d'un point de vue compositionnel, justifiant notamment l'introduction de systèmes d'interfaces syntaxico-sémantiques tels que celui de Montague. Par ailleurs, comme évoqué dans [72], les catégories sémantiques ont (paradoxalement) posé des bases utiles au développement des théories syntaxiques de Carnap d'abord, et par la suite de Chomsky. Or, comme nous allons le voir dans la section suivante, il y a également des liens intéressants à remarquer entre ces théories et notre conception moderne des types sémantiques.

2.3 La hiérarchie générativiste

La période du développement des formalismes de Leśniewski et d'Ajdukiewicz est également celle qui a vu le développement des premiers travaux de Rudolf Carnap sur une théorie logique de la syntaxe, dans la continuité des idées exprimées par Wittgenstein dans son *Tractatus* [187]. Pour celui-ci, il est nécessaire de recourir à un symbolisme exempt des ambiguïtés du langage naturel et qui obéit à des lois logiques, se plaçant en faveur d'une analyse des énoncés du langage par des formules logiques, ces dernières ayant alors pour rôle de préfigurer la signification de façon univoque [187, §3.323 à 3.327]. De là découle l'importance d'une syntaxe logique qui rende compte des combinaisons possibles et correctes des signes du langage. Carnap s'attelle donc à la construction d'une telle théorie qui, bien que centrée sur des langages plutôt formels, se pose comme une première étape dans la construction de systèmes syntaxiques aptes à représenter la langue naturelle (cf. [72, §1.1]). Une conséquence majeure de ce projet et de sa filiation avec la pensée de Wittgenstein est le traitement par Carnap des mots comme des signes vides de toute signification, rejettant toute influence de la sémantique ; de sorte que le langage soit réduit à un calcul formel en application de [187, §3.33]. Ainsi, la théorie de Carnap met l'accent non pas sur les mots, mais sur les règles d'inférence qui permettent par la suite de déterminer la signification des symboles.

En d'autres termes, l'objectif de Carnap est de poser des conditions formelles pour qu'une proposition ou un mot ait une signification. Grâce à sa théorie logique, il établit notamment qu'une proposition, au-delà des signes qui la composent, est spécifiée par une structure syntaxique et logique. De cette remarque naît une distinction fondamentale entre trois types de propositions : les *analytiques*, qui sont vraies par leur forme même parce que leur structure syntaxique contient un théorème logique ; les *contradictoires* qui sont fausses par leur forme même ; et les *synthétiques* dont la vérité doit être vérifiée par des observations empiriques. [34] Quant à

34. On notera, comme relevé par Godart-Wendling [72], une certaine tension dans les objectifs de Carnap,

la signification des propositions, elle passe par la recherche systématique de toutes ses conséquences logiques, avec pour corollaire un traitement purement syntaxique de la synonymie dans la mesure où deux propositions synonymes sont conséquences l'une de l'autre.

Pour constituer son système logique, Carnap [28] demande que chaque mot soit associé à une *catégorie syntaxique* ainsi qu'à une série de définitions, nommées *énoncés protocolaires*, qui doivent être vérifiables de façon immédiate. La notion d'immédiateté entendue ici est assez vague, de l'aveu de Carnap lui-même, mais suffit à remplir ses exigences empiricistes : intuitivement, un énoncé protocolaire doit pouvoir être vérifié sans nécessité de réflexion, en se basant uniquement sur ce qui est observable. Pour un mot comme *pierre*, que Carnap présente comme appartenant à la catégorie des choses, la définition associée est immédiate puisque l'énoncé « x est une pierre », si tant est que x est substitué par un élément de la bonne catégorie syntaxique, est censé se vérifier facilement : on peut déterminer à l'aide d'expériences très simples (regard, toucher) si oui ou non l'objet désigné est une pierre. Un mot comme *arthropode* est en revanche défini selon lui par plusieurs énoncés protocolaires tels que « x est un animal », « x a un corps articulé », et ainsi de suite ; de sorte que la catégorisation de chaque mot peut en réalité monter à des degrés élevés de précision. Mais pour Carnap, ces définitions sont formelles, et donc syntaxiques : les définitions ne sont rien de plus que des équivalences logiques entre un énoncé élémentaire (« x est un arthropode ») et la conjonction de ses énoncés protocolaires.

De là naît un certain paradoxe : en dépit du rejet de toute influence de la sémantique sur sa théorie, l'exigence empiriciste de Carnap le pousse à considérer des catégories syntaxiques très précises, qui dépassent le simple cadre de la grammaire, pour la simple raison qu'il lui faut écarter du langage toute proposition qui serait impossible à vérifier en examinant les énoncés protocolaires associés ; mais ces nouvelles catégories tendent en pratique à incarner une certaine connaissance *lexicale* des mots. Considérons la phrase en (3) ci-dessous :

(3) César est un nombre premier.

Dans le formalisme de Carnap, une telle phrase est considérée comme non signifiante parce qu'il y a une incohérence de catégories syntaxiques entre *César*, qui relève de la catégorie des personnes, et *nombre premier*, qui relève de la catégorie des nombres : aucun énoncé découlant de cette phrase ne pouvant être vérifié, il est nécessaire de la rejetter. Il en va de même pour la phrase en (2b) présenté en section précédente, puisque *siffle* et *soleil* sont aussi de catégories différentes. Et ceci contraste de façon étonnante avec les théories de l'École polonaise, puisque Ajdukiewicz considérait la même phrase (2b) comme possédant une signification cohérente, alors même qu'il reconnaissait l'existence d'une signification inhérente aux mots là où Carnap les considérait comme des signes vides ! Cette inversion paradoxale de traitement s'explique notamment par le fait que Carnap, contrairement à Ajdukiewicz, ne suivait pas la distinction husserlienne d'*Unsinn* et *Widersinn*, les deux étant confondus et considérés comme « agrammaticaux » par Carnap, alors qu'Ajdukiewicz considérait comme sensé tout ce qui évitait l'*Unsinn*.[35] Par conséquent, on observe que les catégories syntaxiques de Carnap s'avèrent finalement plus proches

cherchant à concilier son conventionnalisme (le rejet de référents pour les propositions) et son empiricisme (la nécessité de vérification des propositions synthétiques). La distinction entre propositions analytiques et synthétiques a été abondamment critiquée, notamment par Quine [146, 145] pour qui ces notions donnent lieu à des définitions circulaires, et n'ont donc pas lieu d'être. Pour davantage d'explications sur les idées de Quine, voir [81, 82, 169].

35. De fait, l'*Unsinn* correspond assez bien à ce qu'on appelle aujourd'hui l'agrammaticalité. Le statut du *Widersinn* est plus complexe, et possiblement plus proche des problèmes que nous cherchons à traiter à l'aide des types sémantiques. Carnap nommait le phénomène d'incohérence des catégories syntaxiques *Sphärenvermengung*, « confusion des sphères ». Nous reviendrons sur ces notions de *Widersinn* et de confusion des sphères par comparaison avec d'autres approches en section 2.5.

des considérations sémantiques et lexicales — et donc de nos types sémantiques actuels — que ne le sont les catégories sémantiques de l'École polonaise.

L'héritage des théories syntaxiques de Carnap s'est par la suite naturellement retrouvé dans les travaux sur les grammaires génératives initiés par Chomsky vingt ans plus tard. Cependant, ce qui nous intéresse ici ne représente qu'un petit fragment des théories générativistes qui, par sa relative confidentialité, est possiblement bien plus méconnu que les notions fondamentales des structures syntaxiques que nous utilisons aujourd'hui : il s'agit de l'idée de *degré de grammaticalité*. Chomsky [38, §IV] affirme en effet la nécessité de distinguer des degrés de conformité au langage, comme un système qui permettrait de mesurer une « distance » entre des énoncés et le langage correct, et rendrait en particulier compte de la capacité des locuteurs à ordonner des phrases en fonction de leur proximité avec un tel langage. Pour illustrer cette notion, Chomsky donne les énoncés suivants :

(4) a. Look at the cross-eyed elephant.
b. ?Look at the cross-eyed kindness.
c. *Look at the cross-eyed from.

Selon lui, chacune de ces phrases est plus « proche » — c.-à-d. plus correcte — au regard du langage que les phrases qui la suivent, ce que nous marquons de façon concrète ici avec les symboles qui les précèdent : un astérisque pour marquer l'agrammaticalité, et un point d'interrogation pour marquer une anomalie sémantique. [36] Cette distinction est aussi remarquée par Chomsky, qui déclare :

> We have already noted [...] that grammaticalness cannot be identified with meaningfulness or signifiance. Grammatical sentences may or may not be significant, and we must distinguish between grammatical nonsense and non-grammatical nonsense [...]. [38, p.115]

Cette déclaration de Chomsky rompt avec le point de vue de Carnap, qui ne faisait pas cette distinction et espérait pouvoir rejeter toutes les phrases non-signifinates du langage sur la base de sa théorie logique. [37]

Néanmoins, Chomsky espérait pouvoir lui aussi traiter un certain nombre de phénomènes à travers une approche syntaxique et catégorielle. Comme pour la théorie de Carnap, il s'appuie ainsi sur une notion de catégorie syntaxique pour construire ses degrés de grammaticalité. L'idée générale est la suivante : étant donné un certain degré de grammaticalité, il existe un ensemble de catégories syntaxiques associées aux mots du langage, de telle sorte que chaque mot ait au moins une catégorie syntaxique et que l'union de toutes les catégories restitue l'ensemble des mots du langage. Puis, on exige que le degré de grammaticalité suivant affine l'analyse du degré précédent en subdivisant les catégories déjà existantes. Le résultat forme alors une hiérarchie de catégories de plus en plus fines. Comme développé dans [40], au premier degré, ces catégories recoupent les parties traditionnelles du discours, distanguant les noms, les verbes, les adjectifs, et ainsi de suite. Puis, au second degré, elles subdivisent les verbes entre intransitifs, admettant des objets inaminés, etc. ; les noms entre abstraits, animés, inanimés, et ainsi de suite, chaque degré rajoutant encore des distinctions entre les groupes de mots. Et bien entendu, cette hiérarchie

36. Coïncidence curieuse, comparé à la première phrase qui est correcte, les deux exemples de Chomsky (4b) et (4c) manifestent respectivement les propriétés husserliennes de *Widersinn* et d'*Unsinn*. Cela souligne là encore que la problématique la plus intéressante du langage est posée par ces énoncés qui, bien que satisfaisant notre intuition des règles syntaxiques, produisent une signification que l'on pourrait qualifier de déviante.

37. Cette divergence de point de vue s'explique en partie par l'investissement de Carnap dans le programme philosophique du positivisme logique, dans la continuité de Frege et Russell, qui se donnait notamment pour but de concevoir un langage liberé de ses imperfections — et qui par conséquent ne peut pas générer d'énoncé qui ne serait pas interprétable [72].

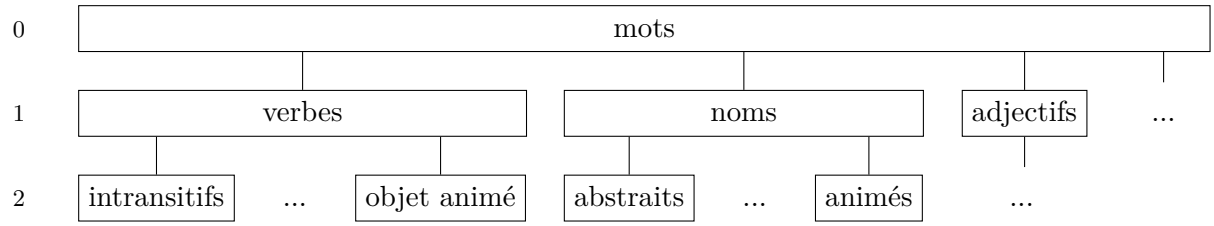


FIGURE 2.1 – Exemple des premiers degrés de la hiérarchie chomskyenne

admet un degré zéro constitué simplement de l'ensemble des mots du langage. Nous illustrons visuellement une telle hiérarchie en figure 2.1.

Une telle hiérarchie est supposée pouvoir être poussée jusqu'à de grands degrés de précisions : en particulier, elle permettrait a priori de reconnaître la phrase (3) ci-dessus comme syntaxiquement inacceptable, à l'instar du formalisme de Carnap. Mais jusqu'où peuvent aller ces degrés de grammaticalité ? Comme l'insinue Chomsky [38, 40], et comme le détaille par la suite Kiefer [97], les plus petites catégories syntaxiques regroupent supposément des mots qui sont substituables entre eux dans absolument tous les contextes : sous réserve de préservation des propriétés sémantiques et des valeurs de vérités, ceci renvoie à l'idée de synonymie. Cependant, contrairement à Chomsky, l'analyse de Kiefer admet l'existence d'une hiérarchie sémantique qu'il ne détaille pas, suggérant notamment que « *there seems to be no formal ways of distinguishing "grammatical" and "semantic" categories* » [97, p.VIII-1]. Cette approche sémantique lui permet alors d'étudier diverses relations sémantiques à travers les catégories, parmi lesquelles la similarité, le contraste et l'antinomie. Par conséquent, il apparaît que les catégories syntaxiques, quand bien même permettent-elles de rendre compte de certains degrés d'acceptabilité des phrases, ne sont possiblement pas suffisantes pour traiter l'acceptabilité sémantique de façon satisfaisante.

Plusieurs autres remarques viennent appuyer d'inadéquation des catégories syntaxiques à cet égard. Ainsi, Cohen [45], rassemblant diverses critiques envers l'idée de Chomsky, commente que dès lors qu'une phrase grammaticale est construite, il est toujours possible de tenter de lui imposer une interprétation en construisant un contexte qui exploite autant que faire se peut les propriétés grammaticales respectée par la phrase, quand bien même celle-ci serait a priori inacceptable sémantiquement. Ce procédé, rapporte-il, a été utilisé pour tenter de donner du sens au célèbre énoncé de Chomsky (5) destiné à illustrer l'archétype des phrases grammaticalement bien construites mais sémantiquement absurdes :

(5) Colorless green ideas sleep furiously.

Cette phrase a ainsi été intégrée dans divers poèmes et textes élaborés dans lesquelles elle a pu acquérir un certain degré de signification acceptable. Il s'ensuit que de trop hauts degrés de grammaticalité sont possiblement inadaptés à la tâche d'écarter formellement du langage certaines formes de phrases ou, comme suggéré par Chomsky [40] en réponse à ces critiques, qu'il soit nécessaire de distinguer les phrases nécessitant un travail de mise en contexte et d'exploitation pour leur donner du sens de celles qui n'en ont pas besoin. Mais, en dépit de la défense de Chomsky, il n'est pas certain que les degrés de grammaticalité soient la meilleure solution pour cette tâche.

Une autre critique, également rapportée par Cohen, est celle de Katz [93] qui argue que les degrés de grammaticalité s'accordent mal avec la notion d'*intelligibilité*, caractérisant la compréhension sémantique intuitive d'un énoncé. Plus précisément, Katz démontre qu'étant

donné un certain degré de grammaticalité i (ces degrés étant ordonnés comme au-dessus par finesse d'analyse croissante) il est aisé de produire des phrases satisfaisant au plus ce degré i et qui sont moins intelligibles que des phrases grammaticales à un degré j avec $j < i$. À titre d'illustration, Katz considère la phrase (6a) ci-dessus comme moins grammaticale au sens chomskyien, mais pourtant plus intelligible que la phrase (6b) :

(6) a. The beef cut sincerity.
b. If there is any truth in what he says, it would be too insist foolish.

Il subsiste cependant un flou sur ce qui est entendu par la notion d'intelligibilité, au-delà de l'aspect intuitif que l'on peut lui trouver : en particulier, elle est indépendante du sens et de la signification, et paraît posséder une composante cognitive sur laquelle peu de discussions sont faites. Mais quelle que soit la nature profonde de cette notion, la démonstration qu'en fait Katz pour sa critique de la hiérarchie des degrés de grammaticalité prouve surtout selon Cohen que cette dernière n'est pas un outil convenable pour la sémantique.

Il est néanmoins frappant d'observer que les approches strictement syntaxiques à la compositionnalité et à l'acceptabilité des phrases décrites dans ces pages aient pu résulter en des notions de catégories syntaxiques qui sont capables d'approximer de façon surprenamment précise des résultats d'ordre sémantique. De telles notions peuvent alors apparaître très proches des types sémantiques tels qu'on les utilise aujourd'hui, principalement du fait que les catégories syntaxiques de haut degré héritent des catégories de degré moindre, et donc dépendent directement des parties du discours, ce qui n'est pas sans évoquer l'homomorphisme de Montague ; et aussi parce qu'elles intègrent des modes de distinctions entre mots d'une même partie du discours selon des critères proches de la sémantique comme la distinction entre noms abstraits et concrets, animés et inaminés, etc. [38] Pourtant, les critiques exposées au paragraphe précédent semblent montrer que les catégories syntaxiques ne sont pas pleinement satisfaisantes pour rendre compte de toute la complexité sémantique inhérente à la langue naturelle. Comme défendu par Cohen [45], la notion de degré de grammaticalité est certainement utile jusqu'à un certain point, mais doit être restreinte à un usage strictement syntaxique : tenter de s'en servir pour des jugements d'acceptabilité sémantique serait une erreur. Il est donc nécessaire de construire un système qui soit distinct de la syntaxe, sans être forcément indépendante de cette dernière. Et une piste majeure, soulignée par Cohen dans sa critique de Katz, serait de se reposer davantage sur les propriétés lexicales des mots ; nous nous dirigeons donc naturellement vers cette nouvelle perspective dans la section suivante.

2.4 La sémantique lexicale structuraliste et au-delà

Alors que l'approche logique au langage faisait son chemin sous diverses formes comme nous l'avons vu précédemment, d'autres linguistes, héritiers de la pensée de von Humboldt ainsi que de de Saussure, et n'ayant a priori que peu de contacts à ma connaissance avec les logiciens précédemment cités, se sont également penchés sur le problème de la signification des mots et de la structure du langage, donnant naissance au courant de la sémantique lexicale structuraliste. Geeraerts [71] décrit l'émergence de cette pensée structuraliste au travers d'un article polémique du linguiste Leo Weisgerber à la fin des années 20, dans lequel ce dernier

38. Ces distinctions apparaissent généralement comme purement sémantiques dans les langues européennes, mais peuvent parfois être distingués grammaticalement dans certaines langues. Par exemple, certaines langues bantoues ainsi que certaines langues aborigènes d'Australie, comme l'anindilyakwa [191, 102], présentent des corrélations entre classes grammaticales de noms et considérations sémantiques sur les signifiés.

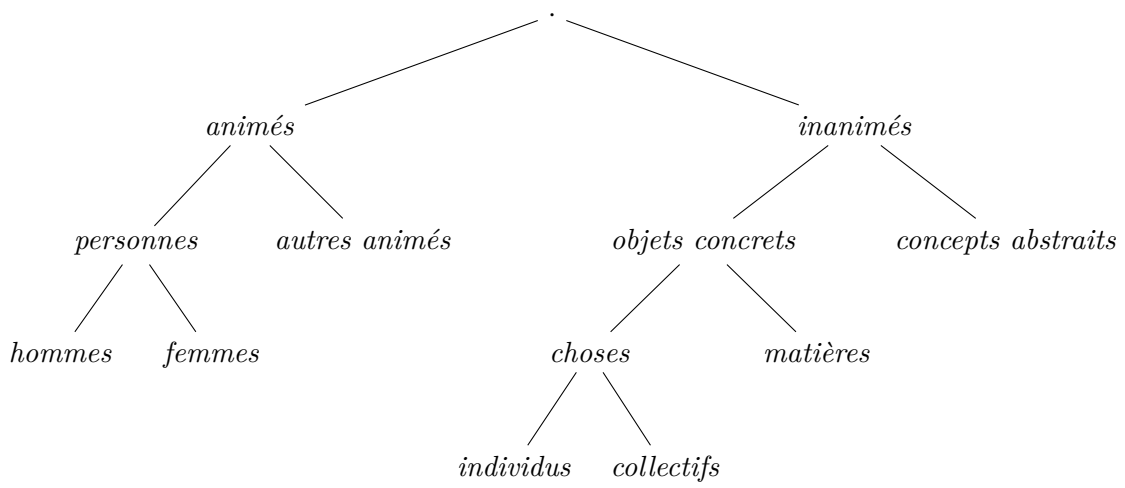


FIGURE 2.2 – Classification des noms d'après [3]

réfute la conception psychologique de la sémantique qui empêcherait de voir le langage comme un système symbolique, arguant que la vision psychologique qui dominait alors en sémantique avait pour conséquence que les significations étaient fruits de la psyché et non du langage lui-même. [39] Cette position a très vite fait école, et est restée l'approche dominante de la sémantique lexicale jusqu'au milieu des années 60.

L'idée principale des structuralistes est de considérer le langage comme un système dont il faut expliquer les règles, de façon comparable au système du jeu d'échec [40] : les mots ne sont pas considérés comme ayant des sens propres, mais comme ayant des sens dont la définition est conventionnelle, et par-là même indépendante de l'état psychologique de ceux qui les emploient, puisque la convention est la même pour tous. Cette simple idée conduit alors à voir le langage comme une couche conceptuelle entre l'esprit et le monde. [41] Les règles du langage que les structuralistes cherchent à retrouver sont alors celles qui régissent la combinatoire des mots, mais d'une façon qui dépasse l'aspect purement syntaxique : en effet, si le sens d'un mot est produit d'une certaine convention au sein du langage, alors étudier les manifestations de cette convention au travers des manières d'utiliser le mot renseignera sur son sens, permettant alors d'acquérir une information précieuse pour la sémantique lexicale.

En particulier, l'une des grandes propositions du structuralisme consiste à s'intéresser aux restrictions de certaines combinaisons de mots pour des raisons d'affinité sémantique. Ceci a notamment conduit à l'introduction de la notion de *collocation* par Firth [61], ainsi qu'à celle de *restriction de sélection* par Katz et Fodor [95]. Par ailleurs, d'une autre conception du structuralisme a conduit à l'hypothèse distributionnelle, c.-à-d. l'idée que le sens d'un mot est reflété par ses contextes d'utilisation — une idée fondamentale pour la linguistique de corpus ainsi que

39. Quand bien même la description de cette approche est réminiscente de celle d'Husserl, il convient de souligner que leur objet diffère, car la tradition saussurienne est ancrée dans l'étude des énoncés du langage tandis que Husserl s'intéressait davantage aux propositions logiques. De fait, les structuralistes du milieu du XX^{e} siècle ont massivement rejetté la pensée d'Husserl [76].

40. cf. [71, §2.1] pour une description complète de l'analogie, issue de la tradition saussurienne.

41. Les premiers structuralistes soutiennent d'ailleurs une vision très relativiste du langage, dans laquelle chaque langue impose sa structure conceptuelle sur le monde : en cela, ils étaient très proches de la position défendue par l'hypothèse Sapir-Whorf, selon laquelle le langage que l'on parle influence la manière dont on perçoit et pense le monde [106]. Notons néanmoins que cette hypothèse n'était pas nécessairement partagée par tous les structuralistes [71, §2.1.1].

pour la représentation vectorielle des mots exploitée en apprentissage automatique. L'une des premières applications concrète de cette hypothèse se retrouve chez Apresjan [3], qui étudie les structures au sein desquelles divers mots sont employés et les relations que ces structures peuvent avoir avec leurs sens, et établit pour cela une classification des noms sur la base des pronoms avec lesquels on peut les substituer. Un des résultats les plus intéressants de cette étude est l'observation que les différents sens d'un mot polysémique ont des distributions complémentaires au sein du langage : les classes définies par Apresjan pour les mots dans le contexte combinatoire du mot polysémique varient avec son sens, et des contextes où les classes sont similaires trahissent une similarité de sens pour le mot concerné. Il est frappant de constater que cette classification, récapitulée ci-dessus en figure 2.2, rappelle beaucoup celles des types sémantiques présentées en section 1.2.4. Il est cependant également difficile d'ignorer les points communs entre cette classification et la hiérarchie des degrés de grammaticalité de Chomsky introduite dans la section précédente.

Aux yeux de certains sémanticiens, il subsiste néanmoins un problème majeur avec cette approche structuraliste, qui consiste en la circularité de la définition des valeurs sémantiques. À moins de recourir à des classifications sur des critères extérieurs comme dans les travaux d'Apresjan, la méthode structuraliste implique que les sens des mots se définissent entre eux, sur la base des combinaisons possibles de chacun, ce qui pose le problème de devoir établir les sens des mots de chaque catégorie grammaticale en fonction des autres catégories. Pour échapper à cette circularité, il a été proposé d'introduire des valeurs particulières des sens des mots, exprimant certaines caractéristiques de leur comportement sémantique et ce indépendamment des valeurs des autres mots, permettant alors d'expliquer les comportements observés sans reposer sur des définitions interdépendantes. Cette approche a pris le nom d'*analyse par composantes*, et a notamment eu un impact majeur sur la sémantique lexicale lorsqu'elle s'est affranchie des idées structuralistes pour s'accorder au courant génératiViste, au travers des travaux de Katz et Fodor sur leur théorie sémantique [95]. Ceux-ci présentent leur théorie sémantique comme permettant d'expliquer le phénomène de *compétence* identifié par Chomsky, en prenant le relais de la grammaire syntaxique lorsque cette dernière n'était plus suffisante pour rendre compte des observations [42], arguant que deux phrases de même structure syntaxique pouvaient avoir des sens très différents. En particulier, ils exigeaient de leur théorie sémantique qu'elle soit capable d'énumérer toutes les ambiguïtés possibles d'une phrase, et proposaient de s'appuyer sur des marqueurs pour distinguer les différents sens d'un mot.

Nous trouvons important ici de souligner le parti pris des auteurs d'organiser leur théorie comme un modèle scientifique rigoureux : [43]

> The controls on a semantic theory of a natural language are nothing more than the usual empirical and methodological constraints imposed on any scientific theory : the requirement that a semantic theory match the fluent speaker's ability to interpret sentences is the particular form, in semantics, of the general methodological requirement that a theory accord with the facts. If certain consequences of a semantic theory conflict with the facts (the performance of fluent speakers), various revisions [...] must be tried out and compared to determine which solution best accommodates the linguistic evidence. [95, p.193]

Cela ne signifie pas bien sûr que les théories précédemment exposées dans ses pages sont dépourvues de rigeur et d'intérêt au regard de la sémantique, mais l'accent mis sur le statut de

42. Rappelons que, comme vu dans la section précédente, Katz se montrait critique à l'égard des degrés de grammaticalité de Chomsky. Même si certaines de ses critiques étaient possiblement mal fondées, comme pour l'intelligibilité, il est généralement reconnu — et nous le rejoignons ici — qu'il avait globalement raison sur l'incapacité de la théorie syntaxique à rendre compte de toutes les subtilités sémantiques de la langue.

43. Cette même approche sera réutilisée par Katz pour constituer sa théorie du sens [94] ; voir aussi le chapitre suivant.

la théorie comme modèle permet de rappeler que l'introduction de certains outils, que ce soient les marqueurs, les catégories ou les types sémantiques, ne sont jamais que des valeurs abstraites qui ont pour objectif d'approximer au mieux le fonctionnement observable de la langue. Dans le cas qui nous intéresse, la définition d'une hiérarchie de types sémantiques ne peut donc être fondée que sur des bases empiriques. Nous reviendrons sur les implications de cet enoncé dans le chapitre 7.

Néanmoins, les marqueurs sémantiques introduits par Katz et Fodor, permettant de classer les différents sens d'un mot selon un arbre hiérarchique — de sorte que chaque sens corresponde à une série de marqueurs ordonnés par précision croissante — ne correspondent pas tout à fait à ce que l'on exige des types sémantiques, principalement parce que certaines distinctions sont trop précises : dans le cas célèbre du mot *bachelor* par exemple, il ne semble pas nécessaire que la distinction de sens entre « célibataire », « diplômé » ou encore « jeune chevalier » soit capturée au niveau des types sémantiques, mais encore en aval dans l'analyse sémantique. Cette impression, assez intuitive à ce stade de notre dscussion, est par ailleurs renforcée par le fait que les exemples de restrictions de sélection donnés par Katz et Fodor portent systématiquement sur des marqueurs placés assez haut dans les hiérarchies de sens, laissant donc en suspens la question de la pertinence des marqueurs plus précis au regard de la compositionnalité. Notons enfin qu'ils reconnaissent des similarités entre les catégories chomskyennes et leurs marqueurs, tout en soulignant l'importance de distinguer ces deux constructions.

Cette approche n'est cependant pas exempte de critiques, dont nous restituons les grandes lignes ici (cf. [71, §3.2] pour davantage de détails). Tout d'abord, il faut relever que, de la même manière que les hiérarchies de types discutées en section 1.2.4 ont été constituées, le choix des marqueurs sémantiques pertinents est en grande partie arbitraire, en dépit des impératifs rigoristes des deux auteurs. De plus, si l'introduction de certains marqueurs semble avoir pour but de résoudre certains problèmes sémantiques isolés, il n'est a priori pas possible de résoudre tous les problèmes uniquement à l'aide de marqueurs de cette forme. En particulier, les postulats de significations proposés par Carnap [30] sont à même de capturer l'emploi des marqueurs sémantiques, mais sont également capables d'exprimer des propriétés inaccessibles à ces derniers, rendant de fait la sémantique logique plus puissante pour l'analyse sémantique qu'une approche uniquement basée sur ces marqueurs.

Que retenir alors du développement des sémantiques lexicales structuraliste et génératiiviste en ce qui concerne nos types sémantiques ? L'une des idées majeures mises en lumière par ces approches réside dans la conception qu'il existerait une partie de la signification qui serait purement linguistique, dans la mesure où elle serait indépendante du contexte pragmatique d'énonciation d'une part, et indépendante de la connaissance encyclopédique du monde d'autre part. Cette partie interne de la signification, généralement appelée *sens*, est traitée de façon essentialiste puisqu'elle réduit le langage à un minimum de composantes primitives qui permettraient de rendre compte de certaines anomalies sémantiques sur la simple observation de compatibilité entre les composantes associées à deux mots combinés dans l'énoncé, un procédé qui n'est évidemment pas sans rappeler celui des types sémantiques modernes. Au cours du développement de ces théories, plusieurs formalisations de cette idée ont été developpées, mais toutes se heurtent à la difficulté de définir quels éléments appartiennent à la connaissance linguistique interne et lesquels appartiennent à la connaissance encyclopédique du monde. Par exemple, y a-t-il ou non une distinction linguistique interne entre les mots *chaise* et *fauteuil*, alors que la frontière entre les objets dénotés par ces mots est floue ? Lyons [112] avance comme solution à ce type de problème qu'il faut s'abstraire des référents désignés par les mots concernés ; mais cela ne résout pas non plus tous les problèmes puisque la théorie du sens qu'il défend comprend l'étude de diverses relations de sens (antonymie, synonymie, méronymie) pour lesquels la démarcation entre

connaissances linguistiques et encyclopédiques n'est pas vraiment plus nette. Par ailleurs, il n'est pas toujours possible d'ignorer le référent, dans la mesure où celui-ci peut contraindre un mot nouvellement apparu pour le désigner, posant un problème de traitement pour les néologismes.

La question des types sémantiques est évidemment liée à la question du sens, les travaux de Katz et Fodor étant globalement reconnus comme le point de départ de l'introduction d'élements lexicaux dans les modèles d'analyse sémantique, ayant notamment conduit aux approches modernes dites « néo-structuralistes » au sein desquelles se retrouve la théorie GL de Pustejovsky [71, §4.1.4]. Par conséquent, tous les problèmes énoncés au paragraphe précédent se posent également pour les types sémantiques, qui semblent devoir se détacher au maximum de toutes les considérations de référent et de relation de sens pour pouvoir exprimer au mieux leur intérêt — et leur différence quant aux autres formes d'analyse sémantique. La caractérisation des types sémantiques passe donc par ce qui est linguistiquement observable, et en particulier les combinaisons effectives des mots de la langue. Ceci rejoint l'idée a priori indépendante de la signification par l'usage définie par le Wittgenstein tardif [188], qui entre partiellement en résonance avec les idées structuralistes présentées ici. Dans la section suivante, nous allons voir comment la conception russellienne des types, la tradition catégorielle d'Aristote et cette vision de la signification par l'usage ont pu être conciliées en une théorie puissante qui retiendra notre attention pour la suite de ce texte, posant les fondement d'une théorie des types sémantiques qui devrait pouvoir répondre à nos attentes.

2.5 L'erreur de catégorie

Nous avions déjà evoqué en section 2.2 les catégories d'Aristote, qui avaient en partie inspiré Husserl pour l'élaboration de ses catégories sémantiques. Comme expliqué dans [179], ces catégories ont été proposées par Aristote comme caractérisant les choses dont il est possible de parler par leurs divisions nécessaires dans notre champ conceptuel, de sorte que deux éléments appartenant à des catégories distinctes ne puissent pas admettre de termes descriptifs communs. La tradition retient d'Aristote une liste de dix catégories majeures : substance, qualité, quantité, relation, lieu, temps, posture, état, action et passion. Cette classification n'est bien sûr pas restée définitive, et de nombreux philosophes se sont penchés sur la question, proposant parfois des méthodologies pour tenter de redéfinir ces catégories à l'instar de Kant. Dans la première moitié du XX^{e} siècle, Husserl n'a pas été le seul à s'intéresser à ces catégories et à en proposer une reformalisation : dans cette section, nous nous intéressons à une succession de travaux philosophiques qui s'inspirent également de cette notion aristotélicienne, à commencer par ceux de Gilbert Ryle.

Dans [157], Ryle n'hésite pas à reconnaître la proximité conceptuelle entre ces catégories et les types logiques de Russell, quitte à les assimilier l'un à l'autre. Il remarque également que certaines idées simples, comme formulées par Aristote, permettent de vérifier que de telles catégories existent effectivement et que nous en avons une certaine conscience : par exemple, les mots interrogatifs qui remplacent les expressions lors de la formation de questions changent en fonction du type d'information donné par ces expressions, permettant de fait d'obtenir naturellement une première source de classification. Bien évidemment, la classification ainsi obtenue n'est pas suffisante pour rendre compte de toutes les subtilités sémantiques du langage, et Ryle se demande ainsi comment généraliser cette approche pour obtenir des catégories satisfaisantes.

Son idée est réminiscente de plusieurs autres constructions catégorielles présentées dans les pages précédentes, se basant sur des notions de substitution et de composition pour répartir les expressions du langage dans différentes catégories. Ryle considère ainsi des expressions complètes

mais aussi des expressions incomplètes, ou « à trous », qu'il regroupe sous le nom de *facteurs*, et observe quels facteurs peuvent occuper la place disponible dans un autre facteur donné. Cependant, sa notion d'acceptabilité pour l'occupation d'une telle position n'est pas seulement subordonnée à la grammaticalité comme pour les catégories sémantiques, mais inclut des considérations sémantiques qui se regroupent sous la notion-clé d'*absurdité*, pour désigner des énoncés qui sont sémantiquement problématiques parce qu'ils comportent des combinaisons dont il est difficile d'appréhender le sens. L'exemple iconique du travail de Ryle est donné en (7) ci-dessous : bien qu'on puisse reconnaître que cette phrase est grammaticalement correcte, elle présente une anomalie sémantique qui tient au fait qu'un jour de la semaine, une notion temporelle, ne peut pas avoir la position spatiale nécessaire à l'occupation d'un lit.

(7) Saturday is in bed.

Pour Ryle, ce genre d'exemple permet de distinguer les types grammaticaux, qui correspondent aux catégories syntaxiques, et les types logiques. Il définit alors une notion de type (ou type logique, ou catégorie, tous ces mots étant substituables ici) qui assure que la combinaison de deux facteurs n'est pas absurde tant que la correspondance des types est respectée.

Cette nouvelle notion rejoint naturellement celle d'Aristote et de sa remarque sur les mots interrogatifs puisque ces derniers peuvent être vus comme des mots de remplissage qui marquent grammaticalement l'existence de trous dans les facteurs correspondants : ces mots s'adaptent alors au facteur qu'ils remplacent et changent de forme en fonction du type logique attendu. Mais par son traitement formel de la notion, Ryle généralise les types bien au-delà des catégories définies par Aristote. Le non-sens ou l'absurdité au sein d'une phrase grammaticale et usant d'un vocabulaire conventionnel est alors l'indice qu'un élément de cette phrase n'est pas du type attendu pour être correctement couplé aux autres, un phénomène que Ryle appelle l'*erreur de catégorie*, et qui se rapproche de la notion de discordance de types utilisée dans notre acception moderne des théories des types (cf. section 1.4). En revanche, il convient de souligner que Ryle n'est pas particulièrement précis sur sa définition de l'absurdité, et sa compréhension de la notion englobe aussi certaines formes de paradoxes, tels qu'illutrés par les phrases suivantes — lesquelles ne sont pas sans rappeler celles qui ont inspiré à Russell sa théorie des types :

(8) a. "Heterological" is heterological.
b. I am now lying.

Le typage d'une expression X dans cette approche est alors obtenu en déterminant dans quelles propositions non-absurdes X peut figurer, et à quelle position dans celles-ci, plaçant ainsi cette notion d'absurdité au centre de la construction des types. Il convient d'ailleurs de souligner que l'absurdité utilisée ici est distincte de la notion de fausseté : une proposition absurde n'est ni vraie ni fausse, puisqu'elle positionne sur un même plan deux éléments qui sont fondamentalement incompatibles et de la combinaison desquels on ne peut rien décider ; et à l'inverse, une proposition fausse est nécessairement non-absurde, puisque nous avons été en mesure d'évaluer la combinaison de ses éléments et de la confronter au réel pour déterminer sa fausseté. Cette distinction est essentielle pour comprendre la portée de cette notion de type. Ajoutons de plus que, comme le reconnaissait Ryle, les types ainsi définis ne renseignent pas vraiment sur la « nature » des choses désignées par les expressions, mais seulement sur les combinaisons qu'il est possible de réaliser avec, ce qui offre donc une précision bien plus faible.

Une contribution importante au développement des idées de Ryle autour de l'erreur de catégorie et de l'absurdité est incarnée par les travaux de Fred Sommers [170, 171, 172, 173, 174]. Ce dernier, influencé d'une part par Russell et les bases de sa théorie des types, et d'autre part

Niveau	Objet	Propriété correcte	Propriété incorrecte
0	syntaxe	grammaticalité	agrammaticalité
1	types	« bon typage »	erreur de catégorie
2	logique	cohérence	contradiction
3	vérité	véracité	fausseté

TABLE 2.1 – Descriptions des niveaux d'analyse proposés par Sommers

par la théorie de la signification par l'usage de Wittgenstein (à laquelle Ryle se rapproche quelque peu, cf. [170]), s'interroge sur la notion de signification et comment la construire. Il remarque notamment que connaître la signification d'un mot revient à savoir comment l'utiliser, ce qui inclut de savoir comment l'inclure dans la formation de phrases non-absurdes. Mais cela ne recouvre pas toute l'étendue de la signification : en effet, des mots antonymes comme *short* et *tall* ont exactement les mêmes usages, mais des significations différentes. Cette connaissance sur l'emploi ne fournit qu'une partie de la signification que Sommers, à l'instar des structuralistes, appelle le *sens*. Le sens se réduit donc à connaître la position d'une expression sur une carte de relations de sens, qui décrit la capacité de ces expressions à se combiner de façon non-absurde, et l'étude de cette relation de sens entre expressions est un fondement de l'établissement de types logiques dans la continuité de Ryle.

La distinction entre signification et sens est de première importance pour Sommers, car elle lui permet de réfuter l'idée de Ryle selon laquelle les phrases en (8) ci-dessus sont absurdes : pour Sommers, les combinaisons de mots dans ces différentes phrases ont bel et bien du sens, et ne sont donc pas des cas d'erreur de catégorie. En revanche, les paradoxes dont elles font preuve font qu'elles échouent à produire une signification. Ce faisant, Sommers sépare l'étude du sens et l'étude de la signification en deux étapes distinctes au sein de l'analyse syntaxico-sémantique. Dans [171, §III], il défend d'ailleurs la vision de *niveaux de rectitude* pour les énoncés du langage, chacune associée à sa méthode d'analyse et ayant des objets distincts. Si cette idée ressemble conceptuellement à celle des degrés de grammaticalité (cf. section 2.3), elle s'en écarte largement par le fait que le recours à des notions de catégories n'est pas systématique, mais dépend des méthodes nécessaires à chaque niveau. Dans cette approche, chaque phrase peut être reconnue comme correcte ou incorrecte à un niveau donné, mais la possibilité de décider de cette correction est subordonnée à la reconnaissance de la correction à tous les niveaux qui précèdent. Le détail des niveaux tels que présentés par Sommers, numérotés par ordre croissant, sont résumés en table 2.1. On voit ainsi que l'étude des erreurs de catégories éventuelles présuppose une phrase bien constituée grammaticalement, et que la vérification de cohérence logique présume du respect des catégories : c'est grâce à cela que Sommers distingue les phrases incohérentes en (8) des erreurs de catégories. Il utilise par ailleurs ce schéma pour proposer une méthode mentale de détection des ambiguïtés [171, p.349].

Si cette hiérarchisation des niveaux d'analyse est acceptable, alors la présente thèse s'interesse exactement au niveau 1, qui s'incarne par l'utilisation des types sémantiques pour les mêmes fins que les catégories de Ryle et Sommers : permettre de décider quelles combinaisons de mots posent un problème sémantique par leur forme intrinsèque, indépendamment des contradictions logiques et encyclopédiques qui y seraient liées. De plus, les catégories en question présentent de nombreuses similitudes avec les types sémantiques tels que nous les souhaitons, étant basées sur des notions de compositionnalité très proches de ce que nous observons chez Montague et ses prédecesseurs, dont la relation entre prédicats et arguments. La théorie catégorielle de Sommers,

ou *ontologie*[44], est justement construite sur cette idée de relation entre prédicats et arguments, à laquelle on se réfèrera désormais comme *relation de prédication*.

Plus précisément, la relation de prédication est valable lorsque la combinaison d'un prédicat et d'un argument a du sens, c.-à-d. ne présente pas d'absurdité. Cette relation s'étend alors en une relation de sens entre les expressions de la langue en fonction des prédications qu'elles peuvent mener conjointement. Sommers définit alors plusieurs formes de types comme des ensembles d'expressions prédicats et arguments, en reliant notamment une forme aux catégories de Ryle et une autre à la définition suivante donnée par Russell :

> The definition of a logical type is as follows : A and B are of the same logical type if, and only if, given any fact of which A is a constituent, there is a corresponding fact which has B as a constituent, which either results by substituting B for A or is the negation of what so results. To take an illustration, Socrates and Aristotle are of the same type, because "Socrates was a philosopher" and "Aristotle was a philosopher" are both facts ; Socrates and Caligula are of the same type, because "Socrates was a philosopher" and "Caligula was not a philosopher" are both facts. To love and to kill are of the same type, because "Plato loved Socrates" and "Plato did not kill Socrates" are both facts. [156]

Des formes supplémentaires sont associées à ces formes préliminaires afin de constituer une définition formelle des catégories. La notion-clé de cette construction est la notion de *portée* d'un prédicat, qui est l'ensemble des arguments que le prédicat accepte de sorte que la combination ne soit pas absurde : les catégories des prédicats sont alors formés comme les classes d'équivalences des prédicats ayant même portée, et les catégories d'arguments s'en déduisent aisément.

La force de la théorie ontologique de Sommers est alors de montrer que ces catégories ont une structure hiérarchique en forme d'arbre, à l'aide de lois formelles et logiquement prouvables. De plus, il asserte une identité entre cette structure ontologique et la structure du langage, permettant de faire le lien entre ce que le langage exprime et les faits observables, fournissant ainsi un équilibre satisfaisant entre le sens et les attributs possibles pour les référents concernés. En outre, la théorie de Sommers admet l'existence d'expressions chevauchant plusieurs catégories incompatibles, sous le nom de *composés hétérotypiques*[45], permettant de répondre par exemple au problème cartésien de la dualité corps-esprit [172, 174]. Un exemple plus concret d'hétérotypie est aussi donné dans [174] par le mot *country*. À travers les phrases en (9), Sommers montre que ce mot est capable d'apparaître dans la portée de deux prédicats qui sont autrement incompatibles, et qu'il est donc un tel composé. La ressemblance entre ces composés et la notion d'objets pointés en GL [135] (voir section 1.2.2) est bien entendu frappante, ce qui renforce encore l'intérêt de cette théorie ontologique pour développer la notion de type sémantique dont nous avons besoin.

(9) a. The peninsula is sunny.
b. The society is democratic.
c. ?The society is sunny.
d. ?The peninsula is democratic.
e. The country is sunny and democratic.

44. L'ontologie, étymologiquement la « science de ce qui est », est présentée par Sommers comme « *the science of categories* » [171, §4]. Ce terme est principalement utilisé dans son acception philosophique pour la présente thèse, mais comprend des similitudes évidentes avec l'acception informatique du terme *ontologie* désignant une structure formelle de connaissance très utilisée en intelligence artificielle, entre autres [78]. Il n'est d'ailleurs pas exclu que l'ontologie philosophique des catégories du langage puisse être exprimée sous la forme d'une ontologie formelle, ayant pour individus les mots du langage rassemblés en catégories, voire en types sémantiques.

45. Nous poserons dans la suite la convention suivante : *composé hétérotypique* désignera une expression associée à plusieurs catégories ; on appellera *hétérotype* la construction du système de type qui permet de modéliser ce chevauchement, et le phénomène associé sera nommé *hétérotypie*.

Cependant, cette théorie suppose d'avoir une évaluation correcte des énoncés absurdes. Comme l'illustre le traitement divergent des paradoxes par Ryle et Sommers, il n'est pas toujours évident de déterminer le genre d'anomalie sémantique en jeu dans une phrase donnée et, comme l'ont remarqué d'autres philosophes, la notion sommersienne de portée et celle de *prédicabilité* (ou capacité à décider si l'application d'un prédicat à son argument donne une proposition vraie ou fausse), qui sont confondues par Sommers [173], sont en fait distinctes [163, 54] : en effet, certaines phrases sans erreur de catégorie et non contradictoires peuvent se révéler *vides*, c.-à-d. impossibles à évaluer comme vraies ou fausses, pour des raisons contextuelles ou pragmatiques indépendantes. Englebretsen [54] fournit une courte liste de tels énoncés, incluant les conditionnelles dont les antécédents sont faux, les énoncés dont l'ambiguïté empêche la décision quant à leur vérité (10a) et les énoncés dont les présuppositions sont fausses, que ce soit par l'impossibilité de produire un référent correct (10b), ou pour des raison de temporalité (10c).

(10) a. The house is taller than the hill.
b. The present king of France is bald.
c. Socrates was in awe of Moses.

Ainsi, la vérité de la première phrase dépend du processus d'évaluation utilisé pour la mesure : s'il s'agit de considérer la distance entre un niveau fixe (p. ex. le niveau de la mer) et les sommets de la maison et de la colline, la maison située sur la colline sera nécessairement plus haute ; mais si l'on considère la distance entre la base et le sommet de chaque entité séparément, la colline sera plus grande. Mais cela n'empêche pas *house* de bien figurer dans la portée de *be taller than the hill*. Pour la deuxième phrase, le problème tient évidemment au fait qu'il n'existe pas actuellement de roi de France, et qu'il n'y a donc pas de référent sur lequel évaluer la vérité de la proposition. Enfin, la troisième phrase présuppose que Socrate aurait connu l'existence de Moïse, ce qui est bien sûr impossible : de ce fait, il est impossible d'affirmer quoi que ce soit sur le sentiment du premier à l'égard du second. Notons néanmoins que l'aspect contextuel de l'étude de ces énoncés placerait l'analyse de la vacuité comme un niveau intermédiaire entre les niveaux 2 et 3 dans la table 2.1.

Un autre aspect important de la théorie de Sommers, probablement le plus contre-intuitif, est la relation de cette théorie avec la négation. En effet, la valeur négative d'une prédication s'accorde généralement mieux avec notre intuition dans les cas où une prédication positive donnerait une erreur de catégorie ; de sorte par exemple que les phrases en (11) sont communément perçues comme vraies :

(11) a. Les chiens ne miaulent pas.
b. Les tables ne miaulent pas.

Pourtant, la théorie sommersienne nous pousse à admettre que ces deux phrases sont différentes, et illustrent deux phénomènes négatifs distincts : le *démenti* et la *négation*. Cette distinction a trait à la relation de sens sous-tendues dans ces exemples. Ainsi, la version positive de (11a) ne contient pas d'erreurs de catégorie : les chiens, quand bien même il est universellement admis qu'ils ne miaulent pas, sont des animaux au même titre que les chats et les deux partagent de nombreux prédicats communs ; en particulier, on peut leur attribuer de manière non-absurde diverses formes de vocalisation, y compris les miaulements. De ce fait, la phrase (11a) affirme la négation du prédicat *miauler* sur *chien*, qui est une prédication correcte : il s'agit d'un *démenti*. À l'inverse, les tables sont incapables de vocalisation, et la conception même de cette possibilité est absurde, de sorte que la version positive de (11b) serait une erreur de catégorie ; elle nous apparaît vraie uniquement parce que nous l'interprétons comme l'affirmation que les tables

n'appartiennent pas à la catégorie des choses qui peuvent miauler ou non : par l'existence de l'erreur de catégorie sous-jacente, il s'agit d'une *négation*.

La reconnaissance de deux formes différentes de négations n'est pas limitée à la théorie de Sommers, et apparaît sous des dénominations diverses au sein des travaux de nombreux autres philosophes [87]. D'un point de vue logique, la distinction qui est faite ici est de séparer la négation des propositions d'une part, et la négation des prédicats d'autre part [173, 174]. La négation d'un prédicat est aussi un prédicat, qui a dans sa portée les mêmes arguments que le prédicat originel, avec simplement une inversion des valeurs de vérité renvoyées ; par conséquent, si l'application d'un prédicat à un argument est une erreur de catégorie, il est attendu que l'application de sa négation à ce même argument le soit également. Il est à noter que des paires de prédicats opposés existent déjà dans les langues, parfois marquées par un simple ajout morphologique (*réel–irréel*, *capable–incapable*), ou parfois issues de deux racines différentes (*allumé–éteint*) ; cependant, de nombreux prédicats n'ont pas de tels opposés disponibles, et se voient contraints de construire leur négation avec le même marqueur utilisé pour la négation des propositions (*ne ... pas* en français, *not* en anglais), d'où la confusion qui peut résulter de cas comme en (11). La compréhension des mécanismes en jeu dans la distinction entre démenti et négation est fondamentale pour la théorie catégorielle de Sommers dans la mesure où elle contribue à la construction des catégories elles-mêmes : une négation prise pour un démenti pourrait en effet résulter en une analyse erronée.

La théorie des catégories de Ryle et de Sommers présente ainsi un fort intérêt pour notre propre construction des types sémantiques, car elle fournit des méthodes formelles de définition des catégories et de l'ontologie associée, la méta-structure des niveaux d'analyse qui rejoint l'impératif de modularité de l'analyse syntaxico-sémantique que nous avons défendu en section 1.3, ainsi qu'une approche compositionnelle qui répond au besoin de discerner les combinaisons de mots sémantiquement problématiques et rend compte de la notion d'objet pointé qui est aujourd'hui essentielle en sémantique lexicale. Assez curieusement, malgré le soutien de quelques philosophes au travers des décennies qui ont suivi [162, 55, 56], la théorie de Sommers ne semble pas avoir reçu la même reconnaissance que les travaux abordés dans les sections précédentes, et son influence sur les théories de types sémantiques actuelles est somme toute assez mince, bien qu'il subsiste encore des travaux qui arguent en faveur d'une intégration de cette théorie dans les systèmes modernes [158, 159]. Tout comme ceux-là, nous voyons dans cette théorie un potentiel certain pour nos objectifs, et quand bien même cette théorie seule ne saurait répondre à toutes nos attentes, son articulation avec d'autres idées explorées dans les sections précédentes pourrait bien nous fournir le matériau nécessaire pour répondre à ces exigences.

Bilan. Dans ce chapitre, nous avons passé en revue de nombreux formalismes qui peuvent être perçus comme des influences ou des précurseurs des types sémantiques tels qu'utilisés dans nos théories actuelles. Le socle commun à toutes ces théories est la reconnaissance du langage comme un système compositionnel où des expressions peuvent apparaître incomplètes et nécessiter leur combinaison avec d'autres expressions pour acquérir leur signification finale. Sur la base de cette simple idée, initiée par Frege, sont venues se construire diverses théories destinées à imposer des règles sur ce phénomène de composition, de telle sorte qu'un non-respect des règles en question se traduise par l'obtention d'une phrase qui soit reconnue comme incorrecte ou anormale selon des critères liés à la compréhension de la signification des phrases. Les sections précédentes ont ainsi été l'occasion de voir que le choix de ces critères influence fortement la portée de ces règles et la forme d'anormalité qu'elles capturent, et que ces critères, en dépit des divergences d'approches

philosophiques sur la notion de signification dont elles dépendent, tendent à se ressembler d'une théorie à l'autre.

Il est même possible de hiérarchiser ces critères en fonction de leur capacité à capturer les anomalies sémantiques, à la manière de l'échelle des niveaux d'analyse élaborée par Sommers (table 2.1). Le premier niveau est celui de la syntaxe, principalement illustré par les catégories d'Husserl, de l'École polonaise, ainsi que de Chomsky, où le critère de rejet se base sur notre capacité à assembler les mots d'une phrase de façon cohérente pour déterminer sa structure même, indépendamment de la signification des expressions utilisées. À l'opposé, le dernier niveau est celui de la sémantique logique voire pragmatique, qui se base sur l'utilisation de la connaissance lexicale voire contextuelle pour déterminer si les combinaisons de mots comportent ou non des contradictions logiques : à ce niveau se rangent les théories de Katz et Fodor ainsi que les postulats de signification de Carnap. La théorie des types initiée par Russell, qui vise principalement à contrer les paradoxes logiques, se situe vraisemblablement à un niveau un peu moins élevé, puisque la reconnaissance de tels paradoxes exploite a minima la structure logique de la propostion formée par la phrase, et au plus des caractères superficiels de la connaissance lexicale. Cependant, la théorie des catégories ontologiques de Sommers semble cerner le mieux un niveau intermédiaire entre la syntaxe et la sémantique, où les critères de rejet des phrases se base sur des propriétés lexicales minimalistes, sans exploitation de relations logiques, et de manière plus fine que ne le font les parties du discours ou autres catégories syntaxiques ; l'approche préconisée par la sémantique structuraliste est également très proche de ce niveau, de même que l'intuition des *noun-classes* de Turing en théorie des types.

S'il y a bien un niveau qui correspond le mieux à nos exigences sur le rôle des types sémantiques au sein de l'analyse du langage, ce serait donc ce niveau intermédiaire, capable de contraindre la compositionnalité sans pour autant devoir creuser dans le détail lexicographique des significations des mots. Cette impression est d'ailleurs renforcée par la similarité frappante entre la théorie sommersienne et plusieurs concepts qui sont aujourd'hui clairement admis dans les théories de types sémantiques, comme l'existence d'une hiérarchie de types comportant une notion de sous-typage et l'existence des composés hétérotypiques qui traversent cette hiérarchie. Il nous faut cependant souligner l'importance aujourd'hui encore très présente de la théorie des types simples, qui a la vertu de fournir des indications sur les arguments nécessaires à la complétion du sens d'une expression, une dimension qui est globalement ignorée du programme de Sommers. On peut alors dresser de ce constat l'ébauche de ce que serait une théorie des types idéale, conciliant la théorie des types et les catégories ontologiques en évitant d'accorder trop d'importance à la logique et aux connaissances encyclopédiques. Dans le prochain chapitre, nous allons donc naturellement tenter de formaliser cette ébauche en une théorie structurée et complète des types sémantiques.

Chapitre 3

Vers une formalisation des principes linguistiques des types

Grizzly bears exist within the set of all real numbers, and are not prime. The square root of a grizzly bear is prime, however, and is the only prime number that a) is not a cardinal number, b) is neither even nor odd, and c) contains an animal component.

SCP-1313

Le chapitre précédent a été l'occasion de dresser un tableau d'ensemble des problèmes, des concepts et des formalismes linguistiques et philosophiques que l'on pouvait relier à l'approche moderne des types sémantiques et, comme nous avons pu le constater, les éléments à prendre en compte dans ce tableau sont nombreux. L'objectif de ce nouveau chapitre est de tenter de dégager de ces observations des principes fondamentaux permettant d'encadrer et de diriger la construction concrète d'une théorie des types sémantiques. Comme annoncé au chapitre 1, il s'agit avant tout d'apporter une réponse aux questions (Q2) et (Q3) de notre problématique en explicitant l'objet véritable des types sémantiques, à savoir leur domaine d'application ainsi que leur portée au regard de la signification, et en déterminant les règles générales de leur comportement, afin de permettre la construction de formalismes qui respectent l'intégralité de ces règles.

Cet objectif dérive du constat qu'il est difficile, voire impossible, de donner une *définition* définitive des types sémantiques : leur objet est trop imprécis, trop variable d'un esprit à l'autre, pour être cerné de manière satisfaisante, et tenter d'approfondir cette direction ne ferait que reproduire les variations qui ont pu apparaître autour de la notion de « signification », par exemple. À la place, nous souhaitons essayer d'approximer la notion de type sémantique à la manière d'un modèle, selon la méthode scientifique défendue notamment par Katz [95, 94] qui s'inspire notamment de la physique : il s'agirait alors de traiter les types sémantiques comme un artéfact hypothétique permettant d'expliquer certaines formes d'anomalies sémantiques, et de déduire de ce postulat et des observations empiriques les règles qui régissent son fonctionnement, à la manière par exemple du postulat pour le boson BEH en physique des particules.

Les types sémantiques sont effets des outils destinés à expliquer un ensemble de phénomènes particuliers, liés à la reconnaissances de diverses formes d'anomalies sémantiques ; ces phénomènes, comme on a pu le voir à travers les discussions de la section 1.2, incluent l'incompatibilité sémantique de certains mots entre eux, la coprédication, les coercions et les usages

créatifs et stylistiques de la langue. Il nous revient donc la tâche de déterminer *comment* ces types peuvent rendre compte des différentes observations linguistiques qui ont déjà illustré les deux premiers chapitres de cette thèse, et ce que nos conclusions à cet égard nous apprennent sur leurs formalisations mathématiques potentielles. Nous commençons pour cela par tenter de séparer explicitement les types sémantiques de tout ce qui ne les concerne pas, ou alors de très loin, parmi les notions majeures de la sémantique, afin d'en faire des « particules » aussi élémentaires que possible : c'est l'objet de la section 3.1. Ensuite, nous formalisons en section 3.2 les idées de cette pré-analyse afin de définir le cadre d'une théorie des types sémantiques, en explicitant ses principaux axiomes fondateurs. Enfin, nous complétons cette théorie au sein de la section 3.3 par la formulation d'un ensemble de principes contraignant les types et rendant compte des idées les plus importantes du chapitre précédent.

3.1 Ce que les types sémantiques ne sont pas

Les théories de types sont des formalismes puissants dans lesquels de très nombreuses opérations sont expressibles. Les types sémantiques, en tant qu'instances de telles théories, ont donc a priori un potentiel d'expressivité extrêmement élevé qu'il est tentant d'utiliser pour y capturer un maximum de phénomènes sémantiques. Plus précisément, les phénomènes sémantiques capturés par une théorie donnée ne dépend que de deux aspects : d'une part, la puissance expressive de la théorie elle-même en tant qu'outil mathématique, et d'autre part, la quantité et le genre d'information sémantique encodé dans les types. La construction d'une théorie des types sémantiques, à l'instar des théories majeures actuelles (GL [135], MGL [149], TCL [4] ou encore UTT [35]), repose systématiquement sur ces deux aspects, et notre discussion dans le présent chapitre ne fera pas exception.

Dans cette section, nous nous intéressons en particulier au second aspect, portant sur l'information sémantique encodée dans les types, que nous cherchons à expliciter au travers d'un prisme minimaliste : quelle est l'information minimale d'un type sémantique ? Pour le déterminer, nous allons confronter la notion de type à diverses notions sémantiques afin d'estimer à quel point les types jouent un rôle essentiel ou non pour l'obtention de ces notions. Ce faisant, nous montrerons également que certaines des théories sus-citées ne sont pas minimales au sens où nous l'entendons ici, c.-à-d. que l'information qu'elles encodent au sein des types est plus large que celle que nous dégageons et pourrait être incorporée de façon alternative dans le système formel — dans le lexique ou dans les formules logiques, par exemple. [46]

3.1.1 Types et référence.

Le rapport entre les mots et le monde est une question centrale et inévitable de la sémantique, qui occupait déjà Frege et Russell il y a un siècle, et qui a certainement occupé bien des philosophes avant eux. Il y a évidemment un jeu d'influences réciproques entre le signifiant et le signifié, entre le mot et son référent, entre l'intension et l'extension, qui font qu'une partie au moins du sens repose sur des observations empiriques. Néanmoins, il est essentiel de reconnaître que l'usage d'un mot, et une partie non-négligeable de sa signification, sont indépendantes de

46. Je reste agnostique sur la question de savoir s'il y a une meilleure approche des types entre le minimalisme que je défends dans ces pages et les approches plus riches des autres théories : ce qui importe avant tout est d'avoir conscience du niveau d'analyse sémantique duquel l'information qu'on utilise provient. Une question intéressante serait d'étudier à quel point les différentes méthodes de représentation influent sur les performances pratiques des systèmes, en terme d'utilisation mémoire et de temps d'exécution, mais cette interrogation dépasse de très loin l'objet de cette thèse.

leurs référents possibles : un moyen simple de s'en convaincre est de rappeler l'expérience de pensée des Terres jumelles proposée par Putnam [144] et expliquée notamment dans [103] : s'il existait un monde parallèle, en tout point semblable au nôtre à ceci près que l'eau est un composé chimique différent, bien qu'ayant les mêmes propriétés que H_2O, un voyageur de notre monde dans celui-là pourrait malgré tout utiliser correctement le mot *eau* dans la plupart des situations. Cette expérience de pensée traduit le fait que le sens du mot *eau* ne tient pas vraiment à ce que l'eau *est* en tant qu'essence (ici en tant que molécule), mais à comment il est possible de l'isoler de son environnement : les propriétés utiles pour cela sont notamment sa nature liquide, transparente, etc., mais il n'est pas possible non plus de réduire l'eau à un même ensemble de caractéristiques spécificiques qui l'isoleraient dans toutes les situations.

Ce qu'il est important de voir ici est qu'il n'y a pas d'anomalie sémantique à appeler *eau* quelque chose qui ne serait pas fondamentalement, chimiquement, de l'eau. Pour donner des exemples plus proches de notre réalité, il n'y a pas d'erreur de catégorie ou autre anomalie sémantique si je désigne à quelqu'un une bouteille pleine étiquetée du nom d'une ville thermale de Haute-Savoie [47] en lui demandant de me servir un verre d'eau, quand bien même le contenu de la bouteille serait en réalité de l'acide chlorhydrique. Par ailleurs, étant donné qu'il est impossible de définir à partir de quelle concentration en HCl de l'eau devient effectivement une solution d'acide chlorhydrique, il existe des situations hypothétiques dans lesquelles différents intervenants pourraient ne pas être d'accord sur la question de savoir si ma demande est correcte ou non. Mais toujours est-il que dans ces situations hypothétiques, quand bien même il y a erreur sur la nature réelle des référents désignés, il est probable que tous les interlocuteurs devinent correctement de quelle entité il est question dans mon énoncé, par exemple si le liquide contenu dans la bouteille est le seul liquide visible. Enfin, il est même envisageable que mes interlocuteurs et moi sachons pertinement que le contenu de la bouteille est de l'eau, mais qu'on puisse en plaisanter sous la forme d'énoncés comme « sers-moi donc un verre de cette vodka » tout en étant compris.

Il ressort de ces exemples un certain décalage entre la signification des mots et les référents qu'ils peuvent effectivement décrire, laissant aux locuteurs de la langue une certaine liberté d'usage qui, bien que possiblement bancale d'un point de vue pragmatique ou épistémique, ne constitue pas à proprement parler une anomalie sémantique. Si les types sémantiques ont pour but de reconnaître de telles anomalies, il faut alors s'attendre à ce que les types n'aient que peu si ce n'est aucune influence sur les référents, ni sur les éventuelles « erreurs de référents » qu'il est possible d'obtenir comme dans les exemples précédents : le genre d'analyse qui permet d'identifier les référents dépasse donc largement le cadre des types. Pourtant, le fait de pouvoir être compris malgré les substitutions de termes comme *eau*, *vodka* ou *acide chlorhydrique* nous apprend quelque chose sur leurs significations respectives : elles comportent au moins une composante qui leur est commune, et qui autorise leur substitution de sorte à préserver l'acceptabilité des phrases concernées malgré la possible rupture avec le contexte réel. Comme l'objectif des types est de contrôler la combinaison des mots de sorte à éviter les anomalies sémantiques, cette préservation de l'acceptabilité hors-contexte des énoncés suggère que ces termes pourraient ainsi avoir, entre autres, des types similaires.

La même forme d'analyse peut s'appliquer à l'exemple célèbre de *étoile du matin* par rapport à *étoile du soir*, proposé par Frege. Quine [146] soutient notamment que ces deux expressions ont des significations différentes parce que le fait qu'ils désignent le même référent ne va pas de soi : il est le fait de résultats astronomiques dont un locuteur peut d'ailleurs ne pas avoir conscience. D'un autre côté, la possibilité que les deux expressions puissent désigner le même référent, que

47. Je laisse les lectrices et lecteurs libres de choisir leur référent préféré pour ce syntagme.

cela soit empiriquement vrai ou faux — et quand bien même le référent en question puisse être un objet astronomique autre qu'une étoile, en l'occurrence une planète —, signifie que ces deux expressions ont bien une fondation commune à leurs deux significations, fondation dont le type sémantique pourrait faire partie. Si cela est exact, n'y a-t-il finalement pas une influence des types sur les référents, contrairement à ce que nous avons avancé juste au-dessus ? En réalité, cela dépend de ce que l'on entend par « influence » ou « indépendance » ici. Les référents auxquel il est possible d'accéder à partir d'un mot sont variés et parfois erronément désignés en fonction du contexte d'énonciation, mais l'observation de ce décalage présuppose un énoncé suffisament correct d'un point de vue sémantique pour autoriser sa confrontation au réel : on retrouve alors l'idée générale des niveaux d'analyse de Sommers, où la vérification de la correction des référents, à l'instar des autres formes d'énoncés vides, sont situés en aval de l'analyse des types. Dès lors, la reconnaissance de relations entre les référents a pour condition nécessaire la correspondance de leurs types ; en revanche, rien n'indique a priori que deux expressions de même type mènent à des référents de même nature : il en va ainsi des étoiles et des planètes, qui sont tous candidats acceptables à la référence d'« étoile du matin ».

Par ailleurs, nombreuses sont les expressions qui n'ont pas de référent du tout ; or de telles expressions sont parfaitement utilisables dans des phrases acceptables ou non, suggèrant qu'elles ont suffisamment de sens pour qu'on puisse en décider malgré l'impossibilité de vérification empirique. On se souviendra par exemple de l'actuel roi de France : le fait qu'il n'y ait personne qui corresponde à cette description dans le monde n'empêche pas de pouvoir évaluer la phrase (1a) comme acceptable, et la phrase (1b) comme sémantiquement problématique.

(1) a. L'actuel roi de France est chauve.
b. L'actuel roi de France est un nombre premier.

Il en va de même pour des noms désignant des entités n'existant pas dans notre monde, comme *fantôme* ou *licorne* : en dépit de leur absence de référent, il est tout à fait possible d'en parler voire de produire des erreurs de catégorie avec. Il paraît raisonnable de supposer que cette capacité est naturellement en lien avec les diverses représentations que l'on peut avoir de ces termes, qui permettent de les rapprocher de référents réels — les licornes des chevaux, par exemple — et d'en déduire une certaine quantité d'information qui influence le pouvoir combinatoire de ces termes. Pour reprendre l'analyse précédente, ces rapprochements suggèrent une possibilité de substitution indiquant une similarité de types. Comme le résume Sommers : « *the difference between a horse and a unicorn is one of existence, not type* » [172, §I].

Englebretsen [56] suggère d'ailleurs que l'existence n'est pas une propriété des individus, mais des totalités, c.-à-d. des mondes possibles : ainsi, l'existence des licornes est négative dans le monde réel mais peut être positive dans un univers de *fantasy*. Cette vision rend obsolète l'importance de la référence au regard de la signification, et donc des types, puisqu'il est toujours possible d'imaginer un monde avec un référent adéquat. Dès lors, la formation d'une signification ne dépend plus du besoin d'avoir un référent empiriquement observable, mais se lie davantage à la manière dont on conçoit les propriétés d'un référent hypothétique et ses interactions possibles avec un monde proche du nôtre, pour une valeur de « proche » qui reste malheureusement non-spécifiée. En particulier, les types sémantiques ne permettent pas de classifier le réel, et ne sont pas soumis aux impératifs de l'existence : si les référents ont pu avoir une influence sur le comportement sémantique des mots qui les désigne, celle-ci est tout au plus résiduelle, et ces comportements peuvent se propager à des mots qui ne désignent rien de notre monde sur une base indépendante. Tout ceci contribue donc à réduire le rôle des types dans l'établissement de la référence comme une condition nécessaire, sans plus.

3.1.2 Types et prototypes.

Néanmoins, le paragraphe précédent invite à se poser une autre question : quel lien les types sémantiques entretiennent-ils avec la représentation mentale des référents possibles ? Après tout, même si nul n'a jamais observé de licorne, l'imaginaire collectif, aux travers de la tradition, de descriptions, de dessins et d'images en 3D, s'est dotée d'une idée assez précise de l'aspect d'une telle entité, que l'on pourrait résumer grossièrement sous la forme d'un cheval muni d'une longue corne frontale. Comme nous l'avons évoqué plus haut, la proximité de cette description avec celle du cheval, pour lequel nous avons une représentation cognitive bien plus nette, pourrait avoir une influence sur la manière dont on conçoit l'interaction imaginaire d'une licorne avec notre environnement : elle partage a priori toutes ses propriétés concevables avec le cheval. Par conséquent, il n'est pas aberrant d'estimer qu'une certaine proximité de significations peut s'acquérir par la proximité des représentations mentales associées.

Cette idée entre en résonance avec certains aspects de la sémantique cognitive, notamment la question de la catégorisation et la théorie des prototypes [152, 153, 70] ; voir aussi [71, §5.1]. Pour la résumer sommairement, cette théorie caractérise notre capacité à catégoriser les concepts, c.-à-d. à les regrouper mentalement par similarité, comme fondée sur des catégories cognitives pour lesquelles nous avons des représentations canoniques, ou *prototypes*, de sorte que le degré de proximité d'un concept avec un prototype traduit son degré d'appartenance à la catégorie associée. Un trait majeur de cette théorie est l'absence de démarcation nette entre catégories : par exemple, *fruits* et *légumes* définissent deux catégories qui comptent des éléments ambigus, par exemple les tomates. Et pour d'autres concepts proches, comme les noix de coco ou les olives, il est encore moins évident de déterminer s'ils appartiennent à ne serait-ce qu'une de ces deux catégories ! Ces catégories s'accordent par ailleurs assez mal avec une approche définitionnelle : a priori, on ne peut pas lister des propriétés communes à toutes les instances d'une catégorie, qui en donnerait une définition précise ; pourtant, l'appartenance d'une entité conceptuelle à une catégorie ou non peut être aisément décidable. La catégorie des oiseaux ne peut pas se résumer aux entités qui volent, puisque cela pourrait inclure les chauve-souris et exclure les manchots, ni aux entités à plumes puisque cela pourrait inclure certains dinosaures ainsi que le serpent à plumes de la mythologie mésoaméricaine. Pourtant, tout locuteur ayant une connaissance suffisante de la signification d'*oiseau* sera capable de dire si oui ou non une entité donnée est un oiseau : ce résultat s'obtient sans nécessiter d'évaluer l'entité selon des critères définis.

Les catégories prototypiques peuvent-elles correspondre à des types sémantiques ? Il paraît assez clair que toutes les propriétés par lesquelles on pourrait essayer de caractériser une telle catégorie sont applicables aux éléments de celle-ci de manière non-absurde : ainsi, toutes les entités tombant dans la catégorie des oiseaux ont une relation décidable avec le prédicat *être capable de voler*, les hirondelles tout autant que les autruches. De manière plus générale, la proximité des éléments d'une même catégorie prototypique paraît suffisante pour affirmer que si un prédicat a l'un de ces éléments dans sa portée, alors il a tout les autres. La simple capacité de pouvoir mettre en relation les différents éléments d'une telle catégorie, comme dans la phrase « une hirondelle n'est pas une autruche », indique au moins que les éléments d'une même catégorie prototypique sont de même type.

En revanche, il n'est pas nécessaire que la réciproque soit valide, et certaines observations nous inviteraient au contraire à penser qu'elle est fausse. Prenons un cas particulier à fins d'illustration : la catégorie des fruits forme-t-elle un type sémantique ou, formulé différemment, y a-t-il un type sémantique des fruits ? Pour que ce soit le cas, si l'on suit la théorie de Sommers, un tel type sera caractérisé par un groupe de prédicats qui ne s'appliquent de manière non-absurde qu'aux fruits, et qui n'admettront rien d'autre dans leur portée. Mais peut-on seulement

trouver de tels prédicats ? La plupart des termes intuitivement associés aux fruits peuvent a priori s'appliquer non-absurdement à d'autres entités : *mûr* ou *issu d'un arbre* s'appliquent aux légumes, *juteux* ou *sucré* à la plupart des aliments, et ainsi de suite. À vrai dire, par leur double statut de produits végétaux et d'aliments, il est fortement probable que tout prédicat ayant les fruits dans sa portée ait l'une ou l'autre catégorie dans sa portée aussi, ce qui rend par conséquent la catégorie des fruits inappropriée pour un type sémantique.

Les catégories prototypiques sont en fait trop restreintes, trop précises pour servir de types sémantiques convenables. Intuitivement, le fait que ces catégories ont des démarcations floues peut même déjà être vu comme un indice de leur différence avec les types : l'absence de limites précises tiendrait justement au fait que des critères pouvant caractériser une catégorie peuvent aussi en caractériser d'autres, suggérant que les portées des prédicats qui s'appliquent aux éléments de ces catégories s'intersectent, et donc en comprennent logiquement la totalité. Un type sémantique se présente alors comme une union de catégories prototypiques qui partagent une certaine compatibilité avec des prédicats communs, union qui se propage jusqu'à ce que la « super-catégorie » résultante fasse état d'une délimitation rigide avec les autres « super-catégories ». En d'autres termes, les types sémantiques sont une notion beaucoup plus basique, beaucoup plus vague que les catégories prototypiques : tout comme pour les référents du monde, la similarité de type est une condition nécessaire à la similarité de catégorisation, mais est loin d'être suffisante.

3.1.3 Types, logique et lexique.

Si la représentation mentale des référents est encore trop éloignée des types, qu'en est-il de la signification et de la connaissance encyclopédique qui est associée ? Comme vu précédemment avec Quine et l'étoile du matin, une divergence de signification n'empêche pas une similarité de type, ce qui suggère là encore la posture des types comme une condition nécessaire et non suffisante. Mais je souhaite développer ici les aspects logiques de la signification pour approfondir autant que possible leur relation avec les types sémantiques, en invoquant notamment la notion de *présupposition*. L'une des limites potentielles à la théorie des prototypes précédemment exposée est qu'elle ne permet de catégoriser que ce qui est effectivement représentable par l'esprit. Pour *licorne*, cela fonctionne bien parce que l'on connaît une définiton de ce mot qui nous fournit des informations suffisantes à la construction d'une représentation mentale, de même que l'on peut avoir des représentations d'entités qui existent bien mais qu'on a jamais vu. Mais supposons que j'introduise en français un nom totalement inventé et parfaitement inconnu, disons *zelpomir* (n.m.), sans en donner de définition, et que je l'utilise dans l'énoncé suivant :

(2) Le zelpomir a mangé tout le gloubi-boulga.

Si l'on reconnaît que cette phrase est correcte (et on n'a pas de raison d'en douter a priori), alors on obtient immédiatement une certaine intuition sur ce qu'est un zelpomir : il s'agit d'une entité capable de manger, donc vraisemblement un être animé, ce qui implique une délimitation spatiale, et équipé d'un organe absorbeur de nourriture, ce qui le rend potentiellement éligible à la capacité d'émettre des vocalisations. Toutes ces observations s'obtiennent seulement par l'intégration que *zelpomir* est un argument sujet acceptable de *manger*, en se basant sur notre connaissance de ce verbe — et indépendamment d'ailleurs de la signification de *gloubi-boulga*.[48] Par ailleurs, on obtient très rapidement sur la base de l'énoncé (2) l'intuition que le nouvel énoncé

48. Relevons tout de même que culturellement, une personne connaissant la signification usuelle de *gloubi-boulga* pourrait aller jusqu'à se représenter un *zelpomir* comme une sorte de dinosaure gentil.

en (3a) ci-dessous est sémantiquement bien formé, alors que celui en (3b) est très probablement une erreur de catégorie :

(3) a. Le zelpomir s'est endormi.
b. ?Le zelpomir a été divisé par 3.

Ces intuitions tiennent notamment à notre connaissance quant aux portées des prédicats utilisés : *manger* et *être endormi* nous sont connus comme compatibles, car les entités pouvant satisfaire l'un de ces prédicats peuvent aussi satisfaire l'autre, et *manger* et *être divisé par 3* nous semblent au contraire incompatibles pour des raisons similaires. C'est donc qu'une certaine quantité d'information sémantique est encodée dans les prédicats eux-mêmes, caractérisant les arguments qu'ils acceptent ; et un argument qui n'obéit pas à ces caractéristiques est donc rejeté, et conduit à une erreur de catégorie. Cette information ne permet cependant pas de construire une représentation mentale d'un zelpomir : ce ne sont que des indices qui esquissent les propriétés qu'il *peut* avoir, mais pas celles qu'il a réellement. [49] On se réfèrera dans la présente thèse à cette information sémantique caractérisant les arguments des prédicats sous le terme de *présupposition*. Ce terme a reçu en linguistique de nombreux usages qui ne s'accordent pas nécessairement avec celui-ci : dans la suite, il faudra considérer ce nom comme un terme technique spécifique à notre discussion.

Les présuppositions s'accordent généralement bien avec l'idée des types sémantiques, comme remarqué par exemple par Asher et Pustejovsky [4, 5] : il s'agit après tout d'encoder l'information sémantique minimale pour garantir que l'application d'un prédicat à un argument sera sémantiquement acceptable. Le lexique, tel qu'entendu au chapitre 1, est également utile ici : si les présuppositions correspondent à un ensemble de traits et de propriétés qui renvoient à la connaissance générale des entités, les lister dans une telle structure facilite leur vérification. Cependant, cette approche se heurte une nouvelle fois au problème structuraliste : y a-t-il une portion de ces informations qui échappe à la connaissance encyclopédique du monde ? Les marqueurs de Katz et Fodor [95], par exemple, sont aussi des présuppositions, or nous avons vu qu'ils manifestent certaines limites au regard de ce que nous attendons des types sémantiques. Les mêmes problèmes sont posés par les postulats de signification, et ainsi de suite.

Il semble que la difficulté majeure de la conception des types sémantiques par rapport à la présupposition et aux autres traits sémantiques logiquement déductibles est d'intégrer la distinction entre un typage correct et une signification correcte — et en particulier vraie. Pour Pap [126], qui préconisait déjà l'étude des présuppositions, les types se distinguent des autres propositions par le fait que les secondes sont systématiquement contingentes, c.-à-d. qu'il est nécessaire de les vérifier empiriquement, alors que celles qui sont issues directement des types sont présentées comme « auto-évidentes ». Mais ceci sous-tend une autre question importante, qui est de savoir si on peut logiquement déduire du typage d'une entité des propriétés qui lui soient obligatoirement vraies. En effet, la plupart de nos déductions logiques reposent sur la vérité des prédications observés, mais pas nécessairement sur leur type. Reprenons notre zelpomir : à partir de l'énoncé (2) ci-dessus, nous avons pu avancer plusieurs de ses caractéristiques possibles, dont un potentiel « organe absorbeur de nourriture ». Mais si un zelpomir n'avait pas de tel organe ? Impossible, dirons-nous, puisque notre premier énoncé asserte qu'il est capable de manger, donc d'absorber la nourriture. Mais cette impossibilité est logique, issue d'une analyse des implications

49. De là l'impossibilité de trouver une définition, une signification, à certains hapax de la langue française pour lesquelles les observations ne sont pas suffisantes : c'est notamment le cas du mot *ptyx*, issu du célèbre sonnet dit « en X » de Stéphane Mallarmé : « Sur les crédences, au salon vide : nul ptyx,/Aboli bibelot d'inanité sonore,/(Car le Maître est allé puiser des pleurs au Styx/Avec ce seul objet dont le Néant s'honore) ».

de la signification de *manger* et de la nécessité d'existence d'un organe pour effectuer cette action ; elle dépasse donc déjà le cadre des types : si la phrase (2) avait été à la forme négative, aurait-on pu aboutir à la même déduction ? Dire qu'un zelpomir mange ou ne mange pas ne permet pas directement de déterminer s'il dispose d'un organe pour l'ingestion de nourriture, mais nous incline malgré tout à se poser la question, ce qui rend le zelpomir « éligible » à la capacité de manger, là où l'on sait qu'il serait absurde de poser la même question pour d'autres types d'entités.

L'usage de la négation, suivant notre discussion de la théorie sommersienne, a la vertu d'aider à mettre en lumière le fait que de nombreuses propriétés sont déduites des présuppositions non pas parce qu'elles forment des phrases signifiantes, mais bien parce que la proposition résultante est tenue pour vraie ou pour possible. Une autre manifestation de ce phénomène se trouve dans certaines interprétations préférentielles, où un mot est si souvent associé à un autre qu'on tend à penser qu'ils fonctionnent à l'exclusion de toute autre combinaison — ce qui n'est généralement pas le cas, même si cela peut être contre-intuitif. Par exemple, l'adverbe *grièvement* n'est guère utilisé en français qu'en collocation avec *blesser* ou d'autres verbes marquant l'idée de dommage physique ; il n'est pourtant pas impossible d'employer ce mot avec d'autres verbes qui autorisent la qualification de leur intensité, tout comme ses synonymes *gravement* ou *sérieusement* : son apparente restriction sélectionnelle résulte ainsi d'une certaine habitude d'usage plus que de propriétés sémantiques intrinsèques à la définition du mot.

Cette habitude d'usage, qui rend des combinaisons finalement plus « naturelles » que d'autre, a inévitablement poussé à rechercher des modèles qui l'expliquent où en tiennent compte ; les informations pertinentes pour cela étant généralement intégrées au lexique. En GL, Pustejovsky utilise ainsi un trait destiné à contenir, si elle existe, l'action pour laquelle un objet est prévu : le quale *télique*. Ainsi, comme il le défend notamment dans [136], le télique du nom *beer* est *drink*, qui est ainsi sa fonction principale. Il fait alors intervenir un constructeur permettant d'encoder cette information sous forme de type ce qui, en tandem avec le type `liquid`, permet d'obtenir le type `beverage`, qui fait partie intégrante de son ontologie. Cela suggère que les liquides sont discriminés par leur capacité à être sélectionnés naturellement comme argument du verbe *drink*, une position confortée par Pustejovsky et Jezek par leur discussion d'un autre liquide, *blood* : « *Blood is a* `liquid` *but it is not meant to be drunk : it can however be re-interpreted as* `beverage` *contextually* » [142, p.207]. La présupposition que certains liquides sont prévus pour être bus et d'autres non dépasse pourtant le cadre strict des types : en particulier, ceci ne permet pas vraiment d'expliquer ce qu'il arrive à *blood* dans le cas d'une proposition négative comme celle-ci :

(4) People do not drink blood.

Cette phrase n'est certainement pas une erreur de catégorie, et peut même être considérée comme vraie. Cela signifie que la prédication de *drink* à *blood* est acceptable. De plus, on peut substituer à *blood* n'importe quel autre liquide, comme *petrol*, et obtenir le même résultat. Il en ressort alors que le type sémantique des arguments de *drink* est bien plus proche de `liquid` que de `beverage`, ce qui questionne sur la pertinence de ce dernier.

En fait, le type `beverage` n'est pas un type sémantique selon l'acception minimaliste que nous cherchons à en faire dans ces pages : il n'y a pas, à ma connaissance, de prédicat qui s'appliquerait de façon signifiante à des boissons mais pas à d'autres liquides, pas même *drink*. Le type `liquid`, en revanche, est un meilleur candidat au rang de type sémantique (en laissant de côté les préocuppations structurelles), car il semble bien discriminer des entités ayant les propriétés physiques pour être bues. Cela ne signifie pas cependant que les types « artéfactuels »

de Pustejovsky sont erronés en tant que tels : il s'agit seulement d'un enrichissement des types sémantiques avec des informations lexicales supplémentaires qui ne relèvent pas de ce niveau d'analyse, ce qui a pour corollaire que l'approche des types en GL n'est pas minimale. En revanche, il est probable que les types qu'il qualifie de « naturels » se rapprochent de l'acception que j'en ai. On notera au passage une propriété un peu déroutante : le type `liquid`, s'il est correct, est strictement inclus dans la portée du prédicat *être liquide*, mais ce dernier est a priori vrai de tout liquide. Cela suggère que certains types peuvent en effet déterminer une propriété logique sur laquelle ils sont vrais [50] ; mais il est difficile d'évaluer si un tel prédicat existe systématiquement.

Ainsi, les types sémantiques ont un lien fort avec la notion de présupposition, dans la mesure où cette dernière forme une collection de propriétés sémantiques à vérifier pour l'obtention d'une signification correcte lors de l'application d'un prédicat à un argument, propriétés dont les types sémantiques eux-mêmes semblent faire partie. Cependant, il y a une fois de plus une distinction subtile à opérer sur l'acception de cette notion de « signification correcte », en séparant ce qui permet à la signification d'être interprétable et ce qui lui permet d'être vraie. Un risque majeur serait de traiter comme relevant des types sémantiques une information qui est en réalité de l'ordre de la déduction logique, et à ce titre, devrait être réservée à des analyses en aval de la vérification des types.

3.1.4 Type et sens.

Une dernière notion à laquelle confronter les types sémantiques est celle du sens. Le terme sens, comme nous l'avons vu au chapitre 2, est souvent compris de différentes manières, de sorte que le sens chez Sommers [170] et le sens chez Katz [94] désignent des concepts fortement distincts. Dans cette dernière section, nous nous intéressons surtout à la seconde acception du sens : celle qui, héritée de l'approche structuraliste (cf. p. ex. [112]), cherche à rendre compte des similitudes et des contradictions qui s'observent entre les mots du lexiques, à travers des propriétés bien connues comme la synonymie, l'antonymie ou encore la méronymie. À cet égard, nous nous intéresserons surtout à la Théorie Autonome du Sens (TAS) discutée par Katz [94].

Ce dernier se montre très critique notamment envers la conception frégéenne du sens, rendue par les héritiers de la pensée de Frege comme une fonction des mondes possibles vers des extensions dans ces mondes, qui a pour conséquence importante que le sens *détermine* la référence, c.-à-d. que le sens est une condition nécessaire et suffisante à l'identification des référents associés. En réponse à cela, il propose un modèle alternatif du sens dont l'unique but est de rendre compte des relations et des propriétés sémantiques observables des phrases et mots, dont les relations de sens du paragraphe précédent, la signification elle-même (ou son absence), l'ambiguïté et la redondance. Une telle approche, argue-t-il, mène à la propriété plus faible que le sens *est médiateur* de la référence, c.-à-d. qu'il n'est plus que condition nécessaire, mais plus suffisante. Cette propriété nous est familière, puisque nous sommes arrivés en sous-section 3.1.1 à des conclusions similaires au regard des types. [51]

50. L'existence de prédicats qui caractérisent exactement un type apparaît à de nombreuses reprises et sous différentes formes dans la littérature du XX^e^ siècle. Carnap [29] utilise notamment le terme de « mot universel », et Pap [126] les appelle « prédicats de type ».

51. Notons que d'après Katz toujours [94, §1.2.1], Putnam adhérait à la conception frégéenne du sens, et son expérience de pensée ne la remettait pas cause mais s'appuyait justement dessus pour éliminer un autre principe reliant la connaissance de la signification à un état psychologique particulier. Notre interprétation de l'expérience de pensée s'appuie en fait sur une conclusion différente de Putnam : ce dernier voyait un problème dans le fait de pouvoir appeler *eau* quelque chose qui n'en est pas, alors que nous nous contentons de pointer du doigt le fait que cela n'empêche pas les phrases qui l'utilisent d'être compréhensibles et dépourvues d'anomalies sémantiques.

Peut-on utiliser la notion katzienne du sens pour construire nos types sémantiques ? Les deux sont très proches, puisque Katz prétend aussi gérer la question de l'absence de signification des phrases grâce à la TAS, ce qui correspond assez bien à la problématique de nos types. Cependant, pour confirmer ou infirmer cette ressemblance, nous sommes amenés à considérer la compatibilité entre types sémantiques et relations de sens. Naturellement, la synonymie ne nous apprend a priori pas grand chose, puisque le critère de substituabilité des expressions dans des relations de prédication transparaît comme un critère incontournable des types sémantiques. L'antonymie, en revanche, est beaucoup plus intéressante : elle traduit en effet l'idée que les sens de deux mots peuvent être opposés, par exemple pour les prédicats adjectivaux *petit* et *grand*. Si l'on s'intéresse aux portées de ces prédicats, on s'aperçoit assez vite qu'elles sont censées être identiques, dans la mesure où tout ce qui peut être qualifié de petit de manière non-absurde peut également être qualifié non-absurdement de grand. En effet, comme nous l'avons évoqué en section 2.5, l'opposition entre *petit* et *grand* tient au fait que chacun est la négation prédicationnelle de l'autre : dire qu'une entité est petite revient alors à asserter un démenti de la propriété d'être grand. Cette analyse s'applique tout aussi bien à toute paire d'opposés de sens, ce qui nous conduit à la conclusion que deux mots antonymes ont le même type sémantique.

Il est même possible d'aller plus loin dans cet esprit : *petit* et *grand* peuvent être vus sémantiquement comme deux pôles qui s'excluent mutuellement sur une échelle abstraite de valeurs qui caractérisent la taille, et d'autres relations d'exclusivité ou de chevauchement de sens peuvent être caractérisés similairement le long d'hypothétiques « espaces » de valeurs. Par exemple, les prédicats de couleurs forment un ensemble de propriétés exclusives (bien que parfois cumulatives) entre elles, qui généralise l'opposition à deux éléments illustrée par l'antonymie ; et là encore, tout argument acceptable pour une couleur est acceptable pour les autres, et a même accès à des nuances moins attendues comme *irisé*, *transparent* voire *incolore*. Il en va de même pour les prédicats exprimant des vocalisations : *miauler*, *aboyer* ou même *parler* sont autant de nuances qui, bien que non complètement exclusives, sont généralement dépeintes comme ayant des délimitations strictes ; mais toute entité que l'on reconnaîtra capable de vocaliser d'une façon ou d'une autre figurera a priori dans la portée de tous ces prédicats. Il apparaît ainsi que les types sémantiques ne sont pas influencés par l'existence de variations de sens autour d'un même concept : la constance du typage semble même se dégager comme condition nécessaire à l'expression de telles nuances.

Le schéma est similaire pour la relation d'hyponymie. Dans l'esprit de la notion de vérité analytique, il est souvent tentant d'exprimer l'hyponymie au travers d'une relation de typage, voire de sous-typage. Pourtant, il semblerait que toutes les hyponymies ne soient pas exprimables à l'aide des types sémantiques. Considérons par exemple l'énoncé suivant :

(5) La bière est une boisson.

D'après la discussion de la sous-section précédente, nous rejettons l'idée qu'il puisse y avoir un type défini par le mot *boisson*. Mais si `liquid` est un type sémantique convenable, il s'avère alors que *bière* et *boisson* sont tous les deux de ce type, auquel cas la relation d'hyponymie est impossible à formuler en terme de types, et ne relève donc pas de ce niveau d'analyse. Et comme il n'y a aucune raison de traiter certaines hyponymies comme des relations de typage et d'autres non, nous en concluons une fois de plus que les types sémantiques ne sont pas appropriés pour l'expression de ces relations de sens.

La question de la méronymie enfin, bien qu'un peu plus complexe a priori, nous semble similaire. Harrison [83] réfute par exemple que la phrase (6) ci-dessous contienne une erreur de catégorie : sa bizarrerie apparente tient davantage à l'ignorance d'un fait contigent à *bed*, c.-à-d.

une connaissance encyclopédique.

(6) The seat of the bed is hard.

Si l'on est d'accord avec la réfutation d'Harrison, cela signifie que *bed* est bien dans la portée de *the seat of*, et donc que leurs types correspondent, alors même qu'il n'y effectivement pas de relation méronymique entre *seat* et *bed*. De façon plus générale, il semblerait que la méronymie nécessite bien trop de connaissance contextuelle ou encyclopédique pour avoir une quelconque influence sur les types sémantiques, mais que l'accord des types est nécessaire à l'expression d'une telle relation.

Mis bout à bout, tous ces exemples démontrent que la plupart des relations couvertes par la TAS et par les autres théories du sens dans son acception structuraliste ne devraient pas être capturables par les types. Une fois de plus, il semblerait plutôt que la correspondance de types entre les expressions connectées par ces relations soit une condition nécessaire et non suffisante, comme si, pour reprendre la formulation de Katz, les types étaient médiateurs du sens. Ainsi, la théorie de Katz contient strictement les types sémantiques, et n'est donc pas minimale : les relations de sens appartiennent au lexique et à l'analyse logique de la langue, mais pas à l'analyse des types directement.

Bilan. Les différents cas traités dans cette section ont contribué cumulativement à dégager les contours des types sémantiques comme une notion extrêmement primitive au regard des différents phénomènes sémantiques et pragmatiques envisageables. En basant leur construction sur l'idée de détecter les anomalies sémantiques de la forme « erreur de catégorie », nous obtenons alors que les types sémantiques ne peuvent pas être des caractéristiques appliquées directement à des référents, puisqu'ils existent indépendamment de ceux-ci et ne suffisent pas à les déterminer lorsqu'ils existent ; ils ne sont pas non plus liés à la catégorisation mentale des entités parce que les catégories que nous formons intuitivement sont basées sur des propriétés vraies et pas sur des propriétés non-absurdes, et il en va de même pour la distinction des types comme une sous-partie stricte des présuppositions logiques à la compositionnalité. Enfin, les types ne permettent pas de caractériser les relations de sens, qui font appel à trop de connaissance encyclopédique. Le point commun à toutes ces notions est néanmoins qu'à chaque fois, une certaine correspondance de type apparaît comme une condition nécessaire et non suffisante.

3.2 Axiomatique générale des types sémantiques

Dans cette section, nous démarrons la formalisation proprement dite de notre théorie linguistique des types, en nous fondant sur toutes les remarques accumulées au fur et à mesure de nos discussions. Rappelons que notre objectif premier ici est d'énoncer une série de règles, ou de principes, qu'une théorie des types sémantiques est supposée devoir suivre pour faire état d'un comportement cohérent avec le but qu'on leur confère, à savoir contrôler l'apparition d'anomalies sémantiques au sein des phrases. Notre matériau de base étant linguistique et philosophique, il ne faudra pas s'attendre à ce que la théorie développée dans le reste de ce chapitre soit strictement mathématique : nous nous accorderons une certaine souplesse conceptuelle appropriée au traitement de ce matériau de base. Néanmoins, les principes qui seront énoncés devront pouvoir être accomodés mathématiquement, dans le sens où il devra être possible de définir des instances formelles de théories des types dont on pourra vérifier qu'ils sont en accord avec lesdits principes. En cela, la formalisation que nous proposons ici peut être vue comme transitoire, permettant de faire le pont entre la linguistique d'une part, et l'informatique théorique d'autre part.

3.2.1 Notes préliminaires sur l'objet d'étude

À ce stade de nos réflexions, les nombreuses discussions que nous avons tenues autour de la notion de type sémantique dégagent un constat clair : les types sémantiques sont nécessaires uniquement dans la mesure où nous nous appuyons sur le principe de compositionnalité, tel que posé et reconnu depuis au moins Frege et Husserl (cf. [33]). Nous prenons pour point de départ le fait que les mots se combinent pour former des phrases, et que ces phrases peuvent être évaluées sur au moins deux critères : un premier syntaxique, déterminant si la phrase est grammaticalement bien formée, et un second sémantique, déterminant si la phrase a une signification. Ces aspects sont tous deux fondés sur la vérification des combinaisons manifestées par les mots, qui doivent suivre une certaine structure et obéir à certaines règles qu'il est nécessaire de préciser. Les catégories syntaxiques tout comme les types sémantiques sont ainsi des valeurs qui se manifestent pour toutes les expressions possibles du langage, y compris les mots pris séparément, mais qui ne prennent leur sens que lorsque ces mots sont mis en situation de combinaison avec d'autres mots.

Une première remarque qu'il nous paraît nécessaire de formuler à ce sujet est que, en dépit de l'approche d'interface syntaxe-sémantique traditionnelle, telle qu'on la retrouve par exemple chez Montague — lequel hérite de l'École polonaise ainsi que du programme générativiste —, et en dépit également de la vision de Sommers quand aux niveaux d'analyse linguistique, et en particuler du niveau 0, les évaluations de la grammaticalité et de la signification peuvent être quelquefois menées indépendamment l'une de l'autre ; et en particulier, il est possible d'évaluer des éléments de signification de certaines expressions agrammaticales. Un exemple qui illustre ce phénomène peut s'observer ci-dessous en (7), lorsqu'on traduit mot à mot de l'anglais au français la phrase proposée par Katz pour illustrer les problèmes des degrés de grammaticalité au regard de l'intelligibilité (cf. section 2.3), et discutée par Cohen qui est à l'origine de la remarque qui va suivre [93, 45] :

(7) a. ?The beef cut sincerity.
b. *Le bœuf a coupé sincérité.

Là où la phrase originale en anglais est parfaitement grammaticale, sa traduction mot à mot en français ne l'est pas : il manque un déterminant devant *sincérité*. Pourtant, ce mot manquant ne nous empêche pas de reconstruire le sens de la phrase, et de nous rendre compte qu'elle contient une erreur de catégorie flagrante. Ainsi, nous sommes capable de réaliser une courte analyse sémantique d'une phrase agrammaticale. Cependant, nous voyons intuitivement que l'agrammaticalité de cette phrase est « légère » : la perte d'un déterminant, ou le remplacement d'un syntagme nominal par un simple nom, qui n'est pas commun en français, est une divergence assez faible au regard de la langue, comparé à d'autres situations comme par exemple la phrase suivante :

(8) *Bœuf sincérité le coupé a.

Ainsi, l'évaluation de la sémantique d'une phrase, ou du moins du respect de ses catégories logiques, n'a pas besoin de la grammaticalité totale : elle peut être menées sur des expressions qui sont « suffisamment grammaticales », pour une acception du terme qui reste à préciser formellement, mais dont on se doute qu'elle a trait à la manière dont les mots sont composés. Au cœur de ce phénomène réside bien sûr l'idée que sont essentiels les mots porteurs de significations, dits *catégorématiques* : les verbes, les noms, et dans une moindre mesure les adjectifs et les adverbes. À l'inverse, les mots *syncatégorématiques* comme les déterminants apparaissent bien

plus dispensables au regard de l'évaluation du sens et du typage. L'ordre des mots peut aussi avoir son importance, comme on a pu le voir dans l'exemple (8) — mais à des degrés divers selon la langue, les langues déclinées ou utilisant moins systématiquement les déterminants étant a priori moins influencées par ce critère. On pourrait ramener cela à la nécessité d'assurer un minimum d'approximation pour les syntagmes nominaux et verbaux, qui forment a priori le noyau dur d'une proposition : une fois les différents groupes établis, le travail principal consiste alors à reconstituer les dépendances des mots, ou leurs relations de prédication, car c'est là-dessus que semble se fonder la détermination du sens, comme discuté notamment dans les travaux de Katz et Postal [96].

Nous supposons alors que des fondations convenables pour les types sémantiques doivent nécessairement se reposer sur l'aspect prédicatif de la syntaxe, et plus particulièrement de la syntaxe profonde, où la prédication est caractérisée par la notion de *rôle thêta* [60, 148] qui fixe des indications sur les arguments syntaxiques des expressions, afin par exemple que le sujet de surface d'une forme passive soit bien reconnu comme le patient du verbe, et non son agent. Nous formons alors à partir de cette idée une notion relativement informelle d'*expression suffisamment connectée* qui rend compte de l'idée que les mots catégorématiques doivent être correctement reliés entre eux par des relations de prédications statuant leurs rôles thêta. Afin de préciser les objets initiaux de notre théorie, nous soulignons par ailleurs que le terme « mot », que nous utilisons également de façon informelle jusqu'ici, doit être entendu comme *lemme* ou *unité lexicale*, traduisant ainsi le fait que les mots composés lexicalisés, comme *pomme de terre*, sont traités comme des éléments atomiques au sein de nos expressions syntaxiques.

3.2.2 Axiomes généraux

Nous récapitulons ces considérations syntaxiques dans les premiers axiomes qui régissent notre théorie des types sémantiques, en commençant par définir notre matériau linguistique de base. Ce premier axiome suppose une notion d'unité lexicale que nous supposons déjà acquise, en vertu de la discussion qui précède.

(A1) Il existe une notion d'*expression* telle que toute unité lexicale soit une expression, telle que la concaténation de deux expressions soit une expression, et telle que l'insertion d'une expression au sein d'une autre soit aussi une expression.

On retrouve dans cette définition les bases de la grammaire générative, l'opération d'insertion rappelant notamment que les langues naturelles ne sont pas hors-contextes [167] ; on notera en particulier que cette définition est compatible avec les opérations de substitution et d'adjonction dans les grammaires d'arbres adjoints (cf. p. ex. [91]). Dans la continuité de ce point de vue, il est nécessaire de pouvoir accéder aux différentes étapes de construction d'une expression, que nous ramenons ici à la simple capacité d'énumérer les sous-expressions qui la constituent :

(A1.1) Il existe pour chaque expression l'ensemble de ses *sous-expression*, tel que chaque expression soit sous-expression d'elle-même, et tel que l'ensemble d'une expression formée par concaténation ou insertion contient toutes les sous-expressions des deux expressions initiales. Deux sous-expressions sont dites *distinctes* si aucune n'est sous-expression de l'autre.

Nous énumérons ensuite une série de caractéristiques des expressions que nous appelons ici des *traits* ; formellement, un trait est une fonction de l'ensemble des expressions vers un ensemble possiblement non-spécifié, imposant par conséquent que chaque expression a exactement une valeur pour chaque trait qui la caractérise. Dans la suite, nous distinguerons également les

propriétés comme le cas particulier des traits à valeur dans l'ensemble $\{0, 1\}$ des valeurs de vérités, 0 étant comme usuellement le faux et 1 le vrai. Ceci permet alors de raisonner sur les propriétés à l'aide de la logique du premier ordre classique. Le premier trait que nous introduisons ici est la notion de catégorie syntaxique, qui permet là encore de faire le lien avec la théorie syntaxique générative et de fonder la propriété de grammaticalité.

(A1.2) Il existe un trait des expressions, appelé *catégorie syntaxique*, qui rend compte de leur nature grammaticale, ainsi qu'un ensemble de règles telles qu'une expression a une catégorie syntaxique définie si et seulement si elle respecte toutes ces règles. Une expression ayant une catégorie syntaxique définie est dite *grammaticale*, et *agrammaticale* sinon.

On comprendra donc que l'ensemble des catégories syntaxiques contient une valeur spéciale de catégorie indéfinie, permettant de marquer les expressions agrammaticales. Le second trait dont nous avons besoin fournit toutes les informations nécessaires sur les arguments obligatoires des expressions, dont leur nombre et leur nature, c.-à-d. a minima la catégorie syntaxique attendue. Nous donnons à ce trait le nom générique de *structure argumentale* pour autoriser l'utilisation de toute théorie pertinente à cette fin, mais comme évoqué plus haut, nous savons par exemple que la notion de rôle thêta conviendra bien pour cette tâche.

(A1.3) Il existe un trait des expressions, appelé *structure argumentale*, qui précise au moins le nombre d'arguments obligatoires de l'expression ainsi que leurs catégories syntaxiques attendues, et tel que la présence d'arguments qui satisfont toutes les contraintes de la structure argumentale soit une condition nécessaire à la grammaticalité.

L'axiome suivant est légèrement plus vague, et porte sur notre capacité à analyser la structure profonde des expressions, en établissant la catégorie syntaxique des sous-expressions qui la constituent, et surtout en restituant les relations de dépendance ou de prédication par le marquage de sous-expressions comme remplissant le rôle d'argument obligatoire d'une autre sous-expression, elle-même marquée comme son prédicat. La manière précise dont cette information est établie et représentée ne nous préoccuppe pas ici : seul compte le fait que cette information existe, puisse être reconstituée et consultée à tout moment. Notons néanmoins que nous n'imposons pas ici qu'une expression marquée comme argument d'une autre remplisse tous les critères de validité imposées par la structure argumentale : les seules contraintes seront que chaque argument obligatoire d'une expression ne pourra être marqué au plus qu'une fois, et que tout marquage implique nécessairement l'existence d'un argument obligatoire correspondant dans une structure argumentale.

(A1.4) Il existe un mécanisme d'analyse des expressions permettant de vérifier si ses sous-expressions ont des catégories syntaxiques, et permettant de marquer une sous-expression comme l'un des arguments obligatoires d'une sous-expression distincte.

Nous en arrivons alors à la construction de la notion d'expression suffisamment connectée évoquée plus haut, qui se repose principalement sur l'idée que les marquages argumentaux reflètent la construction de l'expression : chaque opération de construction doit instaurer une relation de dépendance, de sorte que toutes les sous-expressions sauf une soient arguments obligatoires d'une autre. Il n'est en revanche pas nécessaire là encore que les arguments ainsi marqués remplissent toutes les conditions attendues, y compris du point de vue de la catégorie syntaxique : à l'exemple de la phrase (7b) plus haut, il est possible dans certains cas de traiter sémantiquement des phrases qui ne sont pas pleinement grammaticales. [52]

52. Une possibilité pour renforcer la condition de suffisante connexion serait d'imposer à ce que chaque argument marqué ait une catégorie syntaxique « proche » de la catégorie attendue, pour une notion de « proche » à préciser.

(A1.5) Il existe une propriété des expressions, dite de *suffisante connexion*, telle que toute unité lexicale est suffisamment connectée, et telle que toute expression obtenue par concaténation ou insertion de deux expressions initiales suffisamment connectées est elle-même suffisamment connectée si et seulement si elle marque l'une des expressions initiales comme argument de l'autre.

Nous restituons enfin une relation logique évidente entre les deux propriétés introduites dans nos premiers axiomes, ce qui conclut alors la série des axiomes syntaxiques :

(A1.6) Toute expression grammaticale est suffisamment connectée.

Une seconde série d'axiomes s'intéresse cette fois à l'aspect sémantique des expressions, et en particulier des expressions suffisamment connectées, que nous prenons comme les seules expressions susceptibles d'être évaluées par une analyse sémantique. Notre première étape sera de formaliser l'ensemble des niveaux d'analyses possibles, en ayant naturellement l'échelle de Sommers en tête, sans toutefois nous y limiter : nous faisons plus simplement l'hypothèse qu'il existe une famille de propriétés sémantiques des expressions, que ses dernières peuvent donc satisfaire ou non. De telles propriétés incluent par exemple « avoir du sens », « être logiquement cohérente », etc. Cette famille est par ailleurs structurée selon l'implication logique des propriétés entre elles, quantifiée universellement sur l'ensemble des expressions suffisamment connectées.

(A2) Il existe un ensemble non-vide de propriétés des expressions suffisamment connectées, appelé ensemble des *propriétés sémantiques*, partiellement ordonné par la relation « être une condition nécessaire de », telle que P est plus petite que Q si pour toute expression suffisamment connectée, Q vraie sur cette expression implique P également vraie.

Il convient alors de remarquer que l'on sépare formellement les propriétés syntaxiques et sémantiques, de sorte que les propriétés introduites en (A1.2) et (A1.5) n'appartiennent pas à cet ensemble de propriétés :

(A2.1) Les propriétés de suffisante connexion et de grammaticalité ne sont pas des propriétés sémantiques.

Dans un second temps, nous introduisons une propriété sémantique majeure et bien connue, la signification. Afin de garder la présente théorie aussi ouverte que possible, nous évitons soigneusement de spécifier quelque contrainte que ce soit sur la définition de cette notion. En effet, la nature profonde de la signification, qui fait l'objet de nombreux débats depuis plus d'un siècle, n'est pas indispensable à nos fins : seule compte notre capacité à évaluer une expression comme signifiante ou non. C'est pourquoi les hypothèses que nous plaçons sur cette notion demeurent très superficielles.

(A3) Il existe un trait des expressions suffisamment connectées, appelé *signification*, qui peut être vide, mais tel que toute unité lexicale a une signification non-vide.

Nous parlerons donc de signification vide pour dire que l'expression considérée n'a pas de signification. Ceci autorise entre autres la représentation de la signification sous forme ensembliste, permettant en particulier d'utiliser des ensembles à deux éléments ou plus pour rendre compte des éventuelles ambiguïtés des expressions. Nous n'approfondissons néanmoins pas cet aspect au sein des axiomes, où seule la distinction entre vide et non-vide importe. Pour ce qui est de la structure interne des signification au-delà de cette distinction, nous nous contenterons

Par exemple, on pourrait autoriser uniquement les arguments dont la catégorie syntaxique est contenue dont celle attendue, en tant que tête notamment, autorisant ainsi un nom là où un syntagme nominal serait attendu.

de la première hypothèse suivante, qui n'est autre qu'une simple réaffirmation du principle de compositionnalité :

(A3.1) Si la signification d'une expression suffisamment connectée est non-vide, alors les significations de ses sous-expressions sont également non-vides, et sa signification est fonction des significations des sous-expressions qui la composent, ainsi que de la manière dont ces dernières sont combinées.

La seconde est destinée à représenter notre *compétence* au regard de la signification, c.-à-d. notre capacité intuitive à déterminer si une expression suffisamment connectée a une signification ou non. Rappelons qu'une expression peut échouer à avoir une signification pour diverses raisons non équivalentes entre elles : erreur de catégorie, incohérence logique, référent vide, condition de temporalité non remplie, etc. Cette capacité intuitive sera présentée ici sous le terme d'*oracle*, par analogie avec les oracles des machines de Turing (cf. p. ex. [127, §14.3]), exprimant notamment que la décision de cette propriété est supposée s'effectuer sans analyse approfondie et complexe.

(A3.2) Il existe un oracle permettant de décider si la signification d'une expression suffisamment connectée est non-vide.

Nous en arrivons alors à la dernière hypothèse liée à la signification : nous considérons ici que l'établissement de la signification est l'aboutissement de l'analyse sémantique, ce qui signifie notamment qu'avoir une signification non-vide est la plus grande propriété sémantique que l'on puisse concevoir, imposant par conséquent une contrainte caractéristique sur l'ensemble des propriétés sémantiques :

(A3.3) La propriété d'avoir une signification non-vide est le plus grand élément de l'ensemble des propriétés sémantiques, ordonné selon la relation définie à l'axiome (A2).

Ceci conclut l'énumération de nos axiomes.

3.2.3 Existence et caractéristiques primaires

Maintenant que les axiomes sont posés, nous avons délimité l'objet de notre investigation et séparé convenablement les propriétés syntaxiques et sémantiques. L'objectif de cette nouvelle sous-section va être d'introduire la notion de type sémantique proprement dite, et d'expliciter ses premières propriétés au regard des éléments introduits par nos axiomes. Comme nous avons pu le voir dans la section 3.1, le respect des types sémantiques semble se dégager comme une sorte de propriété primitive des analyses sémantiques, en amont de toute considération logique ou encyclopédique sur le contenu sémantique des expressions considérées. Nous allons par conséquent baser la construction des types sémantiques sur cette idée, à l'aide de plusieurs postulats quant à leur existence et leurs premières propriétés.

Comme annoncé plus haut, nous allons poser l'existence des types comme des valeurs abstraites permettant de rendre compte de nos observations sur la capacité des phrases à avoir une signification et à satisfaire diverses autres propriétés sémantiques. Pour ce faire, nous munissons nos types sémantiques d'une propriété caractérisant leur formation, de sorte notamment qu'un type sémantique « mal formé » corresponde à l'idée d'erreur de catégorie dans les théories de Ryle et Sommers. En ce qui concerne le domaine d'application des types, Ryle [157] écrivait : « *what are types types of ?* », et exprimait notamment son inquiétude quant à la possibilité qu'en traitant les types, nous nous retrouverions plutôt à parler de grammaire. Nous espérons que nos axiomes syntaxiques et sémantiques suffisent à séparer les deux aspects de manière suffisante pour dissiper ses craintes, en fondant comme domaine des types les expressions suffisamment

connectées et en visualisant les types comme des valeurs sémantiques. Notre premier postulat asserte alors l'existence des types sémantiques selon tous ces critères.

(P1) Il existe un trait des expressions suffisamment connectées, appelé *type sémantique*, tel qu'il existe une méthode permettant de déterminer si un type sémantique est bien formé ou non, et tel que la propriété de bonne formation du type est le plus petit élément de l'ensemble des propriétés sémantiques, ordonné selon la relation définie à l'axiome (A2).

Ce postulat affirme en particulier que nous prenons bel et bien la bonne formation du type sémantique comme la plus petite condition nécessaire à l'affirmation de toute propriété sémantique. À cet égard, un second principe se dégage comme une conséquence immédiate du premier et des axiomes (A3) et (A3.3) :

(P2) Toute unité lexicale a un type sémantique bien formé.

Ceci ouvre la voie à l'élaboration sur la propriété de bonne formation des types. En particulier, il nous paraît opportun de rendre compte du fait que si une expression contient une erreur de catégorie, alors toute autre expression admettant la première contiendra également une erreur de catégorie : en d'autres termes, l'erreur de catégorie, c.-à-d. la malformation des types, se propage inductivement au sein des expressions. Nous formulons alors le postulat suivant afin d'asserter ce phénomène dans notre théorie :

(P3) Si une expression suffisamment connectée admet pour sous-expression une expression dont le type sémantique est mal formé, alors le type de cette première expression est lui aussi mal formé.

Par ailleurs, nous avons écrit au début de la sous-section 3.2.1 que les types n'avaient de réel intérêt que dans la mesure où l'on se basait sur le principe de compositionnalité, dont nous avons restitué une formulation en tant qu'axiome (A3.1). Par contraposition du postulat qui précède, les sous-expressions d'une expression dont le type est bien formé ont elles aussi des types bien formés. Afin de consolider les propriétés des types au regard de la compositionnalité, nous proposons alors de reprendre ce même principe dans une application spécifique aux types, qui induit l'idée que les types se construisent à l'aide d'opérations internes :

(P4) Le type sémantique d'une expression suffisamment connectée est fonction des types sémantiques des sous-expressions qui la composent, ainsi que de la manière dont ces dernières sont combinées.

À cet égard, on peut remarquer que ce postulat est le premier à caractériser les types sémantiques eux-mêmes en tant qu'ensemble structuré, que le premier postulat a défini comme le codomaine du trait associé. À ce stade, nous n'avons encore aucune idée de quelles pourraient être ces opérations internes sur les types, et de très nombreuses possibilités s'offrent à nous. Néanmoins, nous sommes loin d'avoir fait le tour des propriétés de ces types, et la section suivante va justement avoir pour objectif d'approfondir cette question en tentant d'identifier les principes structurels internes aux types, sur la base des diverses discussions menées tout au long de la présente partie I.

Toutefois, avant de conclure cette section qui a posé les fondamentaux de la théorie linguistique des types, il nous semble important de renforcer la distinction entre types sémantiques et catégories syntaxiques, dont l'introduction des expressions suffisamment connectées s'est faite garante : le postulat suivant montre à cet effet que les deux propriétés que sont la grammaticalité et la bonne formation des types sont décorrélées. Ce postulat est assez naturel au vu des discussions menées en section 2.5, l'exemple canonique de Ryle en étant une bonne illustration.

(P5) Il existe des expressions grammaticales dont le type sémantique est mal formé.

La présente théorie a par ailleurs pris soin de séparer la grammaticalité d'une part et la suffisante connexion d'autre part. Cette précaution autorise en effet le traitement d'exemples inspirés de celui de Katz, vu ci-dessus en sous-section 3.2.1 : on pourra notamment se convaincre qu'une phrase comme « chat mange souris », quoique grammaticalement incorrecte et réductrice, comporte tout de même suffisamment d'informations pour inférer une signification cohérente, ce qui impliquerait par notre postulat (P1) que son type est bien formé. De fait, la présente théorie n'impose ni ne s'oppose au traitement de ces exemples : on pourra tout à fait l'appliquer en identifiant la grammaticalité et la suffisante connexion — car ceci reste en accord avec l'axiome (A1.5) —, ou au contraire en séparant les deux, auquel cas on pourra alors reconnaître l'existence d'expressions agrammaticales mais suffisamment connectées dont le type sémantique est bien formé, renforçant encore la décorrélation précédemment évoquée.

3.3 Fondations structurelles et ontologiques des types

Grâce aux axiomes et aux postulats de la section précédente, nous disposons d'un environnement général clairement défini à l'interface syntaxico-sémantique dans lequel nous avons inscrit formellement la notion de type sémantique, que nous considérons comme la plus petite condition nécessaire à l'établissement d'autres propriétés sémantiques. Nous avons pour l'instant une vision externe de ces types : les postulats n'ont fait que statuer sur l'existence d'un ensemble de tels types, qui n'en demeure pas moins complètement abstrait. Les seuls aperçus que nous avons de l'intérieur de cet ensemble sont d'une part le fait que les types peuvent être bien formés ou mal formés, et d'autres part le fait qu'ils sont construits par le biais d'opérations internes. Le but de la présente section est donc naturellement de préciser l'organisation interne de cet ensemble de types, en élaborant sur ces deux premiers aperçus à partir des discussions faites au long de cette partie linguistique.

Nous prenons comme point de départ de notre formalisation la reconnaissance que l'organisation interne des types se découpe en deux aspects distincts : l'aspect structurel d'une part, et l'aspect ontologique d'autre part, chacun étant hérité d'un point de vue différent quant aux problèmes sémantiques de la compositionnalité. Le premier a trait au lien entre la syntaxe et la sémantique qui s'incarne dans les dépendances de la syntaxe profonde, portant notamment sur les correspondances entre arguments syntaxique et sémantique ainsi qu'entre phrases grammaticales et propositions logiquement complètes. Cet aspect est en cela le reflet de la théorie russellienne des types dont le système de Montague est une application concrète évidente. Le second aspect, quant à lui, porte principalement sur les considérations lexicales sous-jacentes aux théories du sens de Sommers, et permet en particulier de rendre compte de la notion d'erreur de catégorie — cette propriété étant intrinsèquement liée à celle des types comme nous avons pu le voir en section 3.1.

La séparation de ces deux aspects nous apparaît comme indispensable ici : en effet, la théorie catégorielle de Sommers est fondée sur une relation de prédication dont les exemples au travers des travaux du philosophe ne sont explicités que sur des prédicats monadiques ; or, si cet aspect monadique facilite l'identification entre types d'arguments et types de prédicats dans ce cas précis, il laisse en suspens les considérations similaires qui devraient être établies pour les prédicats non-monadiques, comme les verbes transitifs par exemple, considérations qui pourront être prises en charge par l'aspect structurel de notre théorie. À l'inverse, la structure des types est entendue comme reflètant les caractéristiques syntaxiques des énoncés, dont leur grammaticalité, qui est une condition nécessaire à l'établissement d'une signification complète et directe ;

mais les considérations de la sous-section 3.2.1 nous ont engagés à considérer également des énoncés suffisamment connectés et agrammaticaux, pour lesquels nous pouvons espérer obtenir a minima une signification partielle [53] et surtout que nous pouvons reconnaître comme absurdes ou non malgré leur syntaxe incomplète : la composante ontologique de notre théorie permettrait de rendre compte de cette capacité.

Ceci établit donc une double organisation des types sous ces deux aspects. À ce stade, nous n'avons aucune idée de la manière dont ils s'articulent entre eux, ni même s'ils le doivent. Pour la simplicité de notre exposé, nous prenons le parti d'introduire la notion formelle de *composante*, qui représente une fonction des types dans un autre ensemble, de sorte à raisonner sur les propriétés des composantes plutôt que sur les types directement. [54] Dans la suite, nous commençons par traiter séparément les composantes correspondants aux deux aspects évoqués ci-dessus, avant d'étudier leurs éventuelles interactions pour répondre à certains des phénomènes sémantiques complexes qui motivent notre étude.

3.3.1 La composante structurelle

Notre première étape est donc de discuter de la structure des types. Puisque la langue est compositionnelle, il est souhaitable que les types le soient aussi, et soient en particulier capable d'évaluer les relations de prédication établies dans les expressions suffisamment connectées. L'idée générale est alors que chaque argument obligatoire d'une telle expression se voit attribuer une *spécification* des types sémantiques qui peuvent occuper cette position sans générer d'anomalie. Cette notion de spécification doit être comprise comme une liste de conditions sur les composantes structurelles et ontologiques des types, qui devront être respectées par le type de l'expression argument. C'est principalement à travers ce comportement que nous construisons nos composantes structurelles, qui sont définies par le principe qui suit :

(S1) Il existe une composante des types sémantiques, appelée *composante structurelle*, telle que si une expression suffisamment connectée admet un argument obligatoire dans sa structure argumentale, alors la composante structurelle de son type sémantique contient une spécification des types sémantiques acceptables pour cet argument.

Ce premier principe signifie notamment que le type sémantique reflète la structure argumentale de la syntaxe. En revanche, notre formulation n'implique pas que toutes les spécifications d'arguments des types sémantiques sont nécessairement l'incarnation d'arguments syntaxiques : il est tout à fait possible d'avoir des structures de types admettant un ou plusieurs arguments alors que l'expression syntaxique initiale n'en admet pas. Un exemple d'une théorie sémantique dont les types utilisent des arguments supplémentaires par rapport à la syntaxe est la sémantique néo-davidsonnienne, où les verbes sont enrichis d'un argument représentant les évènements, qui ne renvoie pas à un argument syntaxique obligatoire. De même, le type des intensions dans le système montagovien est également un ajout des types par rapport à la syntaxe. Notons enfin que dans le système montagovien toujours, les noms communs sont traités comme des prédicats monadiques alors que les noms n'admettent aucun argument syntaxique.

53. Nous restons volontairement évasifs sur cette notion puisque la discussion des théories de la signification n'est pas l'objet direct de la présente thèse, — à plus forte raison que notre théorie des types sémantiques ne fait qu'un usage lointain et imprécis de cette notion. Pour donner cependant une idée intuitive de ce que nous entendons par signification partielle, nous pouvons par exemple suggérer qu'une phrase sans déterminant (« chat mange souris ») puisse être vue comme une formule logique dont les quantificateurs sont non-spécifiés.

54. Notons que cette approche consiste à considérer l'ensemble des types comme un système de valeurs cachées dont on ne peut produire que des observations dans un état donné, déterminé par l'expression initiale : en cela, notre théorie traite l'ensemble des types comme une coalgèbre (cf. [89] pour une introduction.)

Par ailleurs, rappelons que l'axiome (A1.5) suppose que les expressions acceptables en argument ne sont pas nécessairement compatibles avec la catégorie syntaxique attendue, mais peuvent être considérées comme « proches » pour une définition du terme qui dépend de l'implémentation de la présente théorie. Or, dans l'hypothèse de la séparation de la grammaticalité et de la suffisante connexion, certaines expressions peuvent avoir un type bien formé même si elles sont agrammaticales, si tant est que leurs prédications sont assez proches de ce qui est attendu : par conséquent, la détection d'anomalies sémantiques sur la base d'une incompatibilité structurelle n'est pas la priorité des types sémantiques, et leurs spécifications doivent donc être considérées comme assez larges pour rendre compte de cette largesse syntaxique, quitte à insérer des accomodations de typage comme opérations internes des types. Il n'en demeure pas moins que ces spécifications sont la clé de la détection d'erreurs de catégories ; aussi sont-elles à la base des propriétés de formation des types : c'est l'objet du principe ci-dessous, qui instaure le cas de base pour l'apparition d'un type mal formé.

(S2) Si le type sémantique d'une expression donnée ne remplit pas les spécifications attendues pour un argument de la composante structurelle du type d'une autre expression, alors toute concaténation ou insertion marquant la première expression comme argument de la seconde pour l'argument correspondant résulte en une expression dont le type est mal formé.

Il est tout aussi important de définir le comportement des types en cas de combinaison réussie. Afin de respecter le principe de compositionnalité énoncé au postulat (P4), il convient d'expliciter comment la structure du type d'une expression peut être établie sur la base de ses plus grandes sous-expressions strictes ; pour cela, nous formulons le principe suivant, qui garantit notamment que les arguments encore non évalués sont préservés dans le type final :

(S3) Si une expression suffisamment connectée est bien typée, alors la structure de son type contient toutes les spécifications de la structure du type de sa sous-expression marquée comme prédicat, excepté celle correspondant à l'argument marqué pendant la formation de l'expression.

Cette formulation, bien que contraignant la composition des types, laisse néanmoins une marge de manœuvre assez large pour accomoder différentes approches. En particulier, il n'est pas précisé si l'information contenue dans le type sémantique de l'argument est réutilisée dans le type final, ou simplement oubliée ; un exemple de cas où elle est réutilisée partiellement est à trouver dans les théories de types avec polymorphisme, où le type d'un argument peut déterminer la valeur d'une variable présente dans d'autres spécifications.

Le principe suivant porte sur la diversité des structures. Pour qu'une théorie des types soit correctement utilisable, il convient qu'elle soit réductive, c.-à-d. qu'elle permette de regrouper plusieurs expressions différentes sous une même structure de type si cela est pertinent. Comme nous l'avons vu au chapitre 2, le regroupement d'expressions est souvent perçu à travers la notion de substituabilité, selon divers critères qui la contraignent ; ici, nous n'avons besoin que d'une similarité en syntaxe profonde. En effet, la structure des types sémantiques repose principalement sur les arguments que l'on peut assigner à l'expression et à sa traduction sémantique, lesquelles sont déterminées par la structure argumentale de l'expression ainsi que par sa catégorie syntaxique, qui préfigurent le rôle de l'expression au sein de la phrase et de la proposition. Ceci permet donc d'établir le principe suivant :

(S4) Si deux expressions ont la même catégorie syntaxique définie et la même structure argumentale, alors leurs types sémantiques ont la même composante structurelle.

Maintenant que les structures sont regroupées en classes selon le comportement des expressions, il reste à définir le type final de ces expressions lorsque tous les arguments obligatoires sont fournis. Toujours par notre postulat de compositionnalité (P4), nous savons que ce type doit être fonction des types des unités lexicales qui la composent ; or, si une unité lexicale a une structure de type incluant plusieurs arguments, la méthode de calcul fonctionnel permettant de construire son type final après obtention de tous ses arguments doit être fournie avec le type de l'unité elle-même, en suivant notamment les règles de formation des catégories syntaxiques. En effet, par le principe précédent deux expressions de mêmes catégories et structures ont la même structure de type, donc si elles reçoivent en argument une même autre expression, le type des deux nouvelles expressions sera toujours structurellement le même, ce qui signifie qu'il y a également des règles de construction qui ne dépendent que de la structure initiale des types du prédicat et de l'argument. Il y a alors deux possibilités pour gérer ces règles : soit on les considère comme externes aux types, soit on les intègre au sein de leur structure. Sauf que le premier cas impliquerait qu'à chaque composition il faille fouiller la liste des règles pour déterminer laquelle s'applique, ce qui a un certain coût opérationnel ; à l'inverse, stocker cette information au sein de la structure des types permet d'obtenir l'information nécessaire en temps constant : nous estimons que cette économie justifie largement d'adopter cette seconde approche. On en déduit le principe suivant, où la notion de « détermination » couvre ici l'ensemble des règles de constructions nécessaires lors de la composition :

(S5) La composante structurelle du type sémantique d'une expression détermine le type que prend cette expression lorsque tous ses arguments syntaxiques obligatoires sont fournis.

Le dernier principe structurel que nous énonçons ici a pour but de définir un cas limite de type sémantique, le type des propositions. L'intérêt de ce type naît de deux remarques. Premièrement, nous supposons que l'analyse sémantique a pour but de construire et d'étudier des représentations de la signification sous forme de formules logiques. D'autres représentations sont bien sûr possibles, mais leur composante logique est une constante indispensable de toutes ces représentations, ne serait-ce que pour être capable de déterminer si la phrase analysée contient un paradoxe logique ou non. Or, l'analyse logique nécessite de pouvoir discerner les formules logiques bien formées afin de pouvoir les comparer entre elles de manière sûre, ce qui requiert donc un marqueur permettant de regrouper les formules effectivement comparables. Les types sémantiques ont le potentiel de remplir ce rôle, puisque les significations complètes (et donc les formules logiques bien formées) correspondent généralement à des phrases, et que par le principe (S4), toutes les phrases ont le même type. Il est donc possible, et probablement souhaitable, de distinguer un type des propositions logiques.

Deuxièmement, nous supposons que les phrases grammaticalement complètes ne démontrent pas de variabilité ontologique dans leur capacité compositionnelle : les unités lexicales qui sélectionnent des phrases pour argument ne font a priori pas de distinction entre différentes sous-classes de propositions dont certaines seraient absurdemment passées en argument. Par conséquent, le type associé aux phrases est neutre d'un point de vue ontologique, et est par conséquent purement structurel, ce qui justifie d'autant plus son introduction en tant que principe structurel. Dans le principe ci-dessous, nous formulons la définition de ce type comme celui des phrases après résolution de tous les arguments sémantiques supplémentaires, afin de donner comme nous le souhaitons un cas limite unifié : par exemple, une phrase dans le système de Montague a canoniquement le type $s \to t$, or nous souhaitons isoler t, d'où cette précaution additionnelle.

(S6) Il existe un unique type t, appelé *type des propositions*, défini comme le type sémantique

d'une phrase lorsque tous ses arguments sémantiques éventuels sont fournis.

Ce type ainsi défini permet notamment de déterminer assez précisément la structure des types associés à certaines catégories syntaxiques, dont les verbes, qui constituent généralement la tête d'une phrase et déterminent donc sa formation : il s'agit d'une collection de spécification correspondant à l'ensemble de leurs arguments, ainsi que des règles aboutissant à l'unique type t une fois tous les arguments complétés. Ceci conclut donc notre analyse structurelle des types : les principes qui précèdent conditionnent suffisamment ces derniers pour rendre compte de la dynamique compositionnelle du language, tout en laissant assez d'espace à l'élaboration de théories formelles des types différentes.

3.3.2 La composante ontologique

Comme expliqué au début de la présente section, l'introduction de la composante ontologique a pour but de renouer avec la théorie catégorielle de Sommers et d'intégrer cette dernière à la théorie que nous développons ici. Bon nombre des propriétés établies par Sommers lui-même seront en fait préservées, permettant ainsi d'imposer une structure très précise sur l'ensemble des composantes ontologiques. L'addition des principes structurels ayant pour objectif de généraliser la théorie sommersienne aux prédicats non-monadiques, il y a nécessairement une interaction substantielle entre les deux composantes, qui s'incarne particulièrement dans les spécifications des arguments. En effet, la composante ontologique, reprenant le rôle des catégories ontologiques, a pour but de classer les combinaisons selon leur effet au regard de leur absurdité : le critère pour établir une erreur de catégorie doit donc figurer là où la composition est vérifiée, c.-à-d. dans les spécifications. Cela implique deux choses : d'une part, chaque spécification doit contenir un critère ontologique, et d'autre part, toute expression qui peut être sélectionnée comme argument d'une autre expression doit contenir une composante ontologique à confronter à ce critère.

La seule exception à ce système, comme on peut s'en douter au vu du dernier principe de la sous-section précédente, est celui des phrases complètes. En effet, la prédication d'une expression à une phrase entière n'a pas fait l'objet de discussions ni chez Sommers ni chez aucun des philosophes cités — ce qui, combiné à l'utilisation généralisée du simple type des propositions dans divers formalismes, tend à encourager l'idée qu'il n'y a pas de distinction ontologique entre différentes catégories de propositions : ces dernières sont unifiées sous un seul et même type, le type t défini au principe (S6), qui échappe donc au critère ontologique. Nous tentons par conséquent une formalisation de la composante ontologique qui regrouperait tous les constats que nous venons de faire :

(O1) Il existe une composante des types sémantiques, appelée *composante ontologique*, telle que le type de toute expression pouvant être marquée comme argument obligatoire d'une autre expression au sein d'une expression suffisamment connectée, excepté les phrases, possède une telle composante, et telle que les spécifications de tous les arguments syntaxiques obligatoires au sein de la composante structurelle d'un type, excepté les propositions, contiennent des critères quant aux valeurs acceptables de cette composante pour l'argument.

Afin de conserver la séparation entre les composantes, nous imposons que le critère ontologique des spécifications ne fait pas partie de la composante structurelle : la composante ontologique d'un type fournit à la fois les critères ontologiques pour tous les arguments, et la composante interne à l'expression si applicable. Ce double rôle de la composante ontologique contribue à faire ressurgir un fait contre-intuitif : une expression peut avoir une composante

ontologique interne séparée des critères ontologiques de ses arguments. Ce phénomène survient lorsqu'une expression peut être à la fois un prédicat attendant des arguments obligatoires, et un argument pour d'autres expressions. Ceci concerne en particulier les syntagmes verbaux, qui attendent un syntagme nominal sujet mais peuvent être sélectionnés préalablement comme arguments d'adverbes, pour lesquels des erreurs de catégories semblent en effet possible :

(9) a. Marie ramasse lentement un bâton.
b. ?Marie tient lentement un bâton.

S'il est correct de reconnaître le fait qu'il est absurde de se questionner sur la vitesse à laquelle on tient quelque chose, il y a alors bel et bien des catégories de syntagmes verbaux au sens de Sommers. Ce dernier, cependant, n'a pas à ma connaissance traité ce cas, se concentrant davantage sur les noms et syntagmes nominaux et sur les adjectifs ; il s'agit ici d'une généralisation de sa méthode pour évaluer d'autres types de prédications, menant ainsi au troublant résultat que les syntagmes verbaux ont une composante ontologique. Par ailleurs, cette composante est distincte des spécifications de l'argument sujet : dans l'exemple ci-dessus, les verbes *ramasser* et *tenir* partagent les mêmes arguments sujets a priori. [55] La même remarque peut de plus être faite quant à l'argument objet des verbes constituant la tête des syntagmes ci-dessus.

Notons enfin qu'une telle composante ontologique, interne à une expression attendant des arguments supplémentaires, peut tout à fait être perdue lors d'une composition : ainsi, si le syntagme *ramasser un bâton* reçoit un sujet, nous obtenons une phrase, pour laquelle aucun critère ontologique ne subsiste en conformité avec notre premier principe ci-dessus : la composante ontologique du syntagme est oubliée. En revanche, en cas d'application d'un adverbe comme *lentement* au même syntagme, le syntagme résultant hérite a priori de la composante du syntagme initial. Il existe donc une certaine dualité dans le type sémantique des syntagmes verbaux, liée à leur capacité à apparaître à la fois comme prédicats et comme arguments, qui survient directement comme corollaire du principe (O1).

Il convient ensuite de caractériser de plus près le comportement interne des composantes ontologiques : comment sont-elles construites ? Comment sont-elles organisées ? C'est ici que la théorie de Sommers rentre en jeu : comme défini notamment dans [171, 174], la catégorie ontologique d'un prédicat monadique est déterminée par sa portée, qui regroupe toutes les expressions qui peuvent figurer en position d'argument de ce prédicat sans absurdité. Plus précisément, il existe une catégorie ontologique A partagée par tous les prédicats qui ont exactement la même portée ; si l'on prend A comme composante ontologique, cela signifie que le critère ontologique qui spécifie l'argument est représenté par A, et que toutes les expressions arguments doivent appartenir à A, en assimilant A à l'ensemble des expressions figurant dans la portée des prédicats concernés.

On serait alors tenté de dire que A est aussi la composante ontologique de toutes ces expressions acceptables en argument, mais il y a ici une subtilité : supposons en effet qu'il existe un prédicat dont la portée est strictement incluse dans A, et appelons B la catégorie associée. Cette nouvelle catégorie est alors « plus précise », dans la mesure où elle détermine plus finement les combinaisons possibles : toutes les expressions B peuvent être acceptées par des prédicats de

55. Certains cas limites peuvent remettre en question cette affirmation, comme avec *statue* par exemple : y a-t-il une erreur de catégorie dans la phrase « la statue ramasse un bâton » ? Bien que ce soit logiquement impossible dans notre monde, la proximité entre les statues et les êtres animés qu'elles représentent pourrait justifier ce rapprochement ontologique. Remarquons en particulier que si la statue représente précisément une personne en train de ramasser un bâton, la phrase a une signification qui va de soi. Il est néanmoins possible que cela cache une coercion du sujet d'un objet vers une entité vivante ; seule une étude approfondie — et dépassant le cadre de cette étude — pourrait nous permettre de trancher.

catégorie A, mais l'inverse est faux. Il est alors plus pratique de retenir d'une part l'information de la catégorie B, et d'autre part l'inclusion de B dans A : ainsi, lors de l'application d'un prédicat spécifiant la catégorie A à une expression de B, la simple vérification de l'inclusion suffit à conclure que l'application est acceptable. Ceci permet alors de dégager une meilleure vue d'ensemble du système des composantes ontologiques : la composante ontologique dans la spécification d'un prédicat caractérise un ensemble d'arguments acceptables le plus large possible, mais à l'inverse, la composante ontologique interne au type d'un argument potentiel le restreint au groupe le plus petit possible qui soit spécifié par un prédicat, et il existe alors un ordre sur les composantes ontologiques qui rétablit les inclusions entre les ensembles correspondants, et permet de vérfier si un argument est acceptable ou non. Nous résumons alors ces conclusions dans un nouveau principe, faisant du respect de cet ordre une condition nécessaire à la vérification des spécifications, et donc à la bonne formation des types sémantiques au cours de la composition.

(O2) Il existe un ordre partiel sur les composantes ontologiques des types sémantiques tel que les spécifications du type d'un argument sont vérifiées seulement si la composante ontologique de l'argument est plus petite pour cet ordre que la composante ontologique imposée par la spécification correspondante dans le type du prédicat.

On comprend alors que cette relation d'ordre sur les composantes ontologiques préfigure en réalité la relation de sous-typage utilisée dans les théories de types en sémantique lexicale. Grâce au principe ci-dessus, le sous-typage devient donc un impératif explicite de ces théories de types. Mais il est possible d'aller encore plus loin dans l'organisation des composantes ontologiques selon cet ordre, toujours grâce au travail de Sommers. Ce dernier a en effet établit une loi linguistique importante nommée loi de l'inclusion catégorielle, selon laquelle deux catégories ontologiques qui s'intersectent ne peuvent jamais s'intersecter partiellement : l'une est nécessairement contenue dans l'autre (cf. [171] pour une preuve de cette loi). Replacée dans le contexte de notre théorie, cette loi fixe un nouveau principe sur l'ordre des composantes ontologiques des prédicats :

(O3) Si le type d'une expression suffisamment connectée satisfait les spécifications d'argument d'un prédicat donné, dont la composante ontologique est A, ainsi que les spécifications d'argument d'un autre prédicat dont la composante ontologique est B, alors A et B sont comparables pour l'ordre défini au principe (O2).

Ce principe établit notamment que deux composantes ontologiques qui admettent une borne inférieure sont comparables, et donc que l'ensemble des majorants d'une composante ontologique est totalement ordonné. Une autre remarque importante porte sur la taille de l'ensemble des composantes ontologiques. En effet, les types sémantiques ont pour vocation de classer les expressions de la langue, lesquelles sont théoriquement infinies étant donné notre compétence langagière ; on pourrait alors craindre que les types sémantiques soient également en nombre infini. Pourtant, le principe (S4) nous invite au contraire à penser que les composantes structurelles des types, qui correspondent plus ou moins aux catégories syntaxiques, elles-mêmes finies, sont également en nombre fini. Qu'en est-il alors des composantes ontologiques ?

Nous nous appuyons ici sur le fait suivant : un syntagme (nominal ou verbal) porte la même composante ontologique que sa tête, excepté peut-être pour quelques exceptions, comme dans le cas des adjectifs privatifs (cf. [92]). [56] Et si exceptions il y a, il est probable qu'ils soient générés par un nombre fini de mots ou d'expressions : alors, l'ensemble des composantes

56. L'effet de tels adjectifs n'est cependant pas entièrement clair au regard du type. S'il est établi *logiquement* qu'une *arme factice* n'est pas une arme, et donc ne peut pas être utilisé comme tel, il n'est pas certain que son type soit modifié. Il pourrait même être possible que les exceptions supposées n'existent pas, mais en l'absence de

ontologiques est déterminé par les têtes possibles des syntagmes, et par l'ensemble fini des expressions modificatrices s'il y en a. Du fait de la finitude du vocabulaire d'une langue à un instant donné, on en déduit la finitude du nombre de composante ontologiques utilisés en pratique, que nous formalisons dans le principe suivant :

(O4) L'ensemble des composantes ontologiques des types sémantiques est fini.

Appliqué aux conséquences du principe (O3), ceci implique que l'ensemble des majorants d'une composante ontologique est totalement ordonné et fini.

Une dernière propriété de l'ensemble des composantes ontologiques et de l'ordre qui l'équipe porte sur la distinction d'un élément particulier de cet ensemble. À ce stade de notre étude, il devient assez clair que les composantes ontologiques se rapprochent des types des entités dans les théories de types introduites au chapitre 1. Or, nous avions introduit ces types sous la forme d'une hiérarchie dont le type général des entités e est le plus grand élément ; ce qui soulève la question de savoir si un tel équivalent peut être défini dans notre théorie linguistique sans être un simple artéfact formel. Là encore, Sommers et sa théorie catégorielle viennent à notre secours : dans ses travaux, Sommers montre en effet qu'une catégorie ontologique maximale existe, parce qu'elle est incarnée par des prédicats concrets de la langue. Il argue en particulier que le prédicat *exister* admet tous les sujets possibles dans sa portée, peut importe s'ils existent effectivement ou non [171] : *licorne*, *bateau* et même *zelpomir* se retrouvent dans sa portée. D'autres exemples incluent des prédicats comme *(être) intéressant* voire *aimer* au sens large ; en anglais, *exist*, *(be) interesting* et *like* font de même. Si cette observation est correcte, nous avons donc pour ces prédicats une composante ontologique qui inclut toutes les autres, et qui reprend donc le rôle du type e dans les formalismes précédents.

(O5) L'ensemble des composantes ontologiques admet un plus grand élément pour l'ordre défini au principe (O2).

Ce principe implique que l'ensemble des majorants d'une composante ontologique contient ce plus grand élément ; ce qui a pour conséquence que l'ensemble des majorants de toute sous-partie de l'ensemble des composantes admet lui aussi un plus grand élément. À un renversement près de la relation d'ordre, cette propriété définit exactement l'ensemble des composantes ontologiques comme un arbre partiellement ordonné. Toute cette discussion permet alors de justifier la définition 1 : si l'ontologie de types qui y est définie représente bien la composante ontologique des types sémantiques, alors il est nécessaire par les principes qui précèdent qu'elle soit partiellement ordonnée et admette un unique plus grand élément. Notons au passage que cette approche traite les composantes ontologiques comme des types sémantiques dépourvus de composante structurelle — autre que leur composante ontologique qui est égale à eux-mêmes. Il n'est pas nécessaire que les composantes ontologiques soient des types a priori, mais ce choix permet néanmoins de retrouver le lien entre la présente théorie des types et les théories initiales de Russell. Par ailleurs, même si les composantes ontologiques sont traitées comme des types par une implémentation de la théorie, il n'est pas nécessaire qu'il y ait des expressions dont les types soient effectivement parmi ceux-là : les formalismes basés sur le système montagovien n'ont généralement pas accès directement à des expressions dont le type soit celui d'une entité.

Ceci conclut alors notre description des propriétés des composantes ontologiques : nous savons désormais comment elles s'intègrent aux types eux-mêmes, démarquant les emplacements des critères de spécifications et un critère interne si l'expression associée peut être selectionnée ou

preuve ou de contre-exemple, nous nous basons sur l'hypothèse raisonnable qu'il doit en exister, mais en quantité limitée.

non comme argument ; et nous avons établi une structure interne à l'ensemble de ces composantes qui justifie notamment l'introduction d'une hiérarchie de types lexicaux ou des entités, qui sont une implémentations de ces composantes. Au vu de ce constat, nous nous réfèrerons désormais aux types qui implémentent ces composantes comme des *types ontologiques*, les distinguant ainsi des autres types sémantiques parmi lesquels ils s'intègrent alors.

3.3.3 Extension aux phénomènes sémantiques complexes

Afin de compléter notre description des types sémantiques, il convient de nuancer un peu le tableau précédent avec le problème posé par les expressions hétérotypiques, ou objets pointés. En effet, leur existence semble remettre en cause la structure d'arbre si élégamment observée pour ces composantes, puisque de telles expressions sont des contre-exemples évidents au principe (O3). Il s'agit d'une subtilité de la théorie de Sommers qui échappe parfois à la compréhension de ceux qui l'étudient (cf. p. ex. [175]) : les expressions hétérotypiques ne sont pas ontologiques, et n'appartiennent donc pas à la hiérarchie que Sommers décrit comme un arbre. Comment peut-on alors en rendre compte dans notre théorie ? La multiplicité des composantes ontologiques d'une expression donnée résulte, argue Sommers, d'une ambiguïté de cette expression entre plusieurs sens clairement distincts. Cette ambiguïté ne saurait être d'ordre syntaxique puisque les expressions hétérotypiques ne sont pas des homonymies : il y a bien des liens sémantiques entre les différents aspects exprimés. Elle ne peut pas être d'ordre ontologique non plus car cela briserait les règles formalisées par Sommers, établies dans un cadre général qui englobe ces expressions aussi : l'hétérotypie ne remet ainsi pas en cause la règle, mais la règle remet en cause notre manière de la représenter. Par élimination, le seul niveau qui nous permette une modélisation correcte est donc celui des composantes structurelles.

Cela impliquerait par conséquent l'existence d'un niveau structurel supplémentaire, comme un complexe dans lequel plusieurs types différents seraient rassemblés dans un seul. Mais, puisqu'une expression hétérotypique conserve son comportement syntaxique quelles que soient les aspects utilisés, les différents types qui constituent ces complexes ont nécessairement la même structure, par application du principe (S4). Seules varient leurs composantes ontologiques, permettant ainsi de capturer fidèlement le comportement ambigu des expressions associées à ces types. Mais si de tels complexes existent bien, il est alors nécessaire de mener à bien deux tâches : préciser la distribution des composantes ontologiques entre les différents constituants d'une part, et de réviser légèrement la vérification des spécifications lors de l'établissement de relations de prédication d'autre part. Pour cette première tâche, il convient notamment de préciser que les composantes ontologiques de deux constituants d'un type complexe doivent nécessairement différer sur au moins une valeur, soit parmi les critères ontologiques des spécifications, soit pour la composante interne au type, et ce selon que l'ambiguïté est liée au comportement prédicatif de l'expression, ou au contraire à son comportement d'argument. Quant à la seconde tâche, il s'agit de préciser que lors d'une prédication, il doit y avoir accord entre au moins une spécification de l'argument correspondant provenant de l'un des constituants du type du prédicat, et au moins un des constituants du type de l'argument.

Ceci nous amène naturellement à l'énoncé de nouveaux principes pour les composés hétérotypiques, récapitulant les conditions de construction évoquées ci-dessus. Il s'agit d'un principe touchant les deux composantes des types sémantiques à la fois, permettant également de cimenter les interactions entre ces deux aspects.

(H1) Il existe une construction interne des types sémantiques, appelée *type complexe*, constituée d'un regroupement de types sémantiques non-complexes appelés *constituants*, telle que tous les constituants d'un type complexe ont la même composante structurelle et ont des

composantes ontologiques qui diffèrent sur au moins un emplacement commun (que ce soit en termes de spécification d'un argument ou en partie ontologique interne), et telle que les composantes ontologiques qui diffèrent sont deux à deux incomparables pour l'ordre défini au principe (O2).

La construction d'un type complexe n'est donc a priori pas récursive ; il est cependant envisageable qu'un tel type dispose d'une organisation interne plus structurée qu'une simple collection de types. Par ailleurs, il dispose par nature d'une composante structurelle qui est similaire à celle de ces constituents. Enfin, sa composante ontologique est non-spécificiée, exceptée en spécifications des arguments dont les composantes des constituants ne diffèrent pas. Il est néanmoins possible de marquer ces ambiguïtés de diverses manières dans une implémentation des types sémantiques, en exploitant notamment la conservation de la structure : ceci permet notamment de voir un type à la Pustejovsky tel que $p \bullet i \to t$ comme une abréviation pour un complexe réel de la forme $(p \to t) \bullet (i \to t)$. D'un point de vue mathématique, ce principe invite en particulier à considérer un constructeur capable de mettre sur un même plan plusieurs possibilités — par exemple avec un type produit.

Il reste alors à réévaluer les conditions de bonne formation des types. Par souci de simplicité, on assimilera ici les types sémantiques non-complexes à des complexes à un seul constituant, qui héritent des composantes de celui-ci.

(H2) Le type d'une expression suffisamment connectée est bien formé si et seulement si les types des expressions initiales sont bien formés, et le type de l'expression argument possède au moins un constituant qui satisfait les spécifications d'au moins un des constituants du type de l'expression prédicat pour cet argument. Le type de l'expression résultante est alors déterminé par les constituants qui répondent à ces exigences.

Même si l'on ignore a priori si un tel phénomène est effectivement possible en pratique, il est légitime de se demander ce qu'il arrive lorsque au moins deux constituants dans chaque type forment des paires compatibles : lorsque c'est le cas, l'idée la plus raisonnable nous semble être de donner comme type final un complexe regroupant comme constituant les types déterminés par chacunes des paires — ceci étant acceptable puisque le principe (S3) garantit que ces nouveaux types ont également la même composante structurelle.

Un dernier mot autour de l'élaboration de notre théorie concerne la notion de coercion, que nous avons identifiée comme centrale dans notre problématique. Dans le cadre que nous venons de poser, une coercion agit comme un correcteur de type : de façon très générale, il s'agit de traiter une expression dont le type est mal formé comme s'il était en fait bien formé. Mais, comme on l'a vu, une telle correction a des conséquences sémantiques importantes, puisqu'elle conduit à modifier la signification de l'une ou l'autre des sous-expressions qui forment l'origine de l'erreur de catégorie — le plus souvent de l'argument. Comme les types sémantiques forment la plus petite condition nécessaire à toutes les propriétés sémantiques, il est naturel que les coercions se produisent à ce niveau ; mais il est donc essentiel que l'application d'une coercion *laisse une trace*, c.-à-d. marque explicitement son application en tant qu'information pouvant être exploitée par les niveaux d'analyses sémantiques en aval.

À cette fin, nous introduisons les coercions comme un ensemble explicite de fonctions des types sémantiques, qui interviennent donc extérieurement à ceux-ci : fondamentalement, une coercion n'est pas automatiquement déployée par la théorie des types, mais est appelée optionnellement par une implémentation de celle-ci si cela est estimé utile. Ces coercions agissent comme si toutes les spécifications des sous-expressions étaient satisfaites, et se distinguent sur les critères sur lesquelles elles interviennent : on pourra ainsi distinguer des coercions ontologiques qui agissent sur les composantes correspondantes, et des coercions structurelles qui en

font autant ; il est également possible d'imaginer des coercions qui font les deux à la fois. Ces critères font que les coercions ne s'appliquent pas forcément à tous les types, c'est pourquoi le principe ci-dessous en fait des fonctions partielles.

(C) Il existe un ensemble de fonctions partielles des types sémantiques dans eux-mêmes, appelées *coercions*, telles que l'application d'une coercion à un type mal formé renvoie le type bien formé qui aurait été obtenu si toutes les spécifications non satisfaites l'avaient été.

Le détail de la construction des coercions et des relations entre elles est laissé au soin de l'implémantation. Par ailleurs, il est également possible de considérer des coercions pour les expressions hétérotypiques, permettant de convertir un type complexe vers l'un de ces constituants, mais nous estimons là encore qu'il n'appartient pas à la théorie générale d'imposer cela : la sélection de l'aspect correct est en effet décrit par un processus différent de celui des usages créatifs de la langue, et une partie de ce processus est déjà posé dans la présente théorie sous la forme du principe (H2) ; le choix d'assimiler ce processus aux autres coercions est donc dépendant de l'implémentation. Avec cet aspect supplémentaire intégré à notre théorie, nous avons alors terminé d'énoncer les principes essentiels que nous considérons devoir encadrer la construction de théories formelles pour les types sémantiques.

Bilan. Ce chapitre a été l'occasion de formaliser un grand nombre de remarques linguistiques autour de la compositionnalité et de l'erreur de catégorie sous la forme de principes, qui constituent en quelque sorte une méta-théorie des théories de types appliquées à la sémantique. Dans les grandes lignes, cette méta-théorie place les types sémantiques comme une classification des expressions selon leur comportement compositionnel, mais d'une manière sensiblement décorrélée de la syntaxe, de sorte que certaines implémentations pourront estimer la correction du sens d'expressions plus larges, que l'on pourrait qualifier de semi-grammaticales par exemple. Elle établit en outre les types sémantiques comme la plus élémentaire des caractéristiques sémantiques des expressions, et explicite les grandes lignes de leur constitution interne, qui doit inclure une composante structurelle reflétant la structure argumentale des expressions syntaxiques correspondantes, ainsi qu'une composante ontologique qui permet de rendre compte des erreurs de catégorie au cours de la composition. Enfin, elle intègre les bases nécessaires à la représentation des expressions hétérotypiques sous forme de complexes regroupant les types de leurs différents aspects, et ajoute une notion de coercion dont l'emploi effectif reste optionnel par construction.

Les contraintes ainsi énoncées sont malgré tout suffisamment larges pour accomoder la plupart des formalismes évoqués depuis le début de ce texte (cf. chapitre 1), bien que ces derniers dans leur forme actuelle ne les respectent pas toujours complètement : par exemple, le formalisme UTT de Luo et Chatzikyriakidis [35], qui utilise les noms eux-mêmes comme des types ontologiques, n'impose qu'une structure assez faible sur leur hiérarchie, et ne reconnaît pas a priori de type ontologique « maximal ». Notons cependant que la théorie présentée ici n'est pas nécessairement immuable, étant soumise notamment aux variations de points de vue qu'il est possible d'avoir en linguistique. Néanmoins, nous estimons qu'elle est une proposition raisonnable pour guider la construction de théories qui rendront compte efficacement des divers phénomènes sémantiques évoqués dans ces pages. En particulier, nous souhaitons à travers cette théorie mettre l'accent sur les résultats de Fred Sommers, dont la vision des catégories ontologiques nous semble être fondamentale pour la bonne appréhension de l'utilité profonde des types sémantiques.

Une dernière note mérite d'être ajoutée sur le lien entre les types et les représentations sémantiques. Dans leur acception formelle, les types sont généralement considérés comme des

valeurs des représentations logiques, mais la présente théorie les représente comme des traits des expressions syntaxiques. Nous estimons qu'il n'y a pas d'incompatibilité entre les deux approches, bien au contraire : dans sa logique de plus petite condition nécessaire, le type sémantique d'une expression contraint sa représentation logique, et réciproquement, les spécificités du système de représentation ont la possibilité de contraindre les types, notamment au niveau des arguments additionnels par rapport à la syntaxe. Comme le choix d'une représentation va généralement de pair avec la définition d'une théorie de types associée, il suffit donc de s'assurer de l'accord entre cette théorie et les principes de la présente méta-théorie pour obtenir un ensemble de types qui s'appliquent à la fois aux expressions et à leurs représentations ; plaçant ainsi les types sémantiques dans un rôle d'intermédiaire entre les aspects syntaxiques et sémantiques de la langue.

Deuxième partie

Études formelles

Chapitre 4

Éléments de théorie des catégories

Category theory provides means to say what a model of, say predicate logic, should look like. It gives a specification, or a hollow structure, which captures the essentials.

Bart Jacobs [88]

Après avoir dégagé un certain nombre de principes linguistiques des types sémantiques dans la partie I, la présente partie II se donne pour objectif de formaliser ces principes en une théorie mathématique applicable à des systèmes et modèles informatiques. Les types sémantiques, introduits au chapitre précédent comme un postulat d'entités abstraites, seront désormais traités comme des objets mathématiques dont il nous faut expliciter les règles de construction, la manière dont ils se lient aux expressions — elles aussi vues comme des objets formels —, et la manière dont ils intéragissent entre eux lors de la composition de ces expressions. Mais avant d'entrer dans l'examen détaillé de ces règles et de ces comportements, nous estimons nécessaire de proposer une introduction complète à une branche particulière des mathématiques, sur laquelle nous allons abondamment nous reposer dans la suite de cette partie : la *théorie des catégories*.

Nous estimons en effet que la connaissance de la théorie des catégories est globalement moins répandue dans le domaine de la linguistique informatique que d'autres sujets comme les lambda-calculs ou la logique ; c'est pourquoi nous préférons fournir directement ici aux lectrices et lecteurs les clés nécessaires à la compréhension des chapitres suivants, plutôt que de les renvoyer immédiatement à d'autres ouvrages de référence. Prévenons cependant que les définitions et propositions énoncées dans ces pages ne seront pas toutes mobilisées dans les chapitres suivants : certaines notions introduites ici n'ont d'autre but que de servir de fondations à la compréhension de celles qui nous intéressent vraiment, le présent chapitre étant pensé comme une structure cohérente où aucun résultat n'est évoqué par hasard. Dans la section 4.1, nous commencerons par dresser un panorama général de la notion de catégorie, de quelques unes de ses propriétés générales et d'outils qui en permettent la manipulation en tant qu'objet mathématique. La section 4.2 abordera ensuite les notions de limite et colimites, et donnera une liste détaillée de structures internes possibles. Enfin, la section 4.3 proposera une introduction à la théorie des topos, sur laquelle nous baserons la plupart de nos résultats. Les personnes familières avec la théorie générale des catégories pourront facilement se passer de la lecture des deux premières sections ; et si elles sont également familières avec la théorie des topos, elles pourront sans problème s'avancer directement au chapitre 5.

4.1 Premières définitions

Comme expliqué par [113], la théorie des catégories tire son origine de l'observation que de nombreuses propriétés mathématiques peuvent être appréhendées facilement sous une représentation unifiée en termes de diagrammes. L'intuition basique de cette observation s'appuie sur le cas ensembliste : si l'on dispose de plusieurs ensembles et de fonctions entre ces ensembles, il est tout à fait possible de représenter cette situation sous la forme d'un graphe dirigé où les nœuds correspondent aux ensembles et les arêtes aux fonctions. Sur cette simple considération, il devient alors possible de construire des propriétés de plus en plus complexes, mais en même temps de plus en plus générales. Très vite, il s'avère que les applications de la théorie des catégories dépassent le simple cadre ensembliste et permettent de raisonner sur des objets plus abstraits encore ; c'est dans cette optique que nous utiliserons la théorie que nous introduisons ici.

Nous commençons par définir les structures essentielles à la compréhension générale des catégories. Outre la notion de catégorie elle-même, la présente section abordera diverses manipulations permettant de construire d'autres catégories à partir de catégories existantes, quelques propriétés internes, les notions de foncteur, de transformation naturelle et d'adjonction, ainsi que le lemme de Yoneda, offrant par conséquent un permier panorama de la théorie des catégories.

4.1.1 Catégories

Il convient naturellement d'initier cette présentation de la théorie des catégories par l'introduction de sa notion centrale, la *catégorie*. Comme indiqué plus haut, il est possible d'y voir une forme contrainte de graphes, avec une collection de « nœuds » que nous appellerons plutôt des *objets*, et une collection d'« arêtes » que nous appellerons plutôt des *morphismes*[57] ; le tout complété par une notion de composition et par deux lois qui régissent son fonctionnement. La définition suivante résume formellement la constitution d'une catégorie :

Définition 7. *Une* catégorie $\mathcal{C}$ *est la donnée d'une classe d'objets* $\mathsf{obj}(\mathcal{C})$*, d'une classe de morphismes* $\mathcal{C}(A, B)$ *de domaine* A *et de codomaine* B *pour chaque paire d'objets* $A, B \in \mathsf{obj}(\mathcal{C})$*, et d'une loi de composition des morphismes* $\circ$ *envoyant toute paire de morphismes* $f \in \mathcal{C}(A, B)$ *et* $g \in \mathcal{C}(B, C)$ *dont le codomaine du premier correspond au domaine du second vers un nouveau morphisme* $g \circ f \in \mathcal{C}(A, C)$*, telles que :*

- *pour tout objet* $A \in \mathsf{obj}(\mathcal{C})$*, il existe un unique morphisme* $\mathrm{id}_A \in \mathcal{C}(A, A)$*, nommé* identité de A*, qui est neutre pour la composition, c.-à-d. que pour tous* $f \in \mathcal{C}(B, A)$ *et* $g \in \mathcal{C}(A, C)$*,* $\mathrm{id}_A \circ f = f$ *et* $g \circ \mathrm{id}_A = g$ *;*
- *la composition est associative, c.-à-d. que pour tous morphismes* $f \in \mathcal{C}(A, B)$*,* $g \in \mathcal{C}(B, C)$ *et* $h \in \mathcal{C}(C, D)$*, on a l'égalité* $(h \circ g) \circ f = h \circ (g \circ f)$*.*

Il est fréquent dans certaines branches des mathématiques d'avoir des notions définissables de plusieurs façons équivalentes, et la théorie des catégories n'échappe pas à ce constat. En particulier, certaines présentations de la définition des catégories préfèrent introduire une classe indifférenciée des morphismes $\mathsf{mor}(\mathcal{C})$ ainsi que deux applications $\mathrm{dom}, \mathrm{cod} : \mathsf{mor}(\mathcal{C}) \to \mathsf{obj}(\mathcal{C})$ représentant les domaine et codomaine des morphismes, le reste des définitions s'adaptant à cette approche. L'introduction directe des classes de morphismes repartis selon leurs domaines et codomaines a cependant l'avantage de permettre une familiarisation plus rapide avec la notation

57. Le mot *flèche* est fréquemment utilisé en lieu et place de *morphisme* ; la présente thèse adopte la convention de n'utiliser que ce dernier. On notera également que les termes apparentés par le sens que sont *application* et *fonction* seront réservés à leurs usages en théories des classes et des ensembles, respectivement.

correspondante, qui dissimule en réalité un bifoncteur (cf. sous-section 4.1.2) et qui s'avèrera utile pour la notion de topos (cf. section 4.3). Elle permet également le raccourci suivant : en supposant que $\mathcal{C}$ est entendu, on notera $f : A \to B$ pour exprimer le fait que $f \in \mathcal{C}(A, B)$. Faute de mieux, nous empruntons le terme anglais *hom-set* pour désigner ces classes de morphismes entre deux objets.

On remarquera également que la définition précédente fait bel et bien appel à des classes d'objets et de morphismes, de sorte que la théorie des catégories n'est pas nécessairement tributaire de la théorie des ensembles (avec notamment le bénéfice de pouvoir manipuler la théorie des ensembles elle-même comme une catégorie). Néanmoins, le recours à des ensembles peut s'avérer souvent plus pratique, et correspond même mieux à la réalité des catégories considérées. On dira plus particulièrement qu'une catégorie est *localement petite* si tous les *hom-sets* de cette catégorie sont des ensembles ; de même, une catégorie est dite *petite* si les collections de ses morphismes et de ses objets sont des ensembles. Dans toute la suite, nous nous appuierons sur l'hypothèse générale que toutes les catégories considérées sont au moins localement petites, sauf mention contraire.

Exemple 1. *La catégorie* Set *a pour objets les ensembles et pour morphismes les fonctions totales entre ces ensembles. Cette catégorie est localement petite car les fonctions définies d'un ensemble à un autre forment encore un ensemble mais, conformément aux risques du paradoxe de Russell, la collection des ensembles n'est pas un ensemble.*

Exemple 2. *La catégorie* $\mathsf{1}$ *est la catégorie composée d'un seul objet et du seul morphisme identité sur cet objet. La catégorie* $\mathsf{2}$ *est constituée de deux objets* $\{0, 1\}$ *munis de leurs identités respectives, et d'un seul morphisme* $0 \to 1$ *additionnel.*

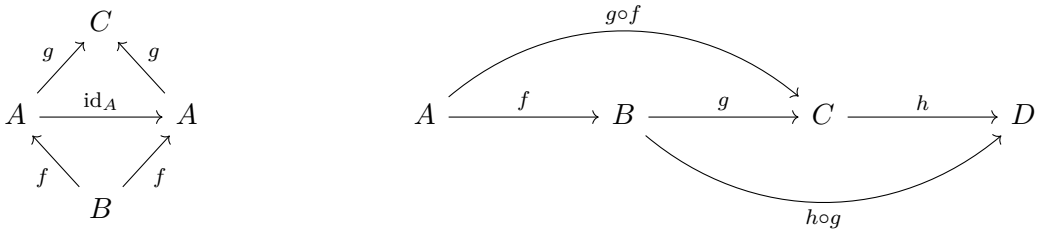
Exemple 3. *Soit* $(S, \leq)$ *un ensemble partiellement ordonné. Cet ensemble peut alors être vu comme une catégorie dont les objets sont les éléments de* S *et telle que pour tous* $a, b \in S$*, il existe un unique morphisme entre* a *et* b *si et seulement* $a \leq b$*. Les identités existent par reflexivité de* $\leq$*, et la composition est donnée par sa transitivité.*

Il existe de nombreuses autres catégories pertinentes qui ont été définies et exploitées ça et là, et certaines d'entre elles seront exploitées dans ces pages. Des listes plus complètes d'exemples peuvent être trouvées dans divers ouvrages de référence, comme par exemple dans [113, §I.2], [132, §I.1], ou encore [177, §4.1].

Diagrammes commutatifs. L'une des tâches les plus communes en théorie des catégories est de vérifier des égalités entre différentes compositions de morphismes, ceci étant à la base de nombre de démonstrations. Afin de faciliter ce travail et de rendre les résultats plus visuels, on utilise couramment une représentation sous forme de graphe telle que nous l'avons évoquée précédemment. Étant donné une catégorie $\mathcal{C}$, on appellera *diagramme* dans $\mathcal{C}$ la représentation d'un graphe orienté dont les nœuds correspondent à des objets de $\mathcal{C}$ et dont les arêtes correspondent à des morphismes, le sens de la flèche allant du domaine vers le codomaine. Il doit être souligné que chaque diagramme ne représente généralement d'une infime partie de la catégorie dont il est issu dans la mesure où seule une sous-collection finie d'objets est représentée, de même qu'un sous-ensemble de morphismes entre chaque paire de ces objets. Par ailleurs, la grande majorité des diagrammes que nous considérerons ici sont finis. Pour la tâche qui nous intéresse, il sera fréquent de discuter d'une propriété particulière de ces diagrammes, que nous introduisons ci-dessous.

Définition 8. *Un diagramme est dit* commutatif *(aussi, qu'il* commute*) si, pour toutes paires d'objets A, B du diagramme, tous les chemins menant de A à B sont égaux, c.-à-d. que les morphismes $A \to B$ obtenus par composition des morphismes représentés par les flèches du diagramme sont égaux entre eux.*

Afin d'illustrer concrètement cette définition, nous donnons ci-dessous des diagrammes commutatifs reprenant les deux lois de la composition telles qu'introduites dans la définition 7, celui de gauche démontrant les propriétés de l'identité et celui de droite illustrant l'associativité :



Le diagramme de gauche illustre en particulier qu'il est possible de représenter plusieurs fois le même objet, ceci permettant notamment de traiter plus facilement les *endomorphismes*, c.-à-d. les morphismes d'un objet dans lui-même. En général, on préférera utiliser la notation $A = A$ plutôt qu'une flèche comme ci-dessus pour représenter l'identité au sein des diagrammes. On notera également que les flèches représentant les composées $g \circ f$ et $h \circ g$ sont superflues dans le diagramme de droite une fois que les notions d'associativité de la composition et de commutativité des diagrammes sont assimilées. Enfin, il convient de préciser qu'il existe une définition plus formelle de la notion de diagramme, utilisant notamment des foncteurs ; nous aurons l'occasion d'en reparler en section 4.1.2.

Constructions basiques de catégories. Avant de nous attaquer aux premières propriétés internes aux catégories, c.-à-d. portant sur des objets et des morphismes au sein de celles-ci, il paraît utile de discuter plusieurs méthodes permettant de construire de nouvelles catégories à partir de catégories déjà connues. Les deux premières méthodes se révèleront certainement familières pour les lecteurs et lectrices habituées aux formalismes des automates ou des graphes, par exemple : il s'agit des notions de produit de catégories et de sous-catégorie, que nous introduisons dans les définitions ci-dessous.

Définition 9. *Soient $\mathcal{C}$ et $\mathcal{D}$ deux catégories. La* catégorie produit *de $\mathcal{C}$ et $\mathcal{D}$, notée $\mathcal{C} \times \mathcal{D}$, est la catégorie ayant pour objets les paires (A, A') avec $A \in \mathsf{obj}(\mathcal{C})$ et $A' \in \mathsf{obj}(\mathcal{D})$, et ayant pour morphismes $(A, A') \to (B, B')$ les paires de morphismes (f, f') avec $f \in \mathcal{C}(A, B)$ et $f' \in \mathcal{D}(A', B')$. L'identité de (A, A') est donnée par la paire $(\mathrm{id}_A, \mathrm{id}_{A'})$, et la composition est donnée par $(g, g') \circ (f, f') = (g \circ f, g' \circ f')$.*

Définition 10. *Soit $\mathcal{C}$ une catégorie. Une* sous-catégorie *$\mathcal{S}$ de $\mathcal{C}$ est la donnée d'une sous-collection $\mathsf{obj}(\mathcal{S})$ de $\mathsf{obj}(\mathcal{C})$, et pour tous $A, B \in \mathsf{obj}(\mathcal{S})$, d'une sous-collection $\mathcal{S}(A, B)$ de $\mathcal{C}(A, B)$, telles que :*

- *pour tout $A \in \mathsf{obj}(\mathcal{S})$, $\mathrm{id}_A \in \mathcal{S}(A, A)$;*
- *pour tous $f \in \mathcal{S}(A, B)$ et $g \in \mathcal{S}(B, C)$, $g \circ f \in \mathcal{S}(A, C)$.*

Il convient de souligner que les contraintes imposées par la seconde définition assurent que toute sous-catégorie est elle-même une catégorie, justifiant son introduction dans cette section. D'autres constructions de catégories, plus complexes, seront introduites plus loin dans ce chapitre (cf. notamment en sous-section 4.2.2).

Une dernière construction simple qu'il nous paraît essentiel de mentionner ici permet également d'évoquer le fait que l'une des grandes forces de la théorie des catégories réside dans sa capacité à offrir une *dualité* dans ses définitions. En effet, les catégories possèdent une dimension directionnelle inhérente à leur définition, qui s'incarne dans les morphismes, et il est courant que chaque définition d'une propriété interne concernant des morphismes en théorie des catégories soit accompagnée d'une version duale qui s'obtient en inversant simplement les domaines et codomaines des morphismes impliqués. Appliquée à la notion de catégorie elle-même, cette dualité donne lieu à la définition suivante :

Définition 11. *Soit $\mathcal{C}$ une catégorie. La* catégorie opposée *ou* duale *de $\mathcal{C}$, notée $\mathcal{C}^{\mathrm{op}}$, est la catégorie définie par $\mathsf{obj}(\mathcal{C}^{\mathrm{op}}) = \mathsf{obj}(\mathcal{C})$ et, pour tous $A, B \in \mathsf{obj}(\mathcal{C})$, par $\mathcal{C}^{\mathrm{op}}(A, B) = \mathcal{C}(B, A)$.*

Autrement dit, $\mathcal{C}^{\mathrm{op}}$ est simplement obtenue en renversant le sens de tous les morphismes dans $\mathcal{C}$. Ponctuellement, la notation en exposant pourra être étendue aux morphismes, de sorte que f^{op} désignera le pendant du morphisme f dans la catégorie opposée. On retiendra alors que $(g \circ f)^{\mathrm{op}} = f^{\mathrm{op}} \circ g^{\mathrm{op}}$ par définition. Cette notion illustre également le fait qu'une catégorie est une construction bien plus abstraite que les exemples classiques ne laissent penser : ainsi, il est bien difficile de donner une autre interprétation aux morphismes de $\mathsf{Set}^{\mathrm{op}}$ que « il y a un morphisme de A dans B pour chaque fonction de B dans A », démontrant qu'un morphisme ne représente pas nécessairement un objet mathématique concret ; et de fait, une remarque similaire s'applique tout aussi bien aux objets.

Propriétés de morphismes. Intéressons-nous maintenant plus en détail aux morphismes et aux propriétés qu'on peut leur définir. La première, certainement l'une des plus importantes, généralise la notion de fonction bijective dans le cas ensembliste. La capacité à reconnaître deux objets comme ayant la même « structure » est en effet essentielle en théorie des catégories, notamment parce que de nombreuses propriétés définissables sur les objets se propagent aux objets partageant cette structure, au travers de la notion d'*isomorphisme*.

Définition 12. *Un morphisme $f : A \to B$ est appelé un* isomorphisme *(ou est dit être* iso*) lorsqu'il existe un morphisme $g : B \to A$ tel que $g \circ f = \mathrm{id}_A$ et $f \circ g = \mathrm{id}_B$. Un tel morphisme est alors noté f^{-1} et est appelé* inverse *de f.*

On dira également que deux objets A et B sont *isomorphes*, noté $A \cong B$, lorsqu'il existe un isomorphisme entre A et B. Précisons par ailleurs que toute identité est iso, avec $\mathrm{id}_A^{-1} = \mathrm{id}_A$. On trouvera fréquemment dans la suite de ce chapitre des propriétés augmentées de la mention « définie à isomorphisme près ». Ceci doit se comprendre comme indiquant que lorsque la propriété est vraie d'un objet A donné, alors elle est également vraie pour tout objet isomorphe à A. Il s'agit par conséquent de la généralisation catégorique de l'unicité. D'autres propriétés des morphismes, présentées ci-dessous, généralisent d'une certaine manière les notions ensemblistes d'injection et de surjection.

Définition 13. *Un morphisme $f : A \to B$ est un* monomorphisme *(ou est* mono*) si, pour tous morphismes $g, h : X \to A$, l'égalité $f \circ g = f \circ h$ implique $g = h$. De même, f est un* épimorphisme *(ou est* épi*) si, pour tous morphismes $g', h' : B \to Y$, l'égalité $g' \circ f = h' \circ f$ implique $g' = h'$.*

On notera également $f : A \rightarrowtail B$ et $g : A \twoheadrightarrow B$ pour indiquer que f et g sont respectivement un monomorphisme et un épimorphisme de A dans B. La relation entre ces propriétés et le cas ensembliste est verifiée au sein de la catégorie Set, dans la mesure où les monomorphismes

de Set sont exactement les fonctions injectives, et les épimorphismes les fonctions surjectives. En revanche, s'il est vrai qu'une fonction est bijective si et seulement si elle est injective et surjective, il n'y a en général pas de résultat équivalent dans les catégories. À moins d'hypothèses supplémentaires (cf. sect. 4.3), nous devrons donc nous contenter de la propriété suivante :

Proposition 1. *Si un morphisme est iso, alors il est mono et épi.*

Retenons donc que la réciproque est généralement fausse : c'est notamment le cas de la catégorie Ring des anneaux unitaires [114, §IV.1]. Les définitions de ces notions de monomorphisme et épimorphisme sont également l'occasion d'illustrer plus concrètement la notion de dualité. On peut en effet vérifier dans la définition 13 que les propriétés sont similaires au sens près des morphismes. On dira alors que monomorphisme et épimorphisme sont des propriétés duales, dans la mesure où l'on peut établir que si f est mono (resp. épi) dans $\mathcal{C}$ alors f^{op} est épi (resp. mono) dans $\mathcal{C}^{\mathrm{op}}$. Cela signifie notamment que toute proposition portant sur les monomorphismes peut être transformée en une proposition sur les épimorphismes en renversant les directions de tous les morphismes, et inversement. Afin d'alléger un peu notre exposé, nous nous contenterons donc d'énoncer uniquement des propriétés sur les monomorphismes plutôt que de répéter les propositions duales. La définition suivante, introduisant une notion plus forte que celle de monomorphisme, sous-tend en conséquence une définition d'une notion plus forte qu'épimorphisme.

Définition 14. *Un morphisme $f : A \to B$ est* split mono *s'il admet un inverse à gauche, c.-à-d. un morphisme $g : B \to A$ tel que $g \circ f = \mathrm{id}_A$.*

Dans cette définition, g est un inverse à gauche de f si et seulement si f est un inverse à droite de g, ce qui signifie en particulier que g lui-même a la propriété duale de *split épi*. Dans pareil cas, on dit que g est une *rétraction* de f, et que f est une *section* de g. Toute section est donc split mono, et toute rétraction est split épi. Notons qu'encore une fois, un même morphisme peut être split mono et split épi sans être iso, car ses rétractions et ses sections peuvent être différentes. Nous concluons cette discussion par quelques propriétés supplémentaires en vrac.

Proposition 2. *Les propriétés suivantes sont vérifiées dans toute catégorie :*

1. *Si f et g sont isos, alors $g \circ f$ est iso et $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$.*
2. *Si f et g sont monos alors $g \circ f$ est mono.*
3. *Si $g \circ f$ est mono alors f est mono.*
4. *Si f est split mono, alors f est mono.*

4.1.2 Foncteurs

Élements incontournables de la théorie des catégories, les foncteurs sont simplement des homomorphismes de catégories. Il s'agit plus précisément de paires d'applications satisfaisant quelques contraintes données comme suit :

Définition 15. *Soient $\mathcal{C}$ et $\mathcal{D}$ deux catégories. Un* foncteur *F de $\mathcal{C}$ dans $\mathcal{D}$ (noté $F : \mathcal{C} \to \mathcal{D}$) est une application qui envoie tout objet $A \in \mathsf{obj}(\mathcal{C})$ vers un objet $FA \in \mathsf{obj}(\mathcal{D})$ et tout morphisme $f \in \mathcal{C}(A, B)$ vers un morphisme $Ff \in \mathcal{D}(FA, FB)$, telle que :*

— *pour tout $A \in \mathsf{obj}(\mathcal{C})$, $F\,\mathrm{id}_A = \mathrm{id}_{FA}$;*

— *pour tous $f : A \to B$ et $g : B \to C$, $F(g \circ f) = Fg \circ Ff$.*

Un foncteur se doit donc de préserver les domaines et les codomaines, ainsi que les identités et les compositions. On parlera également de domaine et de codomaine pour désigner les catégories source et cible du foncteur ; lorsque ces catégories sont identiques, on parlera alors d'*endofoncteur*.

Exemple 4. *Pour toute catégorie $\mathcal{C}$, le* foncteur identité $I_\mathcal{C} : \mathcal{C} \to \mathcal{C}$ *est défini pour tout $A \in \mathsf{obj}(\mathcal{C})$ par $I_\mathcal{C} A = A$, et pour tout $f : A \to B$ par $I_\mathcal{C} f = f$.*

Exemple 5. *L'endofoncteur d'ensemble des parties $\mathcal{P} : \mathsf{Set} \to \mathsf{Set}$ est défini pour tout ensemble A par $\mathcal{P}A = \{U \mid U \subseteq A\}$ et pour toute fonction $f : A \to B$ et tout $U \subseteq A$ par $\mathcal{P}(f)(U) = \{f(x) \mid x \in U\}$. On vérifie aisément que cette définition préserve les identités ainsi que les compositions.*

Dans le cas particulier où l'on a un foncteur $F : \mathcal{C}^{\mathrm{op}} \to \mathcal{D}$ dont le domaine est la catégorie opposée à $\mathcal{C}$, on parle parfois de foncteur *contravariant* de $\mathcal{C}$ vers $\mathcal{D}$, car il est possible de définir F uniquement en termes de morphismes de $\mathcal{C}$ et non de $\mathcal{C}^{\mathrm{op}}$, en renversant le sens des morphismes images de F. Lorsqu'il faudra souligner qu'aucune dualité n'est impliquée dans la définition d'un foncteur, on utilisera le terme de foncteur *covariant*.

Il convient également de noter que les foncteurs peuvent être composés : si l'on dispose de deux foncteurs $F : \mathcal{C} \to \mathcal{D}$ et $G : \mathcal{D} \to \mathcal{E}$, leur composée sera notée $GF : \mathcal{C} \to \mathcal{E}$ et sera simplement définie pour $A \in \mathsf{obj}(\mathcal{C})$ par $GFA = G(FA)$ et pour $f : A \to B$ par $G(Ff) : GFA \to GFB$. Là aussi, il est aisé de vérifier que GF ainsi défini est bien un foncteur.[58] Quelques propriétés supplémentaires définissables sur les foncteurs méritent d'être soulignées, nous les donnons dans la définition suivante.

Définition 16. *Soient $\mathcal{C}$ et $\mathcal{D}$ deux catégories. Un foncteur $F : \mathcal{C} \to \mathcal{D}$ est* fidèle *s'il est injectif sur les morphismes, c.-à-d. si pour toute paire de morphismes $f, g \in \mathcal{C}(A, B)$, l'égalité $Ff = Fg$ implique $f = g$. De même, F est* plein *s'il est surjectif sur les morphismes, c.-à-d. si pour tous $A, B \in \mathsf{obj}(\mathcal{C})$ et tout morphisme $h \in \mathcal{D}(FA, FB)$, il existe un morphisme $f \in \mathcal{C}(A, B)$ tel que $Ff = h$.*

Un foncteur à la fois plein et fidèle est généralement nommé en français *pleinement fidèle*. On notera que ces notions concernent surtout le traitement des morphismes, et ne supposent rien sur la manière dont les foncteurs traitent les objets. Diverses autres propriétés peuvent ainsi être établies sur cette base, et nous en retiendrons une ici : on appellera *plongement* de $\mathcal{C}$ dans $\mathcal{D}$ un foncteur $F : \mathcal{C} \to \mathcal{D}$ fidèle et injectif sur les objets. Cette dernière notion se lie très naturellement à celle de sous-catégorie (cf. définition 10), dans la mesure où une sous-catégorie $\mathcal{S}$ de $\mathcal{C}$ définit de manière évidente un foncteur d'inclusion $\mathcal{S} \to \mathcal{C}$ qui est un cas particulier de plongement. Par ailleurs, si ce foncteur d'inclusion est plein, on parlera de *sous-catégorie pleine*.

Les *hom-sets* comme foncteurs. De même qu'une fonction peut avoir plusieurs arguments, il est possible de définir des foncteurs à plusieurs arguments grâce à la construction de catégories produits (cf. définition 9). En particulier, un foncteur dont le domaine est un produit de deux

58. On peut d'ailleurs vérifier que la composition de foncteurs est associative et que le foncteur identité introduit en exemple 4 en est un neutre. Il est donc possible de définir une catégorie Cat dont les objets sont toutes les petites catégories et les morphismes les foncteurs entre celles-ci. En conséquence du paradoxe de Russell, Cat n'est elle-même pas petite, mais est localement petite. Il existe de même une catégorie CAT des catégories localement petites qui n'est pas localement petite. Précisons au passage que chez certains auteurs, p. ex. [2], le terme de *catégorie* est reservé aux catégories localement petites, tandis que les catégories qui ne le sont pas, comme CAT, sont nommées *quasicatégories*.

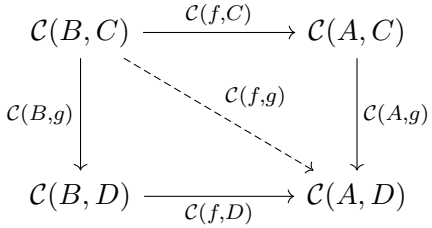
catégories est appelé *bifoncteur*. L'introduction de cette notion est l'occasion idéale de discuter d'un exemple important de bifoncteur omniprésent depuis le début de ce chapitre : le foncteur *hom-set*.

Comme explicité dans la section précédente, les catégories sur lesquelles nous travaillons dans la présente thèse sont supposées être au moins localement petites, ce qui signifie que tous les *hom-sets* considérés ici sont des ensembles. Définir la notion de *hom-set* comme un foncteur n'est possible qu'à cette condition puisque les ensembles, contrairement aux classes, forment effectivement la catégorie Set. Afin de comprendre au mieux comment fonctionne ce foncteur, nous allons étudier séparément ses deux arguments. Nous considérons donc dans la suite une catégorie $\mathcal{C}$ et fixons un objet $X \in \mathsf{obj}(\mathcal{C})$. Pour tous objets A et B de $\mathcal{C}$, $\mathcal{C}(X, A)$ et $\mathcal{C}(X, B)$ sont des ensembles, et il est possible pour tout $f : A \to B$ de définir une fonction $\mathcal{C}(X, f) : \mathcal{C}(X, A) \to \mathcal{C}(X, B)$ par $\mathcal{C}(X, f)(g) = f \circ g$ pour tout $g : X \to A$. On vérifie aisément que $\mathcal{C}(X, \mathrm{id}_A)$ est bien l'identité sur $\mathcal{C}(X, A)$, et que $\mathcal{C}(X, g \circ f) = \mathcal{C}(X, g) \circ (X, f)$. Ceci permet alors d'introduire un foncteur $\mathcal{C}(X, -) : \mathcal{C} \to \mathsf{Set}$ qui décrit le comportement du foncteur *hom-set* par rapport à sa seconde variable.

Le cas de la première variable est légèrement plus subtil. Si $\mathcal{C}(X, f)$ encode la postcomposition avec f, on aimerait pouvoir à l'inverse définir $\mathcal{C}(f, X)$ comme la précomposition avec f. Mais pour que cette définition fonctionne, il est nécessaire de renverser le sens des morphismes lors du passage de f à $\mathcal{C}(f, X)$: on se retrouve alors avec une fonction $\mathcal{C}(B, X) \to \mathcal{C}(A, X)$ telle que $\mathcal{C}(f, X)(h) = h \circ f$ pour tout $h : B \to X$. Comme précédemment, cette définition préserve bien l'identité, mais la composition est elle aussi renversée : on a $\mathcal{C}(g \circ f, X) = \mathcal{C}(f, X) \circ \mathcal{C}(g, X)$. La solution à ces difficultés est de reconnaître dans les propriétés décrites un cas de foncteur contravariant. En effet, en posant $\mathcal{C}(f^{\mathrm{op}}, X) = \mathcal{C}(f, X)$, la conservation du sens des morphismes est retrouvé, ce qui permet *in fine* d'identifier pour la première variable un foncteur contravariant $\mathcal{C}(-, X) : \mathcal{C}^{\mathrm{op}} \to \mathsf{Set}$. L'étude séparée des deux variables nous a finalement conduits à la propriété suivante :

Proposition 3. *Si $\mathcal{C}$ est localement petite, ses* hom-sets *constituent un bifoncteur $\mathcal{C}(-,-) : \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathsf{Set}$.*

Le lien entre ce bifoncteur et l'étude séparée de ses variables s'obtient en remarquant que pour tout objet X et tout morphisme f, on a les égalités $\mathcal{C}(\mathrm{id}_X, f) = \mathcal{C}(X, f)$ et $\mathcal{C}(f, \mathrm{id}_X) = \mathcal{C}(f, X)$. Alors, pour tous $f : A \to B$ et $g : C \to D$, le diagramme commutatif suivant, qui est dans Set, permet de retrouver la définition attendue de $\mathcal{C}(f, g) : \mathcal{C}(B, C) \to \mathcal{C}(A, D)$:



Transformations naturelles. De par leur définition comme applications entre catégories, les foncteurs respectent eux-mêmes une structure formelle particulière, et il est possible de définir des homomorphismes de foncteurs, c.-à-d. des applications qui préservent leur structure intrinsèque. De telles applications sont nommées *transformations naturelles* et forment un concept central en théorie des catégories, dans la mesure où il est généralement considéré que donner une définition mathématique à la notion de transformation naturelle a été la motivation première à l'introduction de la théorie des catégories par Eilenberg et MacLane [53].

Définition 17. *Soient $\mathcal{C}$ et $\mathcal{D}$ deux catégories, et F et G deux foncteurs $\mathcal{C} \to \mathcal{D}$. Une* transformation naturelle *α de F vers G est une application associant à tout $X \in \mathsf{obj}(\mathcal{C})$ un morphisme $\alpha_X : FX \to GX$ dans $\mathcal{D}$ de telle sorte que, pour tout morphisme $f : X \to Y$ dans $\mathcal{C}$, le diagramme suivant commute :*

$$\begin{array}{ccc} FX & \xrightarrow{\alpha_X} & GX \\ {\scriptstyle Ff}\downarrow & & \downarrow{\scriptstyle Gf} \\ FY & \xrightarrow{\alpha_Y} & GY \end{array}$$

La notation $\alpha : F \to G$ est utilisée pour énoncer que α est une transformation naturelle de F vers G. Pour tout $X \in \mathsf{obj}(\mathcal{C})$, le morphisme α_X est appelé *composante de α en X* et l'on dira qu'une transformation naturelle est un *isomorphisme naturel* si toutes ses composantes sont des isomorphismes. De même que pour les foncteurs, il existe pour tout foncteur F une transformation naturelle identité $\iota_F : F \to F$ définie par $(\iota_F)_X = \mathrm{id}_{FX}$ pour tout objet X ; et si $\alpha : F \to G$ et $\beta : G \to H$ sont deux transformations naturelles, la composée $\beta \circ \alpha$ est définie pour tout X par $(\beta \circ \alpha)_X = \beta_X \circ \alpha_X$.[59] Enfin, si $\alpha : F \to G$ est une transformation naturelle avec $F, G : \mathcal{C} \to \mathcal{D}$, et si $H : \mathcal{E} \to \mathcal{C}$ et $H' : \mathcal{D} \to \mathcal{F}$ sont deux autres foncteurs, on définit les transformations $\alpha H : F \circ H \to G \circ H$ et $H'\alpha : H' \circ F \to H' \circ G$, respectivement, par $(\alpha H)_X = \alpha_{HX}$ pour tout $X \in \mathcal{E}$, et par $(H'\alpha)_Y = H'(\alpha_Y)$ pour tout $Y \in \mathcal{F}$. Là encore, cette notion de transformation naturelle permet de définir d'autres propriétés importantes, en particulier les adjonctions et le lemme de Yoneda.

Adjonctions. Avant d'aller plus loin, rappelons que nous supposons par défaut ne manipuler que des catégories au moins localement petites ; cette hypothèse générale s'appliquera à toutes définitions et résultats qui suivent de façon évidente, puisque la plupart impliquent l'utilisation de foncteurs *hom-set*.

Définition 18. *Soient $\mathcal{C}$ et $\mathcal{D}$ deux catégories, et soient deux foncteurs $L : \mathcal{C} \to \mathcal{D}$ et $R : \mathcal{D} \to \mathcal{C}$. On dira que L et R forment une paire de* foncteurs adjoints*, noté $L \dashv R$, s'il existe un isomorphisme naturel entre les foncteurs* hom-set *$\mathcal{C}(-, R(-)) : \mathcal{C}^{\mathrm{op}} \times \mathcal{D} \to \mathsf{Set}$ et $\mathcal{D}(L(-), -) : \mathcal{C}^{\mathrm{op}} \times \mathcal{D} \to \mathsf{Set}$.*

Dans le cas où $L \dashv R$ est vérifié, on dira aussi que L est l'*adjoint à gauche* de R, et que R est l'*adjoint à droite* de L. Cette définition revient à dire que $L \dashv R$ si pour tous $A \in \mathsf{obj}(\mathcal{C}^{\mathrm{op}})$ et $B \in \mathsf{obj}(\mathcal{D})$, on a une bijection $\Phi_{A,B} : \mathcal{D}(LA, B) \cong \mathcal{C}(A, RB)$ dans Set, de telle sorte que, pour tous $f : A' \to A$ dans $\mathcal{C}$ et $g : B \to B'$ dans $\mathcal{D}$, le diagramme suivant commute :

$$\begin{array}{ccc} \mathcal{D}(LA, B) & \xrightarrow{\mathcal{D}(Lf,g)} & \mathcal{D}(LA', B') \\ {\scriptstyle \Phi_{A,B}}\downarrow\wr & & \wr\downarrow{\scriptstyle \Phi_{A',B'}} \\ \mathcal{C}(A, RB) & \xrightarrow{\mathcal{C}(f,Rg)} & \mathcal{C}(A', RB') \end{array}$$

59. On observera une fois de plus que la transformation identité est neutre pour cette définition de la composition, et que cette dernière est par définition associative. Si $\mathcal{C}$ et $\mathcal{D}$ sont deux catégories, il est ainsi possible de définir la catégorie $\mathcal{D}^{\mathcal{C}}$ ayant pour objets les foncteurs $\mathcal{C} \to \mathcal{D}$ et pour morphismes les transformations naturelles entre ceux-ci.

Ces bijections génèrent également des transformations naturelles qui déterminent entièrement l'adjonction en termes de triangles commutatifs. On obtient ces transformations en choisissant dans l'un ou l'autre *hom-set* les deux mêmes objets en domaine et codomaine, images de l'un des deux foncteurs adjoints.

Définition 19. *Soit $L \dashv R$ une adjonction. L'*unité *de cette adjonction est la transformation naturelle $\eta : I_{\mathcal{C}} \to RL$ définie pour tout $X \in \mathsf{obj}(\mathcal{C})$ par $\eta_X = \Phi_{X,LX}(\mathrm{id}_{LX}) : X \to RLX$. De même, sa* co-unité *est la transformation naturelle $\varepsilon : LR \to I_D$ définie pour tout $Y \in \mathsf{obj}(\mathcal{D})$ par $\varepsilon_Y = \Phi^{-1}_{RY,Y}(\mathrm{id}_{RY}) : LRY \to Y$.*

La présentation de ces transformations comme déterminant l'adjonction tient à leur *universalité*, selon une idée que nous retrouverons dans la section 4.2. Dans le cas de l'unité, cette propriété s'obtient en remarquant que pour tout morphisme $f : A \to RB$ dans $\mathcal{C}$, son adjoint $h : LA \to B$ obtenu par la bijection Φ est également l'unique morphisme tel que le triangle à gauche ci-dessous commute. En effet, $Rh \circ \eta_A = Rh \circ \Phi_{A,LA}(\mathrm{id}_{LA}) \circ \mathrm{id}_A$, cette dernière composition correspondant à $(\mathcal{C}(\mathrm{id}_A, Rh) \circ \Phi_{A,LA})(\mathrm{id}_{LA})$. Par commutativité du diagramme précédent, ceci revient à $(\Phi_{A,B} \circ \mathcal{D}(L\mathrm{id}_A, h))(\mathrm{id}_{LA}) = \Phi_{A,B}(h) = f$ comme attendu. Ceci ancre la vision de η_A comme « le plus petit » morphisme de A vers un objet de la forme RX, dans le sens où tout autre morphisme de ce genre est décomposable en terme de η_A composé avec un autre morphisme. Et, comme on peut s'y attendre, une propriété duale tient également pour la co-unité, de telle sorte que pour tout $h : LA \to B$ son adjoint $f : A \to RB$ fait commuter le diagramme de droite ci-dessous.

On notera pour finir la propriété suivante, qui s'obtient aisément en choisissant comme morphismes en diagonale dans les diagrammes ci-dessus des morphismes identités.

Proposition 4. *Soit $L \dashv R$ une paire de foncteurs adjoints et η et ε l'unité et la co-unité correspondantes. Alors, $\varepsilon L \circ L\eta = \iota_L$, et $R\varepsilon \circ \eta R = \iota_R$.*

Lemme de Yoneda. Ce lemme, attribué au mathématicien japonais Nobuo Yoneda, est souvent présenté comme l'un des résultats les plus importants de la théorie des catégories, même si son appréhension et la compréhension profonde de ses conséquences peuvent se réveler longues et difficiles. L'idée sous-jacente est grossièrement que chaque construction mathématique générale correspond à un certain foncteur de codomaine Set, et que cette propriété peut être exploitée pour prouver des résultats importants sur ces constructions. Cette correspondance entre (certains) objets et foncteurs est appelée *représentabilité*, et s'énonce formellement comme suit. Une fois de plus et comme depuis le début, les catégories sont supposées localement petites.

Définition 20. *Soit $\mathcal{C}$ une catégorie. Un foncteur $F : \mathcal{C} \to \mathsf{Set}$ est dit* représentable *s'il existe un objet R de $\mathcal{C}$ tel que F est naturellement isomorphe à $\mathcal{C}(R, -)$.*

Si un tel objet R existe effectivement, on l'appelle alors *représentant* de F. De plus, si $\varphi : \mathcal{C}(R, -) \to F$ est l'isomorphisme naturel associé, on appellera *représentation* de F la paire (R, φ). Avant de poursuivre sur le lemme proprement dit, il est encore nécesssaire de fixer quelques bases supplémentaires décrites ci-dessous.

Définition 21. *Soit $\mathcal{C}$ une catégorie localement petite. Un* préfaisceau *sur $\mathcal{C}$ est un foncteur $\mathcal{C}^{\mathrm{op}} \to \mathsf{Set}$.*

Comme le domaine d'un préfaisceau $F : \mathcal{C}^{\mathrm{op}} \to \mathsf{Set}$ est une catégorie opposée, il est représentable s'il existe un représentant R tel que F soit isomorphe à $\mathcal{C}^{\mathrm{op}}(R, -) = \mathcal{C}(-, R)$. Il est ainsi fréquent de manipuler des foncteurs *hom-set* contravariants lorsqu'on traite de la représentabilité des préfaisceaux. Ces derniers, comme on peut s'y attendre, interviennent de façon incontournable dans l'énoncé du lemme de Yoneda. Si $\mathcal{C}$ est une catégorie, sa *catégorie de préfaisceaux* est $\mathsf{Set}^{\mathcal{C}^{\mathrm{op}}}$, la catégorie des foncteurs $\mathcal{C}^{\mathrm{op}} \to \mathsf{Set}$ avec les transformations naturelles pour morphismes. Un cas notable de préfaisceau est le foncteur *hom-set* lorsque son objet en codomaine est fixé : ceci permet alors de construire un foncteur $\mathcal{Y} : \mathcal{C} \to \mathsf{Set}^{\mathcal{C}^{\mathrm{op}}}$ défini pour $X \in \mathsf{obj}(\mathcal{C})$ par $\mathcal{Y}(X) = \mathcal{C}(-, X)$ et pour $f \in \mathcal{C}(X, Y)$ par $\mathcal{Y}(f) = \mathcal{C}(-, f)$ dont la définition est directe d'après la proposition 3. Ce foncteur $\mathcal{Y}$ est appelé *plongement de Yoneda* et est pleinement fidèle, ce qui est une conséquence du lemme de Yoneda que nous sommes enfin prêts à énoncer.

Lemme 1 (Yoneda)**.** *Soient $\mathcal{C}$ une catégorie et $P : \mathcal{C}^{\mathrm{op}} \to \mathsf{Set}$ un préfaisceau. Pour tout $X \in \mathsf{obj}(\mathcal{C})$, il existe une bijection*

$$\mathsf{Set}^{\mathcal{C}^{\mathrm{op}}}(\mathcal{Y}(X), P) \cong PX$$

envoyant toute transformation naturelle $\alpha : \mathcal{Y}(X) \to P$ sur $\alpha_X(\mathrm{id}_X)$.

Comme annoncé plus haut, il s'obtient en corollaire de ce lemme que $\mathcal{Y}$ est pleinement fidèle. En conséquence, on notera que $\mathcal{Y}$ *reflète* les isomorphismes, ce qui signifie que pour tous objets X et Y, on a $\mathcal{Y}X \cong \mathcal{Y}Y$ si et seulement si $X \cong Y$. Cette propriété peut être exploitée pour prouver un isomorphisme entre deux constructions catégoriques générales en construisant plus simplement une bijection dans Set.

4.2 Propriétés universelles, limites et autres constructions

La section précédente aura permis de poser les bases de la théorie des catégories, avec l'introduction de tous les outils permettant de manipuler des catégories en tant qu'entités mathématiques. Cependant, nous avons vu que les catégories ont une structure interne où d'autres propriétés sont possibles, à l'instar des monos, épis et isomorphismes. Dans la présente section, nous allons nous intéresser à d'autres propriétés internes, portant notamment sur des constructions d'objets particuliers. Ces derniers ont en commun un comportement qui les posent comme des « extrema » parmi les objets qui ont les mêmes propriétés, dans le même esprit que le comportement observé précedemment de l'unité et de la co-unité d'une adjonction. Ce comportement est plus généralement connu sous le terme de *propriété universelle*, et peut se construire formellement à l'aide de la notion de *limite*. Nous aborderons ensuite d'autres constructions utiles qui n'utilisent pas cette notion.

4.2.1 Limites et colimites

Nous commençons ici par introduire formellement la construction des limites et de leur définition duale, les *colimites*. Une fois acquise la méthode générale, nous étudierons plus en détail certains cas particuliers de limites et de colimites. Mais avant de pouvoir faire tout cela, la première étape est donnée par la définition suivante :

Définition 22. *Soient $\mathcal{C}$ et $\mathcal{I}$ deux catégories. Un* diagramme de forme $\mathcal{I}$ *dans $\mathcal{C}$ est un foncteur $\mathcal{I} \to \mathcal{C}$.*

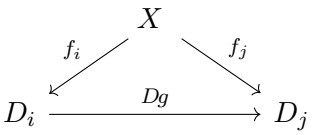
Il peut paraître étrange d'introduire cette notion de diagramme alors qu'elle semble complètement redondante avec la notion de foncteur. La raison de ce choix tient à l'intention qui est placée dans cette nouvelle notion : même si ce n'est pas une obligation, on choisira le plus souvent un domaine $\mathcal{I}$ petit, voire fini, et étant donné un diagramme $D : \mathcal{I} \to \mathcal{C}$, on préfèrera écrire D_i plutôt que $D(i)$ pour $i \in \mathsf{obj}(\mathcal{I})$. On dira aussi d'un diagramme $\mathcal{I} \to \mathcal{C}$ qu'il est *indexé* par $\mathcal{I}$.

Cette définition de diagramme est alors cohérente avec la notion de diagramme commutatif que nous avions informellement introduite en section 4.1.1. En effet, un graphe orienté fini est aisément convertible en une catégorie en prenant ses nœuds pour objets, ses arêtes pour morphismes de bases, et en complétant les morphismes par des identités et une loi de composition intuitive, suivant les chemins du graphe. Une telle catégorie est alors finie, et peut servir de forme pour un diagramme dans n'importe quelle catégorie ; la commutativité du diagramme obtenu découle immédiatement du fait que le diagramme est un foncteur. Il est alors légitime de s'appuyer sur le graphe de départ pour représenter un tel diagramme.

Étant donné une catégorie $\mathcal{I}$, pour tout objet X d'une autre catégorie $\mathcal{C}$, le *foncteur constant* $\dot{X}_{\mathcal{I}} : \mathcal{I} \to \mathcal{C}$ est le foncteur qui envoie tout objet de $\mathcal{I}$ sur X, et tout morphisme de $\mathcal{I}$ sur id_X. Lorsque la catégorie $\mathcal{I}$ est entendue, nous nous permettrons de passer l'indice sous silence en notant simplement $\dot{X}$. De manière évidente, cette notation peut s'étendre en un plongement $\dot{-} : \mathcal{C} \to \mathcal{C}^{\mathcal{I}}$, appelé *foncteur diagonal*, qui envoie chaque objet sur son foncteur constant et chaque morphisme f sur la transformation naturelle constante dont les composantes sont toutes égales à f. Ces foncteurs constants sont au cœur de la définition suivante.

Définition 23. *Soit $D : \mathcal{I} \to \mathcal{C}$ un diagramme. Un* cône *de sommet X sur D est une transformation naturelle $\dot{X} \to D$. Par dualité, un* cocône *de nadir X sous D est une transformation naturelle $D \to \dot{X}$.*

De façon plus concrète, étant donné une forme $\mathcal{I}$ finie, un cône $f : \dot{X} \to D$ sur D sera la donnée d'un objet $X \in \mathsf{obj}(\mathcal{C})$ et d'une famille de morphismes $(f_i)_{i \in \mathsf{obj}(\mathcal{I})}$ avec $f_i : X \to D_i$ pour tout i, telles que pour tous indices i et j et morphisme $g : i \to j$ dans $\mathcal{I}$, le triangle suivant commute :



Et bien entendu, un cocône sous D est la donnée duale, c.-à-d. un cône sur le foncteur $D^{\mathrm{op}} : \mathcal{I}^{\mathrm{op}} \to \mathcal{C}^{\mathrm{op}}$ tel que $D_i^{\mathrm{op}} = D_i$ et $D^{\mathrm{op}} f^{\mathrm{op}} = (Df)^{\mathrm{op}}$. Ces notions de cône et de cocône peuvent facilement être confondues, aussi, suivant la terminologie utilisée par [151], nous glissons une indication sur le sens des flèches en parlant de cône *sur* et de cocône *sous* un diagramme donné.

Nous avons dès lors toutes les bases nécessaires à l'établissement de notre notion de limite. Comme sous-entendu plus haut, cette notion va permettre de généraliser l'application de la notion de propriété universelle, qui pose certains objets comme des extrêmes parmi des objets de même propriétés. Les « mêmes propriétés » en question sont en fait des cônes de même forme ayant lesdits objets pour sommets, et la relation d'ordre est donnée par l'existence de morphismes entre les sommets à travers lesquels se factorisent les composantes du cône. La définition suivante développe ces idées de manière plus formelle.

Définition 24. *Soient $D : \mathcal{I} \to \mathcal{C}$ un diagramme. Un cône $f : \dot{X} \to D$ est dit* universel *sur D si pour tout autre cône $g : \dot{Y} \to D$, il existe un unique morphisme $h : Y \to X$ tel que $f_i \circ h_i = g_i$ pour tout $i \in \mathsf{obj}(\mathcal{I})$. Le sommet d'un cône universel sur D est appelé* limite *de D et noté* $\lim D$.

Par dualité, un cocône $f : D \to \dot{X}$ est universel sur D s'il existe pour tout cocône g de sommet Y un morphisme $h : X \to Y$ tel que $h_i \circ f_i = g_i$ pour tout i, et le sommet d'un tel cocône est nommé colimite *de D, noté* $\operatorname{colim} D$.

Ces notions de limite et de colimite comptent parmi les premières dans le présent exposé à être définies à isomorphisme près. Pour rappel, cela signifie que pour un diagramme D de limite $\lim D$, tout objet isomorphe à $\lim D$ est également limite de D. Au travers de ces isomorphismes, le choix de l'objet précis qui sera utilisé comme limite n'a généralement aucune influence, aussi est-il courant en théorie des catégories de considérer, par abus de langage, un objet $\lim D$ comme *la* limite de D.

Cette propriété de définition à isomorphisme près se substitue néanmoins à l'idée d'unicité telle qu'on peut la retrouver dans d'autres branches des mathématiques. Pour mieux comprendre pourquoi c'est le cas, supposons un diagramme D ayant deux cônes universels $f : \dot{X} \to D$ et $f' : \dot{X}' \to D$. Comme f est universel, il existe un unique $g : X' \to X$ tel que $f_i \circ g = f'_i$ pour tout i, et comme f' est universel, il existe aussi un unique $h : X \to X'$ tel que $f'_i \circ h = f_i$ pour tout i. Comme f est universel, il satisfait aussi cette propriété vis-à-vis de lui-même, c.-à-d. que id_X est l'unique morphisme à travers lequel les f_i se factorisent vers eux-mêmes. Mais c'est également le cas de $g \circ h$ puisque $f_i \circ g \circ h = f'_i \circ h = f_i$ pour tout i. Par unicité, on a donc $g \circ h = \mathrm{id}_X$. Un raisonnement similaire établit $h \circ g = \mathrm{id}_{X'}$, ce qui permet de conclure que g et h sont inverses l'un de l'autre et donc que X et X' sont isomorphes.

Objets initiaux et terminaux. Le reste de cette section sera consacré à la description de cas particuliers de limites et de colimites de grand intérêt en théorie des catégories. Il n'est pas rare pour des modèles catégoriques d'avoir recours à des catégories où certains types de limites sont constructibles dans tous les cas, assurant ainsi la disponibilité de tous les objets nécessaires au modèle. Pour aborder notre premier cas, nous définissons la *catégorie vide* $\mathsf{0}$ comme la catégorie dont les classes d'objets et de morphismes consistent en l'ensemble vide. Pour toute catégorie $\mathcal{C}$, il n'existe alors qu'un unique diagramme vide $\mathsf{0} \to \mathcal{C}$, et les cônes et les cocônes sur ce diagramme sont essentiellement réduits à des objets de $\mathcal{C}$.

Définition 25. *Soit $\mathcal{C}$ une catégorie. Un* objet terminal *dans $\mathcal{C}$ est une limite du diagramme vide. De même, un* objet initial *dans $\mathcal{C}$ est une colimite du diagramme vide.*

Comme ce sera le cas pour toutes limites et colimites, ces notions sont définies à isomorphisme près, et l'abus de langage fixant un objet précis dans chaque cas s'applique également. On utilisera la notation $1_{\mathcal{C}}$ pour désigner l'objet terminal de $\mathcal{C}$, qu'on réduira à 1 si $\mathcal{C}$ est entendu ; l'objet initial sera de même noté $0_{\mathcal{C}}$ ou simplement 0. Comme les cônes et cocônes sur le diagramme vide sont isomorphes aux objets de $\mathcal{C}$, les propriétés respectives de ces limites et colimites se traduisent par l'idée que 0 et 1 sont tels que pour tout $A \in \mathsf{obj}(\mathcal{C})$ il existe un unique morphisme $0_A : 0 \to A$, et un unique morphisme $!_A : A \to 1$, respectivement.

Cette présentation alternative aurait tout aussi bien pu être donnée telle quelle comme définition des notions d'objets initial et terminal. Il s'avère en fait que ces deux notions et celles de limites et colimites sont interdéfinissables ; nous avons choisi ici de définir les premières à partir des secondes, mais l'inverse aurait pu être possible. En effet, étant donné un diagramme $D : \mathcal{I} \to \mathcal{C}$ quelconque, la catégorie des cônes sur D peut être définie comme ayant pour objet les cônes sur D et pour morphisme $f \to g$ entre deux cônes $f : \dot{X} \to D$ et $g : \dot{Y} \to D$ les morphismes $h : X \to Y$ dans $\mathcal{C}$ tels que $h \circ f_i = g_i$ pour tout i. Alors, un cône universel sur D peut être défini comme un objet terminal dans la catégorie des cônes sur D, ce qui permet alors

d'obtenir $\lim D$. De même, un cocône universel sur D sera un objet initial dans la catégories des cocônes sous D.

Exemple 6. *Dans* Set*, l'ensemble vide est l'objet initial, et tous les singletons sont des objet terminaux.*

Par analogie avec le cas particulier de Set, dans toute catégorie $\mathcal{C}$ ayant un objet terminal 1 on appellera un morphisme $1 \to A$ *élément global* de l'objet A. Dans Set, les éléments globaux de A correspondent exactement aux élements de A au sens ensembliste ; en particulier, $\mathsf{Set}(1, A) \cong A$. Il convient de souligner que cette propriété n'est pas valable dans toutes les catégories ayant un objet terminal : certaines formes d'objets construits sur des ensembles peuvent contenir des éléments inaccessibles comme éléments globaux, c'est le cas notamment des catégories d'ensembles équipés d'actions d'un monoïde qui n'est pas un groupe [75, §5.4]. Dans certaines circonstances qui seront développées plus loin dans ces pages, ces objets et les morphismes qui s'y rapportent peuvent avoir de nombreuses propriétés utiles ; mais dans un cadre général, nous nous contenterons de la propriété suivante.

Proposition 5. *Tout morphisme* $1 \to A$ *est mono. De même, tout morphisme* $A \to 0$ *est épi.*

Produits et coproduits. Une hypothèse nécessaire à la construction de ces limites est la notion de *discrétion* : une catégorie est *discrète* si les seuls morphismes qu'elle contient sont des identités. En particulier, tout ensemble peut être vu comme une catégorie discrète.

Définition 26. *Un* produit *est une limite d'un diagramme indexé par une catégorie discrète. De même, un* coproduit *est une colimite d'un tel diagramme.*

Si $\mathcal{I}$ est la catégorie discrète à deux objets, alors un diagramme D indexé par $\mathcal{I}$ est simplement une paire d'objets dans une catégorie $\mathcal{C}$. Soient A et B les objets de $\mathcal{C}$ sélectionnés par D ; alors, s'ils existent, le produit sur D sera noté $A \times B$ et le coproduit sur D sera noté $A + B$. Ces notations pourront être étendues à des petits nombres d'objets. Lorsque $\mathcal{I}$ est trop grand ou infini dénombrable, le produit sur le diagramme $D : \mathcal{I} \to \mathcal{C}$ sera noté $\prod_{i\in\mathcal{I}} D_i$ et le coproduit sur D sera noté $\coprod_{i\in\mathcal{I}} D_i$. Les composantes du cône universel associé au produit sont appelées *projections* et traditionnellement notées $\pi_i : \prod_{j\in\mathcal{I}} D_j \to D_i$ pour tout i ; celles associées au cocône universel pour le coproduit sont nommées *injections* et souvent notées $\iota_i : D_i \to \coprod_{j\in\mathcal{I}} D_j$. Contrairement à ce que cette dernière appellation pourrait laisser à penser, les injections du coproduit ne sont pas nécessairement des monomorphismes, de même d'ailleurs que les projections ne sont pas toujours épis.

Par ailleurs, si X est le sommet d'un cône sur D de composantes $(f_i)_{i\in\mathcal{I}}$, l'unique morphisme $X \to \prod_{i\in\mathcal{I}} D_i$ sera noté $\langle f_i\rangle_{i\in\mathcal{I}}$ (et $\langle f_1, f_2\rangle$ dans le cas binaire), et si Y est le nadir d'un cocône sous D de composantes $(g_i)_{i\in\mathcal{I}}$, l'unique morphisme $\coprod_{i\in\mathcal{I}} D_i \to Y$ sera noté $[g_i]_{i\in\mathcal{I}}$ (et $[g_1, g_2]$ dans le cas binaire). Enfin, si $C : \mathcal{I} \to \mathcal{C}$ est un autre diagramme indexé par $\mathcal{I}$ tel qu'il existe un morphisme $h_i : C_i \to D_i$ pour tout i, on posera :

$$\prod_{i\in\mathcal{I}} h_i = \langle h_i \circ \pi_i^C\rangle_{i\in\mathcal{I}} : \prod_{i\in\mathcal{I}} C_i \to \prod_{i\in\mathcal{I}} D_i$$
$$\coprod_{i\in\mathcal{I}} h_i = [\iota_i^D \circ h_i]_{i\in\mathcal{I}} : \coprod_{i\in\mathcal{I}} C_i \to \coprod_{i\in\mathcal{I}} D_i$$

où π_i^C sont les projections du produit sur C et ι_i^D les injections du coproduit sur D. Dans le cas binaire, on notera ces morphismes $f_1 \times f_2$ et $f_1 + f_2$, respectivement.

Exemple 7. *Les produits dans* Set *correspondent aux produits cartésiens sur les ensembles. Quant aux coproduits, ils correspondent dans* Set *à des unions disjointes d'ensembles.*

Les produits et coproduits étant définis à isomorphisme près comme toutes limites et colimites, il est possible d'obtenir des preuves d'isomorphismes avancés entre ces différentes constructions. Nous rassemblons ces cas d'isomorphismes dans la proposition ci-dessous. Pour celle-ci et les suivantes, on supposera implicitement que les propriétés s'appliquent dès que les limites et colimites concernées existent dans la configuration attendue.

Proposition 6. *Les propriétés suivantes sont vraies dans toute catégorie :*

1. *Le produit et le coproduit sont associatifs, c.-à-d. que pour tous objets A, B et C, on a :*

$$(A \times B) \times C \cong A \times (B \times C)$$
$$(A + B) + C \cong A + (B + C)$$

2. *Le produit et le coproduit sont commutatifs, c.-à-d. que pour tous objets A et B, on a $A \times B \cong B \times A$ et $A + B \cong B + A$.*

L'associativité du produit et l'isomorphisme comme mécanisme d'unicité permettent d'introduire des objets indifférenciées $A \times B \times C$ et $A + B + C$, qui rejoignent les notations $\prod$ et $\coprod$ introduites plus haut. Lorsque la catégorie index d'un diagramme est la catégorie $\mathbf{1}$ à un objet et à un morphisme, qui est un cas particulier de catégorie discrète, le diagramme correspondant est simplement la sélection d'un objet X dans la catégorie cible. Le produit et le coproduit sur un tel diagramme sélectionnant X sont tous les deux égaux à X, avec la projection et injection respectivement égales à l'identité. La catégorie vide $\mathbf{0}$ est elle aussi discrète, ce qui fait des objets terminaux et initiaux des cas particuliers de produits et coproduits vides, respectivement. Ceci permet d'ailleurs des interactions particulières entre ces limites et colimites :

Proposition 7. *Dans une catégorie ayant un objet terminal 1, pour tout objet X, on a $1 \times X \cong X \times 1 \cong X$. De même, dans une catégorie ayant un objet initial 0, pour tout objet X on a $X + 0 \cong 0 + X \cong X$.*

Autre fait notable, il existe une notion de *distribution* entre le produit et le coproduit incarné par l'existence, sans condition autre que l'existence des produits et coproduits requis, d'un morphisme entre deux constructions particulières.

Proposition 8. *Dans toute catégorie, pour tous objets A, B et C, il existe un unique morphisme de distribution $(A \times B) + (A \times C) \to A \times (B + C)$.*

Le morphisme en question est en fait donné par $[\mathrm{id}_A \times \iota_1, \mathrm{id}_A \times \iota_2]$. Il convient de noter cependant que ce morphisme n'est pas nécessairement inversible. Comme sous-entendu plus haut, étant donné un diagramme discret dans une catégorie $\mathcal{C}$, il n'est pas nécessaire que le produit ou que le coproduit associé existe. Les propositions précédentes ne sont valables que lorsque toutes les constructions impliquées existent. Il sera cependant courant de supposer par défaut que la catégorie de travail contient une certaine classe de ces constructions. On dira en particulier qu'une catégorie est *cartésienne* si elle a tous les produits finis, c.-à-d. si tout diagramme indexé par une catégorie discrète finie admet une limite.[60] La notion duale de catégorie *cocartésienne*

60. Notons que cette définition implique qu'une catégorie cartésienne a nécessairement un objet terminal, limite de la catégorie discrète vide. Par contraste, une catégorie présentée comme ayant tous les produits binaires a en fait accès par associativité à tous les produits finis *sauf* l'objet terminal.

est également définissable, mais est rarement employée. Notons en particulier qu'une catégorie cartésienne a nécessairement un objet terminal. On dira alors qu'une catégorie cartésienne avec au moins tous les coproduits binaires est *distributive* si tous les morphismes de distribution sont isos. D'autres propriétés d'intérêt viennent avec la distributivité, que nous présentons ci-dessous en guise de conclusion pour cette discussion des produits et coproduits.

Proposition 9. *Si une catégorie distributive est en plus cocartésienne, alors pour tout objet X on a $X \times 0 \cong 0 \times X \cong 0$.*

Proposition 10. *Si une catégorie est distributive, alors toutes les injections de coproduits sont des monomorphismes.*

Égalisateurs. Les *égalisateurs* et leur notion duale de *coégalisateurs* sont des limites moins systématiques que les précédentes et que les suivantes, mais tout aussi utiles : on y a fréquemment recours pour construire de certains types d'objets, et les propriétés associées à ces constructions en font des outils indispensables pour des preuves avancées. Ces limites reposent sur un diagramme de la forme particulière suivante :

Autrement dit, on considère ici la catégorie $\mathcal{I}$ ayant deux objets et, en plus des identités, deux morphismes distincts allant de l'un des objets vers le second. Un diagramme $D : \mathcal{I} \to \mathcal{C}$ indexé par $\mathcal{I}$ est alors la donnée de deux morphismes parallèles $f : A \to B$ et $g : A \to B$ dans $\mathcal{C}$. On considère la catégorie $\mathcal{I}$ fixée ainsi dans toute la suite de cette partie.

Définition 27. *Soient D un diagramme indexé par $\mathcal{I}$, et f et g les deux morphismes parallèles sélectionnés par D. On appelle* égalisateur *de f et g la limite sur D, et* coégalisateur *de f et g la colimite sous ce même diagramme.*

Plus concrètement, si f et g sont des morphismes $A \to B$, alors leur égalisateur est un objet E équipé d'un morphisme $e : E \to A$ vérifiant $f \circ e = g \circ e$, et tel que pour tout morphisme $u : X \to A$ vérifiant de même $f \circ u = g \circ u$ il existe un unique $h : X \to E$ avec $e \circ h = u$. Le dual est quant à lui donné par un objet E' et un morphisme $B \to E'$ dont la propriété universelle est aisée à deviner. Les égalisateurs et leur dual ont des liens très forts avec les notions de monomorphismes et les épimorphismes, comme en témoigne la proposition suivante :

Proposition 11. *Dans toute catégorie, un égalisateur est mono. De plus, si un égalisateur est épi, alors il est iso. Par dualité, tout coégalisateur est épi, et s'il est en plus mono, alors il est iso.*

Les égalisateurs et coégalisateurs fournissent donc un exemple de cas particulier où la réciproque de la proposition 1 est vraie, et permettent donc de créer facilement des monomorphismes et des épimorphismes. Si la courte description des propriétés de ces limites s'arrête ici pour l'instant, nous verrons que ces notions s'avèrent utiles ailleurs, par exemple lorsque nous discuterons des relations (cf. section 4.2.2).

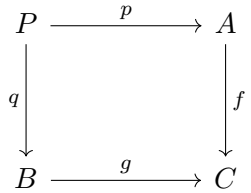
Produits fibrés et sommes amalgamées. La notion de produit fibré est presque omniprésente en théorie des catégories, et est certainement la limite la plus importante de par ses

utilisations nombreuses et ses propriétés très générales. Nous fixons pour cette partie une catégorie discrète $\mathcal{I}$ ayant la forme suivante :

soit trois objets et deux morphismes non identités de domaines distincts et dirigés vers le même codomaine, lui-même distinct des domaines. Un diagramme indexé sur $\mathcal{I}$ est alors essentiellement la donnée de deux morphismes de même codomaine $f : A \to C$ et $g : B \to C$.

Définition 28. *Soient D un diagramme de forme $\mathcal{I}$ et $f : A \to C$ et $g : B \to C$ les morphismes sélectionnés par D. On appelle* produit fibré *de f et g la limite sur D. Par dualité, si D' est un diagramme de forme $\mathcal{I}^{\mathrm{op}}$ sélectionnant deux morphismes $f' : C \to A$ et $g' : C \to B$, leur* somme amalgamée *est la colimite sous D'.*

Le produit fibré de $f : A \to C$ et $g : B \to C$ est parfois noté $A \times_C B$, mais nous éviterons autant que possible cette notation dans la mesure où elle passe sous silence la désignation de f et g. Tout comme le produit, un produit fibré vient avec des composantes également appelées *projections* vers les domaines des morphismes considérés, et on dira d'un carré commutatif comme ci-dessous qu'il est le *carré d'un produit fibré* lorsque l'objet P est le produit fibré de f et g ; les projections associées sont alors les morphismes p et q.



Considérant que la somme amalgamée est d'une utilisation bien plus rare que le produit fibré, la suite de cette partie se concentrera uniquement sur cette dernière ; néanmoins, tous les résultats valables pour l'un sont également transposables à l'autre par dualité. On commencera alors par remarquer que le produit binaire est un cas particulier de produit fibré :

Proposition 12. *Dans une catégorie ayant un objet terminal, un objet P est le produit fibré de $!_A : A \to 1$ et $!_B : B \to 1$ si et seulement si P est isomorphe à $A \times B$.*

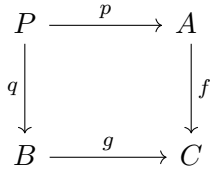
Une autre remarque intéressante permet de faire le lien entre les produits fibrés et les autres limites précédemment abordées : en présence de produits, les égalisateurs et les produits fibrés sont interdéfinissables.

Proposition 13. *Soit $\mathcal{C}$ une catégorie ayant au moins tous les produits binaires. Si $\mathcal{C}$ a en plus les égalisateurs, alors le produit fibré de deux morphismes $f : A \to C$ et $g : B \to C$ est isomorphe à l'égalisateur de $f \circ \pi_1 : A \times B \to C$ et $g \circ \pi_2 : A \times B \to C$.*

Proposition 14. *Soit $\mathcal{C}$ une catégorie ayant au moins tous les produits binaires. Si $\mathcal{C}$ a en plus les produits fibrés, alors l'égalisateur de deux morphismes $f, g : A \to B$ est isomorphe au produit fibré de $\langle \mathrm{id}_A, f \rangle : A \to A \times B$ et de $\langle \mathrm{id}_A, g \rangle : A \to A \times B$.*

Les composantes de ces limites s'entredéfinissent de manière analogue : le morphisme $e : E \to A \times B$ de l'égalisateur donne les projections $\pi_1 \circ e$ et $\pi_2 \circ e$ dans la première proposition, et les projections p_1 et p_2 du produit fibré $\langle \mathrm{id}_A, f \rangle$ et $\langle \mathrm{id}_A, g \rangle$ sont égales et donnent le morphisme de l'égalisateur dans la seconde proposition. Le résultat suivant permet d'affirmer certaines propriétés des projections d'un produit fibré en fonction des morphismes de départ.

Proposition 15. *Soit le carré de produit fibré suivant :*

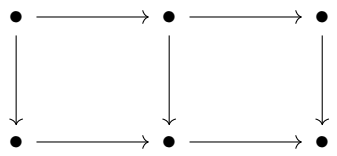


Si f est mono, alors q est également mono ; et si f est iso, alors q est également iso.

Cette dernière proposition est souvent résumée par la formule « les produits fibrés préservent les monos et les isos ». Nous verrons en effet ci-après en section 4.2.2 qu'il est possible de voir un produit fibré comme un certain foncteur qui préserve effectivement ces types de morphismes. En revanche, les épis ne sont généralement pas préservés, même si nous verrons certaines conditions où c'est le cas. Naturellement, dans la proposition f et q sont interchangeables avec g et p sans conditions.

Un dernier résultat essentiel permet d'évaluer si un carré commutatif donné est un produit fibré en fonction de possibles autres carrés de produits fibrés autour. Ce résultat est suffisamment connu et utile pour avoir son propre nom ; en français, nous l'appellerons le « lemme de collage des produits fibrés », que nous abrègerons en LCPF.

Lemme 2 (Collage des produits fibrés)**.** *Soit un diagramme commutatif de la forme suivante :*



Ce diagramme comporte trois carrés commutatifs : les deux petits, et le rectangle externe obtenu par composition des morphismes en haut et en bas du diagramme. Supposons que le carré de droite est un produit fibré. Alors, le carré de gauche est un produit fibré si et seulement si le rectangle externe en est un aussi.

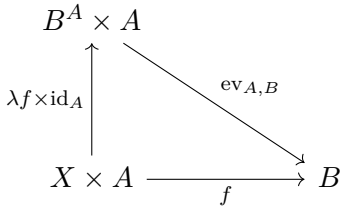
Ceci conclut notre discussion générale sur les produits fibrés. On pourra néanmoins s'attendre à les voir reparaître à plusieurs reprises dans ces pages, avec parfois des propriétés supplémentaires qui seront énoncées au moment opportun.

4.2.2 Autres constructions générales

Le but de cette section est d'introduire plusieurs concepts supplémentaires de la théorie des catégories qui n'ont pas de lien direct avec les limites, mais qui peuvent avoir des interactions intéressantes avec certaines limites. Certaines présentent également leurs propres propriétés universelles. La plupart des notions abordées ici apparaîtront régulièrement dans les chapitres suivants.

Exponentielles. Les *exponentielles* en théorie des catégories sont une généralisation des ensembles de fonctions. L'idée se retrouve très bien dans le cas de la catégorie Set : étant donné deux ensembles A et B, la classe $\mathsf{Set}(A, B)$ des fonctions de A vers B est elle-même un ensemble, et en tant que tel, peut être vue comme un objet de Set. Les exponentielles sont donc une représentation internalisée des *hom-sets*, c.-à-d. des objets représentant des classes de morphismes. La définition suivante fixe cette notion de façon plus formelle et introduit la propriété universelle inhérente à cette construction.

Définition 29. *Soient $\mathcal{C}$ une catégorie et A, B des objets tels que tous les produits binaires avec A existent. L'*exponentielle *de A et B dans $\mathcal{C}$ est la donnée d'un* objet exponentiel B^A *et d'un* morphisme d'évaluation $\mathrm{ev}_{A,B} : B^A \times A \to B$ *tels que, pour tous objet X et morphisme $f : X \times A \to B$, il existe un unique morphisme $\lambda f : X \to B^A$ faisant commuter le diagramme suivant :*



Intuitivement, $\mathrm{ev}_{A,B}$ est la généralisation de la fonction d'application $(g, a) \mapsto g(a)$, et λf est la curryfication de f sur son premier argument, c.-à-d. une généralisation de $x \mapsto (a \mapsto f(x, a))$. Tout comme les limites, une exponentielle est unique à isomorphisme près. On dira qu'un objet A est *exponentiable* si B^A existe pour tout objet B de la catégorie. En particulier, une catégorie cartésienne sera dite *fermée*[61] lorsque tous ses objets sont exponentiables.

La définition précédente sous-tend un opérateur λ envoyant un morphisme f sur son unique correspondant λf. Dans une catégorie localement petite, il se trouve que λ ainsi défini est en fait un isomorphisme naturel dans Set, établissant $\mathcal{C}(X \times A, B) \cong \mathcal{C}(X, B^A)$ pour tous objets X, A et B avec A exponentiable. On reconnaîtra dans cette propriété la définition d'une adjonction : A est exponentiable si et seulement si le foncteur $(-)^A : \mathcal{C} \to \mathcal{C}$ est adjoint à droite du foncteur $(-) \times A : \mathcal{C} \to \mathcal{C}$. Les morphismes d'évaluation pour A sont alors obtenus comme la co-unité de l'adjonction $\mathrm{ev}_A : (-)^A \times A \to I_{\mathcal{C}}$. Rappelons qu'avec le lemme de Yoneda, il est possible d'établir des isomorphismes internes sur la base d'isomorphismes naturels entre *hom-sets* ; il est alors possible d'exploiter l'isomorphisme naturel des exponentielles conjointement avec le lemme de Yoneda pour établir une série de résultats utiles.

Proposition 16. *Soit $\mathcal{C}$ une catégorie avec assez d'objets exponentiables. Alors :*

1. *pour tous exponentiables A et B et tout objet C, on a $C^{B \times A} \cong (C^B)^A$;*
2. *pour tout exponentiable A, et tous objets B et C, on a $(B \times C)^A \cong B^A \times C^A$.*

Si de plus $\mathcal{C}$ a un objet terminal, alors :

3. *pour tout A, on a $A^1 \cong A$;*
4. *pour tout A exponentiable, on a $1^A \cong 1$.*

Enfin, si $\mathcal{C}$ est de plus une catégorie distributive, alors :

5. *pour tous A, B exponentiables et tout C, on a $C^{A+B} \cong C^A \times C^B$;*
6. *pour tout A, on a $A^0 \cong 1$.*

L'accès aux exponentielles permet par ailleurs d'établir certaines propriétés supplémentaires autour de diverses autres limites. De nombreuses applications de la théorie des catégories reposent notamment sur des catégories cartésiennes fermées, qui ont accès aux propriétés décrites ci-dessous.

Proposition 17. *Soit $\mathcal{C}$ une catégorie cartésienne fermée avec un objet initial. Alors, pour tout objet $X \in \mathsf{obj}(\mathcal{C})$:*

61. On rencontre quelquefois le terme équivalent de catégorie *close*, qui est un calque du terme anglais correspondant.

1. $X \times 0 \cong 0 \times X \cong 0$;
2. *s'il existe un morphisme* $X \to 0$, *alors* $X \cong 0$;
3. *le morphisme* $0_X : 0 \to X$ *est mono.*

Il convient de remarquer que le point 1 de cette dernière proposition est très similaire à la proposition 9. En fait, ce résultat est accessible à toute catégorie cartésienne ayant un objet initial et soit les exponentielles (donc si elle est fermée), soit les coproduits binaires (donc si elle est distributive). Ceci conclut notre discussion des exponentielles, nous aborderons en section 4.3 un cas particulier de ces constructions que nous utiliserons intensivement.

Catégories virgules. Dans certains cas, il peut s'avérer utile de pouvoir manipuler des morphismes commes des objets dans une nouvelle catégorie, une application possible étant par exemple de construire des foncteurs reliant des objets à des morphismes. Il existe pour cela une construction très générale de *catégories virgules* dont plusieurs cas particuliers méritent notre attention.

Définition 30. *Soient* $\mathcal{C}$, $\mathcal{D}$ *et* $\mathcal{E}$ *des catégories, et deux foncteurs* $S : \mathcal{C} \to \mathcal{E}$ *et* $T : \mathcal{D} \to \mathcal{E}$. *La* catégorie virgule *de* S *et* T, *notée* $S \downarrow T$, *est la catégorie ayant pour objets les triplets* (A, B, f) *avec* $A \in \mathsf{obj}(\mathcal{C})$, $B \in \mathsf{obj}(\mathcal{D})$ *et* $f \in \mathcal{E}(SA, TB)$ *et pour morphismes* $(A, B, f) \to (A', B', f')$ *les paires* (g, h) *avec* $g : A \to A'$ *et* $h : B \to B'$ *telles que le diagramme suivant commute :*

$$\begin{array}{ccc} SA & \xrightarrow{\quad f \quad} & TB \\ {\scriptstyle Sg}\downarrow & & \downarrow{\scriptstyle Th} \\ SA' & \xrightarrow{\quad f' \quad} & TB' \end{array}$$

La composition s'obtient composante par composante, et l'identité sur (A, B, f) *est donné par la paire* $(\mathrm{id}_A, \mathrm{id}_B)$.

En fonction des foncteurs choisis, il est donc possible de construire différentes catégories d'intérêt. Nous en listons ci-dessous quelques exemples.

Exemple 8. *Étant données une catégorie* $\mathcal{C}$ *et un objet* $X \in \mathcal{C}$, *la catégorie virgule* $I_{\mathcal{C}} \downarrow \dot{X}_1$, *où* $I_{\mathcal{C}}$ *est le foncteur identité sur* $\mathcal{C}$ *et* $\dot{X}_1$ *le foncteur constant* $1 \to \mathcal{C}$ *égal à* X, *est appelée* catégorie tranche *sur* X *et notée plus simplement* $\mathcal{C} \downarrow X$. *Ses objets sont les morphismes de* $\mathcal{C}$ *de codomaine* X, *et si* $f : A \to X$ *et* $g : B \to X$ *alors un morphisme* $(A, f) \to (B, g)$ *est* $h : A \to B$ *tel que* $h \circ g = f$. *La construction duale* $X \downarrow \mathcal{C}$ *des morphismes de domaine* X *est nommée* catégorie cotranche *sous* X.

Exemple 9. *Étant donnée une catégorie* $\mathcal{C}$, *la catégorie virgule* $I_{\mathcal{C}} \downarrow I_{\mathcal{C}}$ *est nommée* catégorie flèche *de* $\mathcal{C}$ *et plus simplement notée* $\mathcal{C}^{\to}$. *Ses objets sont tous les morphismes de* $\mathcal{C}$, *et ses morphismes les paires de morphismes réalisant les carrés commutatifs appropriés. Cette catégorie est alors isomorphe à la catégorie de foncteurs* $\mathcal{C}^{\mathsf{2}}$, *où* $\mathsf{2}$ *est la catégorie à deux objets et un morphisme non identité.*

L'introduction des catégories virgules facilite l'expression de certaines propriétés universelles en terme de limites dans une catégorie bien construite. Un cas notable est celui de l'unité et de la co-unité d'une adjonction. Soient $L \dashv R$ une adjonction avec $L : \mathcal{C} \to \mathcal{D}$. Rappellons que l'unité

de cette adjonction est la transformation naturelle $\eta : I_{\mathcal{C}} \to RL$, laquelle jouit d'une propriété universelle (cf. section 4.1.2). Pour tout objet A, notons $A \downarrow R$ la catégorie virgule du foncteur constant $\dot{A} : 1 \to \mathcal{C}$ et de $R : \mathcal{D} \to \mathcal{C}$. Alors, pour tout $A \in \mathsf{obj}(\mathcal{C})$, la composante η_A est un objet initial dans $A \downarrow R$. De même, pour tout $B \in \mathsf{obj}(\mathcal{D})$ la composante ε_B de la co-unité de cette adjonction est un objet terminal de $L \downarrow B$.

Les catégories virgules permettent également d'exprimer les détails de certaines constructions sous forme de foncteurs, l'exemple canonique étant le cas des produits fibrés. Si $f : A \to C$ est un morphisme dans une catégorie $\mathcal{C}$ qui a tous les produits fibrés avec f, alors tout morphisme $g : B \to C$ permet d'obtenir un produit fibré P avec deux projections $p : P \to A$ et $q : P \to B$. On appelle alors *tiré de g le long de f* le morphisme p ainsi obtenu. Cette opération de « tirage » le long de f peut se formaliser sous la forme du *foncteur de produit fibré* $f^* : \mathcal{C} \downarrow C \to \mathcal{C} \downarrow A$ qui à tout $g : B \to C$ associe la projection sur A du produit fibré de f et g. Comme annoncé par la proposition 15, ce foncteur f^* préserve les monomorphismes et les isomorphismes, c.-à-d. que l'image par f^* de tout mono est aussi mono, et que l'image par f^* de tout iso est aussi iso.

Le foncteur de produit fibré fera à nouveau son apparition plus loin dans ces pages. Une autre définition, qui concluera pour l'instant notre discussion des catégories virgules, sera également réutilisée plus tard ; il nous semble cependant logique de l'inclure ici.

Définition 31. *Une catégorie $\mathcal{C}$ est* localement cartésienne fermée *si, pour tout $X \in \mathsf{obj}(\mathcal{C})$, la catégorie tranche $\mathcal{C} \downarrow X$ est cartésienne fermée.*

Cela signifie notamment qu'une catégorie $\mathcal{C}$ localement cartésienne fermée a tous les produits fibrés, car un produit dans une catégorie tranche de $\mathcal{C}$ est exactement un produit fibré dans $\mathcal{C}$. Si en plus $\mathcal{C}$ a un objet terminal alors elle est elle-même cartésienne fermée (car on peut alors construire les produits comme des produits fibrés sur 1), et a en fait toutes les limites finies : une catégorie qui satisfait cette dernière propriété est dite *finiment complète*. De façon équivalente, une catégorie est localement cartésienne fermée si et seulement si elle a tous les produits fibrés et que tout foncteur de produit fibré admet un adjoint à droite. Nous reviendrons sur cet adjoint en section 4.3.

Sous-objets et relations. La notion de *sous-objet* et plusieurs propriétés liées seront abondamment utilisées au cours de notre exposé. Quoique la manipulation des sous-objets va très souvent de pair avec le type de catégorie que nous verrons en section 4.3, il paraît raisonnable d'en introduire dès à présent les bases, à titre de transition bienvenue entre les propriétés de limites et cette future section. En préambule, considérons deux monomorphismes parallèles $f : A \rightarrowtail B$ et $g : A \rightarrowtail B$. On dira alors que f et g sont équivalents s'il existe un isomorphisme $h : A \to A$ tel que $f = g \circ h$. On vérifie aisément que ceci définit bien une relation d'équivalence sur tous les monomorphismes parallèles. Cette petite définition nous permet alors d'inaugurer la suivante, qui est celle qui nous intéresse :

Définition 32. *Soit B un objet dans une catégorie $\mathcal{C}$ quelconque. Un* sous-objet *de B est la donnée d'un objet A et d'une classe d'équivalence de monomorphismes $A \rightarrowtail B$.*

Avec la définition d'équivalence pour les monos donnée plus haut, on devinera donc qu'un sous-objet ne dépend pas précisément du monomorphisme choisi ; on se contentera la plupart du temps de choisir un représentant quelconque dans la classe d'équivalence et de ne travailler qu'avec lui. Par analogie avec Set, où les monos sont exactement les fonctions injectives, on se rend compte qu'être un sous-objet est légèrement plus général qu'être un sous-ensemble : en effet, il n'est pas question ici de préserver exactement les mêmes éléments — car la notion

d'éléments ne s'applique pas à toutes les catégories — mais de s'assurer au moins qu'un sous-objet « s'injecte » toujours dans la même image, d'où la classe d'équivalence.

Cela signifie également qu'il est possible d'avoir deux sous-objets différents avec un même domaine. Un exemple élémentaire de sous-objet est justement le cas des éléments globaux de forme $1 \to A$, la proposition 5 nous assurant que tout morphisme de cette forme est mono. Comme il n'existe qu'un seul morphisme $1 \to 1$, qui n'est autre que l'identité, chaque morphisme $1 \to A$ définit un sous-objet de A distinct. Dans le cas de Set, cela revient à dire que chaque élément d'un ensemble est distinct des autres. De même, d'après la proposition 17, dans une catégorie cartésienne fermée avec objet initial tout morphisme $0 \to A$ définit un sous-objet de A, qui est unique puisqu'il n'y a qu'un seul tel morphisme pour tout A. Enfin, en application de la proposition 11, tout égalisateur définit un sous-objet du domaine des morphismes qu'il égalise.

Étant donnés un objet C, il est possible de définir un ordre partiel sur ses sous-objets par la construction suivante : si $f : A \rightarrowtail C$ et $g : B \rightarrowtail C$ sont deux sous-objets de C, on pose $f \leq g$ lorsqu'il existe $h : A \to B$ tel que $f = g \circ h$[62]. La relation $\leq$ ainsi obtenue est bien réflexive et transitive. De plus, si $f \leq g$ et $g \leq f$, il existe h et i tels que $f = g \circ h$ et $g = f \circ i$. On établit alors que $f = f \circ i \circ h$ d'où $i \circ h = \mathrm{id}_A$ car f est mono, et de même on établit $h \circ i = \mathrm{id}_B$. Donc, h et i sont isos et inverses l'un de l'autre, et f et g sont équivalents. Or, un sous-objet est donné par la classe d'équivalence de ces monomorphismes, donc f et g représentent alors bien le même sous-objet et $\leq$ est antisymétrique. Il s'agit donc d'un ordre, et on notera $\mathrm{Sub}(C)$ l'ensemble partiellement ordonné des sous-objets de C avec l'ordre $\leq$.

La notion de sous-objet peut également servir de base à la construction de relations internes à une catégorie, comme expliqué par la définition suivante.

Définition 33. *Soit $\mathcal{C}$ une catégorie avec assez de produits, et soit $n \geq 2$. Une* relation interne *d'arité n sur les objets $A_1, \ldots, A_n$ dans $\mathcal{C}$ est un sous-objet du produit $\prod_{i=1}^{n} A_i$.*

On se permettra en général d'oublier la mention d'« interne » lorsque la situation est entendue. Une relation binaire sur A et B est alors un sous-objet de $A \times B$, et une relation binaire sur A est un sous-objet de $A \times A$. S'il s'agit pour l'instant d'une construction abstraite, nous verrons dans des chapitres ultérieurs qu'elle aura bien des applications importantes. Dans Set, elle recoupe évidemment la notion traditionnelle de relation. Un cas particulier de relation mérite d'être retenu : pour tout objet A tel que le produit $A \times A$ exsite, le *morphisme diagonal* $\delta_A : A \to A \times A$ est défini par $\delta_A = \langle \mathrm{id}_A, \mathrm{id}_A \rangle$. Si de tels produits existent pour tout objet, cela fait de δ une transformation naturelle $I_{\mathcal{C}} \to (-) \times (-)$. On vérifie aisément que ce morphisme est bien mono ; la relation qu'il définit est intuitivement la plus petite relation réflexive : dans Set, δ_A est la fonction $x \mapsto (x, x)$.

On dira par conséquent d'une relation $r : R \rightarrowtail A \times A$ sur A qu'elle est *réflexive* si δ_A se factorise à travers elle. De façon équivalente, elle est réflexive s'il existe un morphisme $h : A \to R$ qui soit section de $\pi_1 \circ r$ et de $\pi_2 \circ r$. D'autres propriétés classiques des relations peuvent être exprimées de manière interne. Ainsi, une relation $r : R \rightarrowtail A \times A$ sera *symétrique* s'il existe un morphisme $s : R \to R$ telle que $\pi_1 \circ r \circ s = \pi_2 \circ r$ et $\pi_2 \circ r \circ s = \pi_1 \circ r$, et sera *transitive* si, étant donné le produit fibré P ci-dessous,

$$\begin{array}{ccc} P & \xrightarrow{q_1} & R \\ {\scriptstyle q_2}\downarrow & & \downarrow{\scriptstyle \pi_2 \circ r} \\ R & \xrightarrow{\pi_1 \circ r} & A \end{array}$$

62. Par proposition 2, un tel h est alors nécessairement mono.

il existe un morphisme $t : P \to R$ tel que $r \circ t = \langle \pi_1 \circ r \circ q_1, \pi_2 \circ r \circ q_2 \rangle$. Intuitivement, dans Set, P représente les triplets (x, y, z) tels que $(x, y) \in R$ et $(y, z) \in R$, et $\langle \pi_1 \circ r \circ q_1, \pi_2 \circ r \circ q_2 \rangle$ est la fonction $(x, y, z) \mapsto (x, z)$ qui doit donc se factoriser à travers t pour définir une relation transitive. On appellera *congruence* une relation interne réflexive, symétrique et transitive.

Cette introduction des congruences permet au passage d'illustrer un cas d'utilisation des coégalisateurs : étant donné $r : R \rightarrowtail A \times A$ une congruence sur A, les projections $\pi_1 \circ r$ et $\pi_2 \circ r$ sont deux morphismes parallèles $R \to A$. Le coégalisateur de ces deux morphismes est alors noté A/R et appelé *objet quotient*[63]. Dans Set, cette notion rejoint exactement la notion de quotient par une relation d'équivalence, et le morphisme $f_R : A \to A/R$ est la fonction envoyant chaque élément de A vers sa classe d'équivalence.

4.3 Topos élémentaires

La dernière section de ce chapitre est consacrée à la présentation des propriétés d'une classe de catégories particulière, les *topos*[64]. Les topos peuvent être vus comme une généralisation de la catégorie Set, dans la mesure où tout topos, même lorsque ses objets sont très différents des ensembles, exhibe des propriétés catégoriques générales similaires à celles de Set. Ceci inclut non seulement l'existence des limites et des colimites, mais également d'autres constructions abstraites qui seront détaillées dans cette section.

Classificateur de sous-objet. La première de ces constructions abstraites, certainement la plus importante, a un lien très fort avec la notion de *sous-objet* précédemment introduite, et est présentée dans la définition qui suit. Rappellons qu'une catégorie *finiment complète* est une catégorie ayant toutes les limites finies.

Définition 34. *Soit $\mathcal{C}$ une catégorie finiment complète. Un* classificateur de sous-objet *dans $\mathcal{C}$ est la donnée d'un objet Ω et d'un élément global $\top : 1 \to \Omega$ tels que pour tout sous-objet $m : A \rightarrowtail B$ dans $\mathcal{C}$, il existe un unique morphisme $\chi_m : B \to \Omega$ tel que le carré suivant soit un produit fibré :*

$$\begin{array}{ccc} A & \xrightarrow{!_A} & 1 \\ {\scriptstyle m}\downarrow & & \downarrow{\scriptstyle \top} \\ B & \xrightarrow{\chi_m} & \Omega \end{array}$$

Le classificateur de sous-objet est unique à isomorphisme près. On appellera alors χ_m la *caractéristique* de m, et on dira également qu'un morphisme h de codomaine Ω *classifie* un sous-objet m lorsque $h = \chi_m$. Notons par ailleurs que le sous-objet classifié par h est unique, et donné par la classe d'équivalence du tiré de $\top$ le long h. Cette définition donne donc une correspondance entre sous-objets et morphismes vers Ω, qu'il est possible de formaliser à l'aide des ensembles partiellement ordonnés de sous-objets introduits précedemment.

63. Notons que cette définition est plus restrictive que dans [2], où le terme d'objet quotient est utilisé pour désigner la notion de « co-sous-objet ».

64. On trouve fréquemment, en français comme dans d'autres langues, le pluriel savant « topoï » — couramment usité pour le terme homonyme en rhétorique. Selon Cartier [31], l'introduction de ce terme en théorie des catégories est dû à Alexandre Grothendieck, qui écrivait le pluriel « topos ». Nous suivrons cet usage dans le présent exposé.

Proposition 18. *Soit $\mathcal{C}$ une catégorie localement petite et finiment complète. Alors, $\mathcal{C}$ a un classificateur de sous-objet Ω si et seulement s'il existe un isomorphisme $\theta_X : \mathrm{Sub}(X) \cong \mathcal{C}(X, \Omega)$ naturel en X.*

Cette propostion fait alors de $\mathrm{Sub}(-)$ un foncteur $\mathcal{C}^{\mathrm{op}} \to \mathsf{Set}$ représentable par Ω. Pour tout morphisme $f : X \to Y$, son image $\mathrm{Sub}(f) : \mathrm{Sub}(Y) \to \mathrm{Sub}(X)$ est la fonction qui à tout sous-objet de Y associe (la classe d'équivalence de) son tiré le long f.

Exemple 10. *Dans* Set*, l'ensemble à deux éléments $\{0, 1\}$ est un classificateur de sous-objet. Étant donnés un ensemble B et un sous-objet $m : A \rightarrowtail B$, la caractéristique de m est la fonction $\chi : B \mapsto \{0, 1\}$ définie par $\chi(x) = 1$ si $x \in m(A)$, et $\chi(x) = 0$ sinon, où $m(A)$ est l'image de A par m dans B, c.-à-d. $\{m(x) \mid x \in A\}$. La définition de χ recoupe alors celle de fonction caractéristique d'un sous-ensemble.*

Topos. Le classificateur de sous-objet est la pierre angulaire de la définition d'un topos. Cependant, plusieurs autres propriétés sont nécessaires, et les interactions entre ces propriétés sont telles que de nombreuses définitions de ce qu'est un topos sont possibles, si bien qu'en fonction des constructions que les auteurs veulent appuyer, divers ouvrages de référence (p. ex. [75, 114, 90]) utilisent des définitions différentes. Compte tenu des définitions que nous avons déjà abordées plus haut, et pour assurer la compacité de ce chapitre, nous retenons ci-dessous une définition simple et permettant d'accéder plus rapidement aux résultats qui nous intéressent.

Définition 35. *Un* topos élémentaire *est une catégorie finiment complète fermée avec un classificateur de sous-objet.*

Le terme « élémentaire » traduit la minimalité de la définition des topos au regard d'autres constructions (notamment les topos de Grothendieck), mais par commodité nous nous permettrons de l'omettre dans la suite de cet exposé. On comprend d'après cette définition que tout topos est en particulier une catégorie cartésienne fermée. Une autre propriété importante et non mentionnée par cette définition, bien qu'elle en soit une conséquence directe, est la suivante :

Proposition 19. *Tout topos est une catégorie finiment cocomplète et distributive.*

Autrement dit, les topos ont accès à toutes les colimites finies et l'interaction entre le produit et le coproduit se fait comme attendu, avec toutes les conséquences que cela peut avoir : toutes les propositions liées à ces propriétés sont vraies dans un topos.

Objets puissances. Autre construction essentielle de la théorie des topos, l'*objet puissance* sera beaucoup utilisé dans ces pages pour ses propriétés riches et versatiles. Au vu de la définition que nous avons donnée des topos, leur introduction ici sera assez directe, mais les objets puissances possèdent plusieurs caractérisations équivalentes qui ont chacune leur utilité et leur importance, et nous prendrons le temps d'explorer ces différentes approaches afin de cerner au mieux le comportement de ces objets.

Définition 36. *Soient $\mathcal{C}$ un topos et $A \in \mathsf{obj}(\mathcal{C})$. L'*objet puissance *de A dans $\mathcal{C}$, noté $\mathcal{P}A$, est l'exponentielle Ω^A.*

On comprend aisément en quoi cette définition est directe. Dans le détail, cela signifie que l'on dispose d'un morphisme d'évaluation $\mathrm{ev}_A : \mathcal{P}A \times A \to \Omega$ pour tout objet A, avec la propriété universelle associée : étant donné un morphisme $f : X \times A \to \Omega$, le morphisme $X \to \mathcal{P}A$

correspondant sera appelé *nom de* f et noté $\mathrm{name}(f)$. En particulier, pour $X \cong 1$, le nom d'un morphisme $A \to \Omega$ sera un morphisme $1 \to \mathcal{P}A$. On dira également qu'un morphisme $g : X \to \mathcal{P}A$ *nomme* f si $g = \mathrm{name}(f)$. Par ailleurs, on notera que la transformation $\mathcal{C}(X \times A, \Omega) \cong \mathcal{C}(X, \mathcal{P}A)$ est naturelle en X.

Il est tout à fait possible d'introduire les objets puissances en présentant sa propriété universelle sans mentionner les exponentielles. MacLane et Moerdijk [114] présentent par exemple les topos comme des catégories ayant un objet terminal, les produits fibrés, un classificateur de sous-objet et les objets puissances, puis parviennent à démontrer en conséquence de cette définition que les topos ont toutes les exponentielles. Il est également possible d'introduire les objets puissances indépendamment du classificateur de sous-objet en exploitant la propriété de ce dernier : au lieu de considérer $\mathrm{ev}_A : \mathcal{P}A \times A \to \Omega$ et $f : X \times A \to \Omega$, on prendra les tirés de $\top$ de long de ceux-ci. La propriété universelle est alors la suivante : un objet puissance est la donnée pour tout A d'une relation $e_A : \in_A \rightarrowtail \mathcal{P}A \times A$ telle que pour toute relation $r : R \rightarrowtail X \times A$, il existe un unique $f_r : X \to \mathcal{P}A$ de sorte que r soit le tiré de e_A le long de f_r, c.-à-d. qu'il existe le produit fibré à gauche ci-dessous :

$$\begin{array}{ccc} R & \longrightarrow & \in_A \\ {\scriptstyle r}\downarrow & & \downarrow{\scriptstyle e_A} \\ X \times A & \xrightarrow{f_r \times \mathrm{id}_A} & \mathcal{P}A \times A \end{array} \qquad \begin{array}{ccc} \mathcal{P}B \times A & \xrightarrow{\mathrm{id}_{\mathcal{P}B} \times f} & \mathcal{P}B \times B \\ {\scriptstyle \mathcal{P}f \times \mathrm{id}_A}\downarrow & & \downarrow{\scriptstyle \mathrm{ev}_B} \\ \mathcal{P}A \times A & \xrightarrow{\mathrm{ev}_A} & \Omega \end{array}$$

On comprend donc qu'il peut exister des catégories ayant des objets puissances sans qu'elles aient un classificateur de sous-objet. Cependant, ce sera le cas si elles disposent d'un objet terminal.

Proposition 20. *Dans toute catégorie ayant les objets puissances et un objet terminal,* $\mathcal{P}1$ *et* $\in_1 : 1 \to \mathcal{P}1$ *forment un classificateur de sous-objet.*

Dans un topos, on a donc l'isomorphisme $\mathcal{P}1 \cong \Omega$. Un autre isomorphisme notable, valable dans tout topos mais pas nécessairement dans toute catégorie, est $\mathcal{P}0 \cong 1$. Mais les propriétés de $\mathcal{P}$ ne s'arrêtent pas là : il est possible d'étendre cet opérateur en un foncteur $\mathcal{C}^{\mathrm{op}} \to \mathcal{C}$ en rendant l'isomorphisme $\mathcal{C}(X \times A, \Omega) \cong \mathcal{C}(X, \mathcal{P}A)$ également naturel en A. Étant donné $f : A \to B$, son image $\mathcal{P}f : \mathcal{P}B \to \mathcal{P}A$ est l'unique morphisme tel que le diagramme à droite ci-dessus commute. On vérifie aisément que pour tous morphismes f et g, on a bien $\mathcal{P}(g \circ f) = \mathcal{P}f \circ \mathcal{P}g$, et que $\mathcal{P}(\mathrm{id}_A) = \mathrm{id}_{\mathcal{P}A}$, confirmant donc que $\mathcal{P}$ ainsi défini est bien un foncteur.

Avec cette construction supplémentaire, il convient de souligner que dans un topos, un sous-objet donné pourra avoir trois représentations différentes, grâce aux bijections :

$$\mathrm{Sub}(X) \cong \mathcal{C}(X, \Omega) \cong \mathcal{C}(1, \mathcal{P}X).$$

Ces représentations sont donc un sous-objet $m : A \rightarrowtail X$, sa caractéristique $\chi_m : X \to \Omega$, et le nom de cette dernière $\mathrm{name}(\chi_m) : 1 \to \mathcal{P}X$. Tout l'art de manipuler les topos inclut celui de savoir jongler adroitement avec ces différentes représentations afin de prouver les résultats souhaités.

Factorisation épi-monique et autres résultats. Dans cette partie, nous rassemblons certains résultats des topos autour des morphismes et de leurs propriétés, notamment autour de l'interaction entre épis et monos. Commençons par lister les résulats suivants :

Proposition 21. *Dans un topos :*

1. *un morphisme est mono et épi si et seulement s'il est iso ;*
2. *tout mono est un égalisateur ;*
3. *tout épi est un coégalisateur.*

Plus précisément, un monomorphisme $m : A \rightarrowtail B$ est l'égalisateur est de χ_m et de la composée $\mathrm{true}_B = \top \circ !_B$, et un épimorphisme $e : A \twoheadrightarrow B$ est le coégalisateur des projections du produit fibré de e avec lui-même — on appelle ces projections la *paire noyau* de e. Nous avons vu plus haut que les égalisateurs étaient tous monos, avec une propriété duale pour les coégalisateurs ; nous savons désormais dans les topos que le lien et entre égalisateurs et monos (et dualement) est une équivalence.

Nous faisons suivre ce résultat d'une autre propriété encore plus importante. Étant donné un morphisme $f : A \to B$ quelconque, on appelle *image* de f le plus petit sous-objet de B (pour l'ordre partiel sur $\mathrm{Sub}(B)$) à travers lequel f se factorise. S'il existe, ce sous-objet est noté $f(A)$ et le monomorphisme associé est noté $\mathrm{im}\, f$.

Proposition 22. *Tout morphisme $f : A \to B$ dans un topos admet une image $\mathrm{im}\, f : f(A) \rightarrowtail B$, et le morphisme $e : A \to f(A)$ tel que $f = \mathrm{im}\, f \circ e$ est épi. De plus, cette décomposition est unique à isomorphisme près : si $m : C \rightarrowtail B$ et $e' : A \twoheadrightarrow C$ sont tels que $f = m \circ e'$, alors $C \cong f(A)$.*

On appelle *factorisation épi-monique* cette décomposition d'un morphisme en un épi suivi d'un mono. Cette construction nous sera très utile dans la suite dans la mesure où nous donnerons une signification particulière aux sous-objets, et que l'image sera un moyen simple d'en fabriquer. Par ailleurs, cette notion entretient une relation étroite avec la logique interne que nous introduirons d'ici la fin de cette section. Continuons cependant notre étude des monos et des épis avec la proposition suivante, spécifique aux topos.

Proposition 23. *Dans tout topos, le tiré d'un épimorphisme le long d'un morphisme quelconque est également épi.*

Autrement dit, dans un topos les produit fibrés préservent les épis, en plus de préserver les monos et les isos comme c'est toujours le cas d'après la proposition 15. Rappelons que pour $f : A \to B$ dans une catégorie $\mathcal{C}$, le foncteur de produit fibré associé est $f^* : \mathcal{C}\downarrow B \to \mathcal{C}\downarrow A$; si $\mathcal{C}$ est un topos, il préserve donc les trois types de morphismes. Relevons à ce propos que comme il préserve les monos, ce foncteur induit une fonction des sous-objets $f^{-1} : \mathrm{Sub}(B) \to \mathrm{Sub}(A)$.[65] Par la bijection $\mathrm{Sub}(X) \cong \mathcal{C}(1, \mathcal{P}X)$ et en application du lemme de Yoneda, $\mathcal{P}X$ peut être vu comme la version *interne* de $\mathrm{Sub}(X)$; et dans la même logique, $\mathcal{P}f$ pourra être vu comme une version interne de f^{-1}. Cette équivalence entre éléments internes au topos (c.-à-d. objets et morphismes) et externes au topos (c.-à-d. ensemble de sous-objets, fonctions associées) est une caractéristique majeure des topos, qui permet grâce au lemme de Yoneda de mener des raisonnements complexes sur des topos quelconques sur la base de manipulations dans Set ; nous utiliserons cette propriété pour certaines démonstrations dans le chapitre 6.

Terminons cette sous-partie par l'introduction d'un morphisme particulier, le *morphisme singleton*. Pour tout objet A, rappelons que le morphisme diagonal $\delta_A : A \to A \times A$ est un

65. Il conviendra de distinguer la notation f^{-1} pour la fonction des sous-objets engendré par f avec celle utilisé pour l'inverse éventuel d'un morphisme f ; on retiendra en particulier que le premier est un morphisme dans Set tandis que l'autre est généralement traité comme un morphisme dans une catégorie quelconque : le contexte permettra normalement d'éviter la confusion dans les prochains chapitres.

monomorphisme par définition : le morphisme singleton sur A est alors défini comme le nom de la caractéristique de ce morphisme diagonal, et est noté $\{\cdot\}_A : A \to \mathcal{P}A$. Ce nouveau morphisme porte un nom relativement intuitif : dans Set, il s'agit exactement de la fonction $x \mapsto \{x\}$. S'il nous intéresse autant ici, c'est par sa capacité à associer un objet avec son objet puissance d'une part, et sa propriété supplémentaire donnée ci-dessous d'autre part :

Proposition 24. *Pour tout objet A, le morphisme singleton $\{\cdot\}_A$ est mono.*

Les morphismes singletons interviendront à plusieurs reprises dans la construction de nos modèles sémantiques.

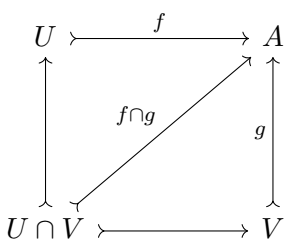
Algèbres de sous-objets et logique interne. L'un des intérêts principaux des topos est leur capacité à intégrer une logique interne puissante, les rendant aptes à interpréter des systèmes logiques d'ordre supérieur. L'intuition de ce fonctionnement est donné par le cas de Set, dont on a vu que le classificateur de sous-objet était (isomorphe à) l'ensemble $\{0, 1\}$, et que les sous-objets eux-mêmes étaient assimilés à des fonctions caractéristiques. Ce classificateur de sous-objet contient donc une valeur vraie 1 et une valeur fausse 0, sur lesquels il est possible de définir facilement des fonctions logiques. Il s'avère que ces constructions peuvent être étendues à tout topos, fournissant un arsenal de morphismes logiques permettant la construction d'une logique interne ; par la propriété du classificateur de sous-objet, cette logique interne correspond à une organisation particulière des ensembles de sous-objets d'un point de vue externe.

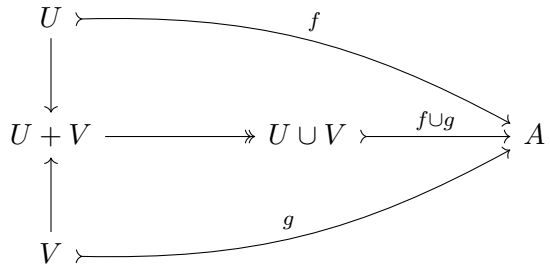
Il existe plusieurs manières de construire cette logique. Dans le présent exposé, nous prenons le parti de commencer par l'organisation externe en étudiant les ensembles de sous-objets, avant d'expliquer comment ces propriétés se répercutent de manière interne au topos. En guise de point de départ, nous commençons par signaler un résultat essentiel de la théorie des topos, fréquemment appelé « théorème fondamental » ; son importance dans notre discussion réside dans le fait que les ensembles de sous-objets s'injectent naturellement dans les catégories tranches.

Théorème 1 (Théorème fondamental des topos)**.** *Soit $\mathcal{C}$ un topos. Pour tout objet $A \in \mathsf{obj}(\mathcal{C})$, la catégorie $\mathcal{C}\downarrow A$ est aussi un topos.*

Étant donné un objet A, essayons de caractériser l'ensemble des sous-objets $\mathrm{Sub}(A)$. Nous rappelons que cet ensemble est partiellement ordonné par la relation de factorisation : $f : U \rightarrowtail A$ est plus petit que $g : V \rightarrowtail A$ s'il existe $h : U \to V$ tel que $f = h \circ g$. Cette relation a un plus grand élément évident : le sous-objet défini par l'identité sur A, qui est un cas particulier de monomorphisme. De même, nous savons que le morphisme $0_A : 0 \to A$ est mono ; ses propriétés en font le plus petit élément de $\mathrm{Sub}(A)$. Mais d'autres constructions sont accessibles à cet ensemble : en effet, la relation qui lie une catégorie à ses catégories tranches fait notamment que les produits fibrés dans la première sont des produits dans les secondes ; or nous savons que les produits fibrés préservent les monomorphismes, ce qui rend ce produit également disponible dans l'ensemble des sous-objets. Ainsi, si $f : U \rightarrowtail A$ et $g : V \rightarrowtail A$ sont des sous-objets de A, leur produit fibré engendre un nouveau sous-objet $f \cap g : U \cap V \rightarrowtail A$ selon le diagramme à gauche ci-dessous ; ce nouvel objet est appelé l'*intersection* de U et V et induit un opérateur binaire $\cap : \mathrm{Sub}(A) \times \mathrm{Sub}(A) \to \mathrm{Sub}(A)$. De même, il est possible de définir l'*union* de U et V en considérant leur coproduit, dont les propriétés assurent qu'il est équipé d'un morphisme $[f, g] : U + V \to A$; $f \cup g : U \cup V \rightarrowtail A$ est alors obtenu comme l'image de ce morphisme par factorisation épi-monique, suivant le diagramme à droite ci-dessous. Là encore, cette construction

induit un opérateur binaire $\cup$ sur $\mathrm{Sub}(A)$.





Du fait que ces deux constructions dérivent des produits et coproduits de la catégorie tranche $\mathcal{C} \downarrow A$ qui est un topos, elles intéragissent parfaitement bien entre elles : des propriétés essentielles comme la distributivité sont en particulier obtenues. Une autre propriété énoncée ci-dessous consolide par ailleurs la vision du coproduit dans $\mathcal{C}$ comme une union disjointe, où la disjonction se traduit ici par le fait d'avoir un produit fibré « vide », c.-à-d. étant un objet initial :

Proposition 25. *Soient $\mathcal{C}$ un topos et $A \in \mathsf{obj}(\mathcal{C})$. Pour tous sous-objets U et V de A, si $U \cap V \cong 0$, alors $U + V \cong U \cup V$.*

Soulignons par ailleurs qu'en tant que topos, $\mathcal{C} \downarrow A$ a les exponentielles, et que suivant un argument complexe que nous ne reproduirons pas ici (cf. [114, §IV.6-8]), l'exponentielle de deux sous-objets est encore un sous-objet. Ceci induit donc l'existence d'un opérateur binaire $\Rightarrow$ sur $\mathrm{Sub}(A)$ qui est adjoint à droite de l'intersection, ce qui signifie notamment que pour tous sous-objets U, V et X de A, on a $X \cap U \leq V$ si et seulement si $X \leq U \Rightarrow V$. L'ajout de cet opérateur achève de donner à $\mathrm{Sub}(A)$ une structure particulière, régissant un certain nombre d'interactions entre les opérateurs ainsi définis, que nous énonçons dans la proposition suivante :

Proposition 26. *Soit $\mathcal{C}$ un topos quelconque. Pour tout $A \in \mathsf{obj}(\mathcal{C})$, $(\mathrm{Sub}(A), \leq, 0, A, \cup, \cap, \Rightarrow)$ est une algèbre de Heyting.*

Nous ne détaillerons pas les propriétés des algèbres de Heyting ici, mais l'on pourra se reporter par exemple à [75, §8.3] ou [114, §I.8] pour davantage d'informations. Notons toutefois qu'une telle algèbre, vue comme une catégorie, est alors cartésienne fermée et distributive ; la plupart des résultats pertinents ont donc déjà été exprimés dans la section précédente.

Pour compléter notre étude des ensembles de sous-objets, il nous reste à nous pencher sur les interactions entre les algèbres de sous-objets d'un même topos. Nous pouvons commencer par ce premier résultat :

Proposition 27. *Soient C un topos et $f : A \to B$ un morphisme quelconque. Alors, le foncteur $f^{-1} : \mathrm{Sub}(B) \to \mathrm{Sub}(A)$ est un homomorphisme d'algèbres de Heyting.*

Par ailleurs, si l'on considère toujours un morphisme $f : A \to B$ quelconque dans un topos $\mathcal{C}$, il se trouve que le foncteur de produit fibré $f^* : \mathcal{C} \downarrow B \to \mathcal{C} \downarrow A$ admet un adjoint à gauche et un adjoint à droite, lesquels induisent des adjoints similaires à la fonction f^{-1}. L'adjoint à gauche de f^{-1} est noté $\exists_f : \mathrm{Sub}(A) \to \mathrm{Sub}(B)$ et est la fonction qui à tout sous-objet $g : U \rightarrowtail A$ associe le sous-objet de B représenté par l'image de $f \circ g$. Son adjoint à droite est noté $\forall_f : \mathrm{Sub}(A) \to \mathrm{Sub}(B)$, mais il est beaucoup plus délicat d'en donner une définition concrète comme pour $\exists_f$.[66] Pour en obtenir une bonne intuition cependant, nous pouvons expliciter à quoi ces adjoints

66. Ceci n'est cependant pas impossible, mais nous estimons que la difficulté de cette construction n'apporte pas d'éclairage supplémentaire dans le cadre du présent exposé ; cf. [75, §15.3] pour une caractérisation concrète.

ressemblent dans Set : si A et B sont des ensembles, et si U est un sous-ensemble de A, alors $\exists_f(U) = \{y \in B \mid \exists x \in A.f(x) = y \wedge x \in U\}$, et $\forall_f(U) = \{y \in B \mid \forall x \in A.f(x) = y \Rightarrow x \in U\}$.

Du fait des isomorphismes naturels entre $\mathrm{Sub}(A)$ et les *hom-sets* $\mathcal{C}(A, \Omega)$ et $\mathcal{C}(1, \mathcal{P}A)$ et en application du lemme de Yoneda, toutes ces constructions ont des versions internes correspondantes : les opérateurs $\cap$, $\cup$ et $\Rightarrow$ sur $\mathrm{Sub}(A)$ engendrent notamment des morphismes $\mathcal{P}A \times \mathcal{P}A \to \mathcal{P}A$, et pour tout $f : A \to B$, les fonctions $\exists_f$ et $\forall_f$ induisent deux morphismes $\exists f$ et $\forall f : \mathcal{P}A \to \mathcal{P}B$.[67] Tout comme leurs équivalents externes, ces quantificateurs présentent un comportement fonctoriel dans le sens où $\exists(g \circ f) = \exists g \circ \exists f$ et $\forall(g \circ f) = \forall g \circ \forall f$. Le quantificateur existentiel présente par ailleurs des propriétés importantes, que nous réutiliserons dans nos applications :

Proposition 28. *Dans tout topos, si $m : A \rightarrowtail B$ est un monomorphisme, alors $\mathcal{P}m \circ \exists m = \mathrm{id}_{\mathcal{P}A}$. De plus, si $f : U \rightarrowtail A$ est un sous-objet de A, alors* $\mathrm{name}(\chi_{m \circ f}) = \exists m \circ \mathrm{name}(\chi_f)$.

Notons que la première partie de cette proposition nous dit que si m est mono, alors $\exists m$ est split mono (et incidemment, $\mathcal{P}m$ est split épi). Quant à la seconde partie, elle internalise simplement la propriété de $\exists_m$ à prolonger un sous-objet de A en un sous-objet de B.

Suivant l'isomorphisme $\mathcal{P}1 \cong \Omega$, les opérateurs des algèbres de sous-objets s'internalisent également en connecteurs logiques directement sur Ω, donnant lieu à des morphismes $\wedge, \vee, \Rightarrow$: $\Omega \times \Omega \to \Omega$ ainsi qu'aux quantificateurs $\exists_A, \forall_A : \Omega^A \to \Omega$ pour tout A, que viennent compléter le morphisme $\top : 1 \to \Omega$ fourni par la définition du classificateur de sous-objet et correspondant à A dans $\mathrm{Sub}(A)$, représentant la valeur vraie, et le morphisme $\bot : 1 \to \Omega$ classifiant $0_1 : 0 \rightarrowtail 1$, qui correspond à 0 dans $\mathrm{Sub}(A)$ et à la valeur fausse. Ces morphismes interagissent avec la propriété classificatrice de Ω comme attendu : par exemple, pour tous sous-objets $f : U \rightarrowtail A$ et $g : V \rightarrowtail A$, on a $\wedge \circ \langle \chi_f, \chi_g \rangle = \chi_{f \cap g}$, et idem pour les autres connecteurs ; de plus, $\mathrm{true}_A = \chi_{\mathrm{id}_A}$ et $\bot \circ !_A = \chi_{0_A}$.

Ajoutons par ailleurs que dans une algèbre de Heyting, le pseudo complément par rapport à 0 définit le complément, ou la négation, d'un élément. Pour tout sous-objet $f : U \rightarrowtail A$, on définit alors $\overline{f} = f \Rightarrow 0_A$ et $\overline{U} = U \Rightarrow 0$. Ce complément s'internalise alors en complément interne $\mathcal{P}A \to \mathcal{P}A$, et surtout comme la négation $\neg : \Omega \to \Omega$, qui satisfait notamment $\neg \circ \top = \bot$ et $\neg \circ \bot = \top$. De nombreuses autres égalités (ou inégalités) logiques sont possibles, reproduisant les propriétés des algèbres de Heyting. La logique ainsi obtenue est au moins intuitionniste, bien que certaines conditions peuvent donner accès à la logique classique. Parmi les nombreuses propriétés logiques possibles, il nous paraît utile d'en rappeler une qui aura son utilité en section 6.2 ; il s'agit d'une proposition valide en logique intuitionniste que nous exprimons ici de manière externe.

Proposition 29. *Pour tous morphisme $f : A \to B$ et sous-objet $U \rightarrowtail A$, on a $\forall_f(\overline{U}) = \overline{\exists_f(U)}$.*

On comprend ainsi comment un topos peut refléter un système logique, grâce à sa logique interne à laquelle correspond une certaine organisation externe, au niveau des ensembles de sous-objets. Nous terminerons cette introduction aux topos par une rapide discussion de la notion de *bivalence*. De façon générale, il est assez intuitif que chaque morphisme $1 \to \Omega$ désigne une valeur de vérité, et nous en avons rencontré au moins deux : les morphismes $\top$ et $\bot$. Un topos $\mathcal{C}$ est dit *bivalent* si ces morphismes sont les seuls possibles, c.-à-d. si $\mathcal{C}(1, \Omega)$ est l'ensemble $\{\top, \bot\}$. Mais il existe des topos admettant des valeurs de vérités supplémentaires, fournissant en particulier des modèles catégoriques pour des logiques multivaluées. Nous n'utiliserons pas de tels systèmes logiques dans la présente étude, mais les affirmations générales qui y seront

67. Il est même possible d'introduire une notion d'*adjonction interne*, et de montrer que ces morphismes sont alors adjoints internes à gauche et à droite de $\mathcal{P}f$. Pour plus de détails sur cette définition, cf. [114, §IV.9].

faites pourront également s'appliquer à de tels topos. Mentionnons par ailleurs que la propriété de bivalence est distincte de celle que *classicité*, qui détermine si le topos est un modèle de la logique intuitionniste ou de la logique classique : un topos est *classique* si $1 + 1 \cong \Omega$, c.-à-d. si l'objet des valeurs de vérité ne contient rien de plus que les images des morphismes $\top$ et $\bot$. Si un topos classique est nécessairement bivalent, il existe des topos qui prouvent que la réciproque est fausse. Nous discuterons des implications de ces propriétés au regard de nos objectifs linguistiques en sous-section 5.4.3, mais par défaut, nous tâcherons d'utiliser les topos indépendamment de toute hypothèse sur sa bivalence ou sa classicité.

Chapitre 5

Modèles catégoriques des théories de types

Wie plump fällt die Sprache mit ihrem Einen Worte über so ein polyphones Wesen her!

Friedrich Nietzsche, *Morgenröthe*

5.1 Principes généraux

Les théories de types, telles qu'introduites au chapitre 1, sont des constructions formelles abstraites, reposant sur des ensembles de symboles syntaxiques et une construction récursive de couples terme-type au moyen de règles prédéfinies ; en conséquence, elles sont fondamentalement dépourvues de signification propre. En réalité, le choix des symboles utilisés est pensé pour avoir du sens : la flèche est implicitement associée aux types fonctions, le signe de multiplication aux types produits, les lettres lambda aux termes fonctions, et ainsi de suite — mais cette signification est mentalement projetée sur les symboles par force d'habitudes et de conventions, alors qu'à ce stade, une théorie des types ainsi définie n'est qu'un objet syntaxique sans interprétation de ces significations. La notion de *modèle* est l'élément qui permet justement de fixer concrètement cette signification, de définir ce que les symboles représentent et comment ils doivent être compris. Les modèles correspondent en cela à la notion de sémantique dans les langages de programmation [68] : il s'agit de fonctions qui permettent de traduire la syntaxe abstraite en objets mathématiques, et la signification est alors imposée par la manière dont ces objets mathématiques interragissent entre eux.

Un modèle de la théorie des types est plus précisément une paire de fonctions, l'une interprétant les types et l'autre interprétant les termes, avec pour condition de maintenir une relation logique entre ces interprétations qui reflète la relation de typage et sa construction au moyen des règles de la théorie. Dans notre approche des représentations formelles de la langue naturelle, nous avons pris les théories de types comme des langages permettant de modéliser la sémantique linguistique, aussi cette notion de modèle pour les théories de types pourrait être présentée comme « la sémantique de la sémantique », c.-à-d. la structure mathématique qui permet de connecter la représentation logique des significations au monde réel lui-même. [69]

68. Pour quelques exemples de telles interprétations, cf. [150].

69. Il serait peut-être plus juste de parler de pré-connexion ici, dans la mesure où ce que dit effectivement un énoncé linguistique du monde réel dépend également de facteurs pragmatiques qui ne sont pas capturés par ce

On comprend donc que les modèles qu'il nous faudra construire devront présenter des propriétés cohérentes avec les observations linguistiques, et en particulier, proposer une interprétation des types sémantiques qui respecte les observations que nous avons faites dans la partie I. Inévitablement, ces modèles varient selon les ensembles de symboles disponibles, et plus les théories de types possèdent de traits additionnels, plus les modèles associés seront complexes. Dans le cas qui nous intéresse, il faudra donc fournir des interprétations adéquates pour rendre compte notamment du sous-typage, des coercions et des changements de types, de préférence dans un modèle uniforme.

Mais dans un premier temps, nous allons commencer par illustrer la notion de modèle sur un exemple basique : le modèle ensembliste du lambda-calcul simplement typé. Considérons plus précisément le système de types $\mathbb{T}$ librement engendré par un ensemble quelconque T de types de base et par le système de constructeur $\{\to (2)\}$. Pour la constructions des termes, on suppose disposer d'un ensemble de *variables* V. L'ensemble Λ des termes du lambda-calcul est alors récursivement défini par les trois conditions suivantes : (i) $V \subset \Lambda$, (ii) si $u, v \in \Lambda$, alors $uv \in \Lambda$,[70] et (iii) si $x \in V$ et $u \in \Lambda$, alors $\lambda x.u \in \Lambda$. Cette même définition est plus rapidement capturée par la grammaire en forme de Backus-Naur suivante :

$$\Lambda ::= V \mid \Lambda\Lambda \mid \lambda V.\Lambda$$

Néanmoins, la théorie des types simples impose des restrictions sur les termes de l'ensemble Λ qui sont typables. En effet, la construction d'un terme bien typé se fonde sur l'unité syntaxique de base que constitue la variable : il est donc nécessaire de connaître le type qu'on attribue aux variables pour pouvoir en déduire le type d'un terme plus complexe qui les utilise. Cela rend le typage dépendant d'une collection d'hypothèses sur le type des variables, que l'on rassemble dans une concaténation de paires variables-types appelée *contexte*, construite de telle sorte que chaque variable y apparaisse au plus une fois. Afin de prévenir les potentiels problèmes de noms associés aux variables, nous adoptons ici la convention fixée par Jacobs [88] : dans n'importe quel contexte, les variables présentes seront de la forme x_i, où l'indice i numérote les variables dans l'ordre où elles apparaissent dans le contexte. Ainsi, un contexte Γ quelconque sera de la forme $x_1 : \sigma_1, \ldots, x_n : \sigma_n$, et si $\Delta = x_1 : \tau_1, \ldots, x_m : \tau_m$ est un autre contexte, alors leur *concaténation* s'écrira $\Gamma, \Delta = x_1 : \sigma_1, \ldots, x_n : \sigma_n, x_{n+1} : \tau_1, \ldots, x_{n+m} : \tau_m$. On s'autorisera cependant à utiliser les méta-variables $x, y, z \ldots$ lorsque la numérotation n'est pas strictement nécessaire, ou pour les petits exemples.

Un terme du lambda-calcul n'est alors typable qu'en contexte, un phénomène qui est capturé par la notation de *jugement* : la déclaration qu'un terme u a le type τ dans un contexte Γ sera noté $\Gamma \vdash u : \tau$. Les règles de typage du lambda-calcul simplement typé permettent alors la construction récursive de l'ensemble des jugements corrects, et on dira formellement qu'un terme u donné est *typable* s'il existe un contexte Γ et un type τ tels que $\Gamma \vdash u : \tau$ est un tel jugement correct. Nous donns ci-dessous les trois règles principales pour la construction des termes typés :

$$\frac{}{\Gamma, x : \tau, \Gamma' \vdash x : \tau}\,(\text{ax}) \qquad \frac{\Gamma \vdash u : \sigma \to \tau \qquad \Gamma \vdash v : \sigma}{\Gamma \vdash uv : \tau}\,(\text{app}) \qquad \frac{\Gamma, x : \sigma \vdash u : \tau}{\Gamma \vdash \lambda x.u : \sigma \to \tau}\,(\text{lam})$$

niveau d'analyse. Fort heureusement, nous verrons plus bas que cette nuance peut être facilement capturée en laissant aux modèles une part d'abstraction irrésolue : les modèles que nous construirons n'ont en effet pas pour but immédiat de formaliser le réel, mais plutôt de garantir les propriétés nécessaires d'une telle formalisation.

70. Comme usuellement en lambda-calcul, nous utilisons les parenthèses pour clarifier l'ordre de lecture lorsqu'un terme contient plusieurs applications. Nous nous permettrons néanmoins de les omettre dans le cas d'applications successives d'un même terme initial : si $u_1, \ldots, u_n$ sont des termes, alors $u_1 \ldots u_n$ désignera le terme $(\cdots((u_1 u_2)u_3)\cdots u_n)$.

À ces règles de typage peuvent optionnellement s'ajouter des règles structurelles sur la manipulation des contextes, que nous nous passerons de reproduire puisqu'elles sont facilement retrouvables comme règles admissibles du système, ainsi qu'une *théorie équationnelle*, qui permet d'imposer une égalité de fait à des termes syntaxiquement différents. Pour le lambda-calcul, on distingue deux égalités majeures appelées β-conversion et η-conversion. [71] L'écriture des règles associée utilise la notion de *substitution*, c.-à-d. le remplacement de toutes les occurrences libres d'une variable par un terme : on notera $u[v/x]$ la substitution de la variable x par v dans le terme u. Les règles sont alors les suivantes [72] :

$$\frac{\Gamma, x : \sigma \vdash u : \tau \qquad \Gamma \vdash v : \sigma}{\Gamma \vdash (\lambda x.u)v = u[v/x] : \tau}\,(\beta) \qquad \frac{\Gamma \vdash u : \sigma \to \tau \qquad x \notin \mathrm{fv}(u)}{\Gamma \vdash \lambda x.ux = u : \sigma \to \tau}\,(\eta)$$

où dans la seconde règle, la condition $x \notin \mathrm{fv}(u)$ signifie que la variable x utilisée en conclusion n'est pas libre dans u.

Toutes ces composantes, types, termes, règles de typage et théorie équationnelle, définissent donc la théorie des types simples. Mais comme avancé au début de cette section, les symboles ainsi formés n'ont encore aucune signification propre, et il appartient aux modèles de formaliser les propriétés attendues par cette théorie. En l'occurrence, nous souhaiterions intuitivement exprimer le comportement fonctionnel des lambda-abstractions. Le modèle le plus simple que l'on puisse imaginer pour ce faire — et également l'un des premiers proposés, cf. [86] —, assimile les types à des ensembles et le typage à la relation d'appartenance ensembliste : chaque type de base $\tau \in T$ est ainsi associé à un ensemble A_τ, avec la condition que si $\sigma \neq \tau$ alors A_σ et A_τ sont disjoints. Puis, on associe récursivement à chaque type flèche $\sigma \to \tau$ l'ensemble $A_{\sigma\to\tau} = A_\tau^{A_\sigma}$ des fonctions de A_σ vers A_τ, de sorte à disposer d'un ensemble associé à chaque type dans $\mathbb{T}$.

On devine alors qu'un terme typé pourra être interprété comme un élément de l'ensemble associé à son type. Cependant, il faut garder à l'esprit que les termes typés dépendent eux-même d'un contexte, et que par conséquent, leurs interprétations seront paramétrées selon la manière dont les variables libres qu'il contient sont interprétées. Nous utilisons pour cela une *valuation*, une fonction $\rho : V \to (\bigcup_{\tau\in\mathbb{T}} A_\tau)$ associant à chaque variable de V un élément dans l'un des ensembles interprétant les types. Si ρ est une telle valuation, alors $\rho(x \mapsto a)$ désignera la valuation telle que $\rho(x \mapsto a)(x) = a$ et $\rho(x \mapsto a)(y) = \rho(y)$ pour tout $y \neq x$. Étant donnée une valuation ρ, il est possible de définir une interprétation des termes comme une fonction $[\![.]\!]_\rho$ définie comme suit :

$$\begin{aligned}
[\![x]\!]_\rho &= \rho(x) \\
[\![uv]\!]_\rho &= [\![u]\!]_\rho([\![v]\!]_\rho) \\
[\![\lambda x.u]\!]_\rho &= a \mapsto [\![u]\!]_{\rho(x\mapsto a)}
\end{aligned}$$

Naturellement, cette définition n'est pas entièrement satisfaisante dans la mesure où elle n'est pas cohérente si la valuation utilisée ne respecte pas les bonnes hypothèses sur le type des variables. Un moyen de s'assurer que le typage est bien respecté est d'introduire les interprétations des variables utilisées comme des paramètres de l'interprétation finale. En pratique, on doit

71. Nous écartons ici la notion d'α-conversion, qui peut être capturée grâce à notre discipline de numérotation des variables en contexte couplée à la règle structurelle d'*échange* permettant de permuter les types des variables en contexte au prix d'une substitution parallèle des variables concernées.

72. Bien sûr, la théorie équationnelle ne peut être complète qu'en ajoutant des règles qui fondent la relation $=$ sur les termes comme une relation d'équivalence — réflexivité, symétrie et transitivité — ainsi que les règles de congruence qui permettent d'établir des égalités entre les sous-termes. Nous nous passons de les reproduire ici par commodité, mais il convient de garder à l'esprit qu'elles forment un noyau dur commun à toutes les théories équationnelles.

donc interpréter des jugements de typage entier pour récuperer toutes les hypothèses issues du contexte : si Γ est le contexte $x_1 : \sigma_1, \ldots, x_n : \sigma_n$, l'interprétation du jugement $\Gamma \vdash u : \tau$ est définie à partir d'une valuation ρ quelconque mais fixée par

$$[\![\Gamma \vdash u : \tau]\!] = (a_1, \ldots, a_n) \mapsto [\![u]\!]_{\rho(x_1,\ldots,x_n \mapsto a_1,\ldots,a_n)}$$

qui est donc une fonction $A_{\sigma_1} \times \cdots \times A_{\sigma_n} \to A_\tau$. Lorsque le terme considéré est *fermé*, c.-à-d. qu'il ne contient pas de variables libres, il est possible d'obtenir un jugement de typage avec un contexte vide, auquel cas l'interprétation est simplement un élément de l'ensemble associé à son type : $[\![\vdash u : \tau]\!] = [\![u]\!]_\rho \in A_\tau$. Comme l'interprétation d'un tel terme ne dépend plus de la valuation, on se passera alors d'écrire l'indice ρ.

Rendre compte de la théorie équationnelle dans un tel modèle revient donc à montrer que les interprétations des termes concernés sont effectivemment égales dans le modèle. Pour la β-conversion, il faut d'abord souligner que pour une valuation ρ proprement choisie, on a l'égalité $[\![u[v/x]]\!]_\rho = [\![u]\!]_{\rho(x \mapsto [\![v]\!]_\rho)}$, qui se démontre par récurrence en suivant la définition de la substitution. Alors, si l'on fait abstraction des paramètres de contexte en posant, pour des valeurs $a_1, \ldots, a_n$ bien choisies, la valuation $\rho' = \rho(x_1, \ldots, x_n \mapsto a_1, \ldots, a_n)$, la β-conversion en contexte est bien capturée par le modèle puisqu'on peut établir :

$$\begin{aligned}
[\![(\lambda x.u)v]\!]_{\rho'} &= [\![\lambda x.u]\!]_{\rho'}([\![v]\!]_{\rho'}) \\
&= [\![u]\!]_{\rho'(x \mapsto [\![v]\!]_{\rho'})} \\
&= [\![u[v/x]]\!]_{\rho'}
\end{aligned}$$

Un raisonnement similaire peut s'appliquer pour l'η-conversion, en déroulant la définition de l'interprétation sur le terme η-étendu :

$$\begin{aligned}
[\![\lambda x.ux]\!]_{\rho'} &= a \mapsto [\![ux]\!]_{\rho'(x \mapsto a)} \\
&= a \mapsto [\![u]\!]_{\rho'(x \mapsto a)}([\![x]\!]_{\rho'(x \mapsto a)}) \\
&= a \mapsto [\![u]\!]_{\rho'(x \mapsto a)}(a) \\
&= [\![u]\!]_{\rho'(x \mapsto a)} \\
&= [\![u]\!]_{\rho'},
\end{aligned}$$

l'égalité de la dernière ligne étant obtenue en se rappelant que, par hypothèse, la variable x n'apparaît pas libre dans le terme u, ce qui rend son interprétation totalement indépendante de la valuation assignée à cette variable.

La construction d'un tel modèle permet donc de formaliser la signification des lambda-abstractions comme des fonctions, et permet également de justifier la théorie équationnelle en interprétant l'égalité comme une relation d'équivalence pertinente. Il est même possible d'aller plus loin en considérant des lambda-calculs avec *constantes* : il s'agit d'enrichir la théorie des types avec une signature supplémentaire $(Q, \mathfrak{t})$ où Q est un ensemble de symboles et $\mathfrak{t}$ une fonction $Q \to \mathbb{T}$, qui permet d'introduire des termes fixes dans le calcul ; on complète cet enrichissement par l'ajout de la règle suivante :

$$\frac{q \in Q}{\Gamma \vdash q : \mathfrak{t}(q)}\,(\text{const})$$

L'interprétation d'une constante q se fait en choisissant un élément distingué de $A_{\mathfrak{t}(q)}$ pour le représenter, on peut donc reprendre notre modèle précédent en y ajoutant simplement une seconde valuation $\nu : Q \to (\bigcup_{\tau \in \mathbb{T}} A_\tau)$ fixée, de sorte que pour toute constante $q \in Q$ et toute valuation de variables ρ, on ait $[\![q]\!]_\rho = \nu(q)$.

Cette introduction des constantes peut éventuellement mener à une théorie équationnelle étendue, qui est encore une fois justifiable à partir des modèles. À titre d'exemple, nous allons considérer un lambda-calcul arithmétique fondé sur l'unique type de base $\mathtt{n}$ et ayant trois constantes $\mathbf{0} : \mathtt{n}$, $\mathbf{succ} : \mathtt{n} \to \mathtt{n}$, et $+ : \mathtt{n} \to \mathtt{n} \to \mathtt{n}$. On utilisera pour cette dernière la notation infixe par souci de lisibilité. La théorie qui en découle permet de construire de nombreux lambda-termes bien typés, comme p. ex. $\mathbf{succ}(\mathbf{0}) : \mathtt{n}$ ou $\lambda x.\, x + \mathbf{succ}(x) : \mathtt{n} \to \mathtt{n}$. Pour ce qui est du modèle, on va naturellement commencer par poser $A_{\mathtt{n}} = \mathbb{N}$, et construire la valuation de constantes ν à l'aide des égalités suivantes :

$$\begin{aligned} \nu(\mathbf{0}) &= 0 \\ \nu(\mathbf{succ}) &= n \mapsto n+1 \\ \nu(+) &= n, m \mapsto n+m \end{aligned}$$

En appliquant alors le principe du modèle ensembliste précédemment détaillé, il est alors possible d'interpréter les termes bien typés comme des entiers ou des fonctions impliquant des entiers : pour les termes précédemment donnés, on obtient ainsi $[\![\mathbf{succ}(\mathbf{0})]\!] = 1$, et $[\![\lambda x.\, x + \mathbf{succ}(x)]\!] = n \mapsto 2n+1$. Mais les capacités du modèle permettent également, comme annoncé au début du paragraphe, d'étendre la théorie équationnelle de ce système en y ajoutant certains des axiomes de Péano. Par exemple, il est possible d'imposer dans ce système les règles d'égalité

$$\frac{\Gamma \vdash x : \mathtt{n}}{\Gamma \vdash x + \mathbf{0} = x : \mathtt{n}} \qquad \frac{\Gamma \vdash x : \mathtt{n} \quad \Gamma \vdash y : \mathtt{n}}{\Gamma \vdash \mathbf{succ}(x+y) = x + \mathbf{succ}(y) : \mathtt{n}}$$

en vérifiant qu'on a bien $[\![x+\mathbf{0}]\!]_\rho = [\![x]\!]_\rho = \rho(x)$ d'une part, et $[\![\mathbf{succ}(x+y)]\!]_\rho = [\![x+\mathbf{succ}(y)]\!]_\rho = \rho(x) + \rho(y) + 1$ d'autre part, pour toute valuation ρ cohérente avec le typage attendu.

Relevons que dans cet exemple, le mot *vérifier* n'est pas choisi au hasard : il y a en effet une relation asymétrique entre les égalités de termes dans la théorie et les égalités de leurs interprétations dans le modèle, où la première implique la seconde, alors que l'inverse n'est pas nécessairement vrai. En particulier, bien qu'il soit évident que l'opération $+$ sur $\mathbb{N}$ est commutative, il n'est pas possible de démontrer le jugement $\Gamma \vdash x+y = y+x : \mathtt{n}$ par applications successives des règles précédentes, ce qui rend la constante $+$ dans le langage des termes non commutative : pour ajouter cette propriété à la théorie, il est nécessaire d'introduire une règle supplémentaire ayant le jugement ci-dessus comme conclusion. De manière générale, les théories de types que nous présentons ici sont *intensionnelles*, ce qui signifie qu'elles ne capturent pas nécessairement toutes les égalités qui peuvent exister dans leurs modèles, par opposition aux théories dites *extensionnelles* qui imposent une équivalence entre égalités de termes et d'éléments. L'une des conséquences de cette position est qu'il est possible de définir plusieurs théories de types différentes qui admettent un même modèle.

Réciproquement, une théorie des types peut admettre plusieurs modèles. Dans l'exemple précédent, il est essentiel de remarquer que la théorie équationnelle qu'il est possible de développer ne dépend pas précisément de l'ensemble $\mathbb{N}$ dans lequel on l'évalue, mais plutôt des propriétés intrinsèques de cet ensemble, et de celles des éléments qui interprètent ses constantes. Il est en effet possible de capturer la même théorie dans d'autres modèles, pourvu qu'ils aient ces mêmes propriétés : en l'occurrence, il nous suffit de choisir un monoïde $(M, \star, e)$ quelconque ainsi qu'un élément $f \in M$, puis d'imposer l'interprétation suivante :

$$\begin{aligned} \nu(\mathbf{0}) &= e \\ \nu(\mathbf{succ}) &= x \mapsto x \star f \\ \nu(+) &= x, y \mapsto x \star y \end{aligned}$$

On vérifie alors aisément les égalités données en règles ci-dessus dans ce nouveau modèle, de la même manière que précédemment. Ce schéma donne ainsi un large choix de modèles possibles autres que $(\mathbb{N}, +, 0)$ avec $f = 1$: pour rester dans les entiers, on a par exemple le modèle dégénéré où $f = 0$, qui satisfait toujours les égalités de la théorie, le modèle $(\mathbb{N}^*, \times, 1)$ avec f quelconque, ou encore le modèle $(2\mathbb{N}, +, 0)$ avec $f = 2$; mais bien entendu, il est également possible de choisir des monoïdes plus exotiques.

Ce qu'il est important de retenir ici, c'est qu'une théorie des types données admet une entière classe de modèles possibles, obéissant aux mêmes propriétés générales et imposant les mêmes conditions sur l'interprétation des constantes. Cette remarque invite donc à davantage d'abstraction, en mettant l'accent sur les propriétés des modèles plutôt que sur leur contenu précis. On peut d'ailleurs se rendre compte que l'interprétation ensembliste n'est pas toujours suffisante pour capturer toutes ces propriétés : il est souvent nécessaire de s'intéresser à des structures plus riches, comme des monoïdes ou des anneaux par exemple. Cela est également valable pour les applications linguistiques auxquelles nous aspirons : le système UTT de Luo et Chatzikyriakidis [35], à l'instar de nombreuses théories basées sur la théorie intuitionniste de Martin-Löf [115], interprète par exemple les types comme des *sétoïdes*, c.-à-d. des paires constituées d'un ensemble et d'une relation d'équivalence sur cet ensemble pour remplacer l'égalité. De manière générale, nous nous intéresserons donc plutôt aux propriétés des modèles plutôt qu'aux modèles eux-mêmes, et l'un des meilleurs moyens pour ce faire est de faire appel à la théorie des catégories, qui apporte la quantité d'abstraction nécessaire à une telle fin. Dans la section suivante, nous allons ainsi revisiter la construction des modèles en termes catégoriques.

5.2 Modèles catégoriques pour la théorie des types simples

La force des modèles catégoriques pour l'interprétation des théories de types est leur capacité à rendre compte des propriétés générales nécessaires à de telles interprétations sans besoin de préciser outre mesure en quoi consiste le modèle. Cette capacité est d'autant plus importante qu'elle est renforcée par l'observation que des théories de types non-équivalentes sont caractérisées par des modèles catégoriques eux-mêmes non-équivalents, se distinguant les uns des autres par des propriétés uniques. Il existe par conséquent une correspondance manifeste entre des classes de théories des types et des classes de catégories, parmi lesquelles figure la correspondance entre lambda-calculs simplement typés et catégories cartésiennes fermées, démontrée originellement par [100]. Dans la présente section, nous allons généraliser la notion de modèle introduite précédemment à des interprétations catégoriques, et déterminer quels enseignements on peut tirer d'un tel procédé. Nous nous basons pour ces explications sur divers exemples issus de la littérature, dont en particulier [88, 177].

Nous allons cependant traiter une forme altérée de la théorie de types présentée en section 5.1 : elle sera désormais enrichie d'un nouveau constructeur $\times$ devant être compris comme un produit. L'intérêt d'ajouter un tel constructeur ici est triple : premièrement, il répond aux conclusions tirées de la première partie de ce travail, notamment en sous-section 3.3.3, où nous avons établi la nécessité de types complexes pour une bonne modélisation de l'hétérotypie, pour lesquels des types produits semblent bien adaptés, nous rapprochant donc de notre objectif final ; ensuite, il permet de faciliter la construction du modèle catégorique, offrant ainsi une meilleure clarté d'exposition pour le propos de cette section ; et enfin, nous verrons qu'il tient un rôle essentiel dans les constructions que nous formulerons au chapitre 6. Pour accompagner ces types produits, nous introduisons également un type unité, noté 1, qui représentera intuitivement un produit vide. Nous considérons donc un nouveau système de types $\mathbb{T}$ librement engendré par un

ensemble de types de base T et par le système de constructeurs $\{\to (2), \times(2), 1\,(0)\}$, un ensemble de variables V, une signature de constantes $(Q, \mathfrak{t})$ et la grammaire de termes suivantes :

$$\Lambda ::= * \mid V \mid Q \mid \Lambda\Lambda \mid \lambda V.\Lambda \mid \langle \Lambda, \Lambda \rangle \mid \pi_i \Lambda \qquad (i \in \{1, 2\})$$

où $*$ est un symbole distinct des constantes dans Q.

Les règles de typages de ce lambda-calcul sont fondées sur les mêmes règles que dans la section précédente — incluant les règles structurelles que nous n'avons pas reproduites —, auxquelles s'ajoutent les règles permettant de manipuler les nouveaux termes et types, que nous listons ci-dessous :

$$\frac{}{\vdash * : 1}\,(\text{unit}) \qquad \frac{\Gamma \vdash u : \sigma \qquad \Gamma \vdash v : \tau}{\Gamma \vdash \langle u, v \rangle : \sigma \times \tau}\,(\text{pair}) \qquad \frac{\Gamma \vdash u : \sigma \times \tau}{\Gamma \vdash \pi_1 u : \sigma}\,(\text{proj}_1) \qquad \frac{\Gamma \vdash u : \sigma \times \tau}{\Gamma \vdash \pi_2 u : \tau}\,(\text{proj}_2)$$

Ces nouveautés sont bien évidemment complétées par une théorie équationnelle étendue à l'aide des règles suivantes :

$$\frac{\Gamma \vdash u : 1}{\Gamma \vdash u = * : 1} \qquad \frac{\Gamma \vdash u : \sigma \qquad \Gamma \vdash v : \tau}{\Gamma \vdash \pi_1 \langle u, v \rangle = u : \sigma} \qquad \frac{\Gamma \vdash u : \sigma \qquad \Gamma \vdash v : \tau}{\Gamma \vdash \pi_2 \langle u, v \rangle = v : \tau} \qquad \frac{\Gamma \vdash u : \sigma \times \tau}{\Gamma \vdash \langle \pi_1 u, \pi_2 u \rangle = u : \sigma \times \tau}$$

La première de ces règles définit $*$ comme l'unique terme de type 1, et les trois autres décrivent les différentes relations entre les paires et les projections π_1 et π_2.

Comment interpréter un tel calcul à l'aune de la théorie des catégories ? Dans le modèle ensembliste introduit en section 5.1, nous interprétions les jugements comme des fonctions entre ensembles. Or, si l'on se rémémore les exemples évoqués au chapitre 4, Set est une catégorie ayant les ensembles pour objets et les fonctions totales pour morphismes. L'analogie est alors rapide à faire : on interprétera les termes en contexte comme des morphismes entre des objets bien choisis, par exemple des ensembles. Mais avant d'explorer plus en détail ces modèles, il est en fait possible de construire une catégorie canonique associée à une théorie des types : il s'agit de la *catégorie classificatrice* [88, 177] de la théorie en question, qu'on appelle aussi *catégorie des contextes* suite à leur introduction initiale par Cartmell [32]. Ce type de catégorie a en règle générale pour objets les contextes, et pour morphismes des n-uplets de termes typables dans ces contextes, et permet de représenter de façon canonique n'importe quelle structure algébrique.

Néanmoins, pour les théories de types qui nous intéressent dans ces pages, nous disposons du constructeur de types produits $\times$, qui permet une simplification non-négligeable : en effet, en présence de tels types, les contextes $x_1 : \tau_1, \ldots, x_n : \tau_n$ et $x : \tau_1 \times \cdots \times \tau_n$ sont équivalents, car si u est typable dans le premier, alors $u[\pi_1 x / x_1, \ldots, \pi_n x / x_n]$ est typable dans le second. [73] Dans pareil cas, la simplification est la suivante : on prendra pour objets de la catégorie classificatrice directement les types dans $\mathbb{T}$. Si σ et τ sont deux objets dans cette catégorie, alors les morphismes $\sigma \to \tau$ sont les classes d'équivalences des termes u satisfaisant $x : \sigma \vdash u : \tau$ pour la relation équationnelle $=$ entre termes. Cette construction donne effectivement lieu à une catégorie car l'identité sur σ est fournie par le jugement $x : \sigma \vdash x : \sigma$, et la composition est donnée par substitution d'un terme dans un autre : en effet, si on peut prouver les jugements $x : \xi \vdash u : \sigma$ et $y : \sigma \vdash v : \tau$, alors on peut prouver $x : \xi \vdash v[u/y] : \tau$.

La théorie des types simples telle qu'introduite ci-dessus est un schéma général qui repose sur deux paramètres : l'ensemble T des types de bases, et la signature $(Q, \mathfrak{t})$ des constantes. Par commodité, nous regroupons ces paramètres en une seule signature $\Sigma = (T, Q, \mathfrak{t})$. Alors, la

73. Cette notation se base sur un raccourci syntaxique de convenance défini par $\langle u_1, \ldots, u_n \rangle = \langle u_1, \langle \cdots, u_n \rangle \rangle$ et $\pi_n u = \pi_1 \pi_2 \ldots \pi_2 u$, où π_2 est présent exactement $n - 1$ fois.

catégorie classificatrice associée à notre théorie de type sera notée $\mathcal{C}\ell_{\mathrm{STT}}(\Sigma)$.[74] En exploitant les propriétés du lambda-calcul, il est alors possible d'énoncer la proposition suivante, dont on trouvera une démonstration dans [88, §2.3] :

Proposition 30. *Pour toute signature Σ, la catégorie $\mathcal{C}\ell_{\mathrm{STT}}(\Sigma)$ est cartésienne fermée.*

L'objet 1 de $\mathcal{C}\ell_{\mathrm{STT}}(\Sigma)$, qui interprète le contexte vide, est l'objet terminal de cette catégorie. De plus, le produit de σ et τ est exactement $\sigma \times \tau$, et l'exponentielle est obtenue en posant $\tau^\sigma = (\sigma \to \tau)$[75] avec pour évaluation associée

$$\mathrm{ev}_{\sigma,\tau} = [(\pi_1 x)(\pi_2 x)] : \tau^\sigma \times \sigma \to \tau,$$

terme obtenu à partir du jugement $x : (\sigma \to \tau) \times \sigma \vdash (\pi_1 x)(\pi_2 x) : \tau$ — en notant que $[u]$ désigne ici la classe d'équivalence pour la relation $=$ du terme u. Par ailleurs, la transformation naturelle associée à cette exponentielle envoie l'interprétation $[u] : \xi \times \sigma \to \tau$ du jugement $z : \xi \times \sigma \vdash u : \tau$ vers le morphisme $\lambda[u] = [\lambda y.u[\langle x, y\rangle/z]] : \xi \to \tau^\sigma$. Ceci explique en particulier pourquoi nous utilisons la lettre λ pour représenter cette transformation : elle interprète exactement la règle (lam) de lambda-abstraction.

De manière plus générale, toutes les règles de typage de cette théorie sont interprétées par des correspondances entre le morphisme construit sur le jugement en conclusion de la règle et un autre morphisme construit catégoriquement dans $\mathcal{C}\ell_{\mathrm{STT}}(\Sigma)$ à partir de ses prémisses. Par exemple, il découle de la définition du produit que $[\langle u, v\rangle] = \langle [u], [v]\rangle$, interprétant ainsi la règle (pair). Autre exemple, la règle d'application (app), sur la base des interprétations de ses prémisses comme les morphismes $[u] : \xi \to \tau^\sigma$ et $[v] : \xi \to \sigma$, s'interprète comme l'égalité entre les morphismes $[uv]$ et $\mathrm{ev}_{\sigma,\tau} \circ \langle [u], [v]\rangle : \xi \to \tau$, qui se vérifie ainsi :

$$\begin{aligned}
\mathrm{ev}_{\sigma,\tau} \circ \langle [u], [v]\rangle &= \mathrm{ev}_{\sigma,\tau} \circ [\langle u, v\rangle] \\
&= [(\pi_1 x)(\pi_2 x)[\langle u, v\rangle/x]] \\
&= [(\pi_1 \langle u, v\rangle)(\pi_2 \langle u, v\rangle)] \\
&= [uv]
\end{aligned}$$

Il en va de même pour l'interprétation de la théorie équationnelle : elle s'incarne également par des égalités de morphismes dans la catégorie classificatrice, mais qui résultent cette fois de propriétés universelles. Ainsi, les interprétations de la β-conversion et de l'η-conversion résultent toutes deux de la propriété qui définit l'exponentielle : pour la première, elle établit en effet l'égalité $\mathrm{ev}_{\sigma,\tau} \circ (\lambda[u] \times \mathrm{id}_\sigma) = [u] : \xi \times \sigma \to \tau$, où le morphisme de gauche est exactement l'interprétation du jugement $z : \xi, y : \sigma \vdash (\lambda x.u)y : \tau$ tandis qu'on vérifie à droite $[u] = [u[y/x]]$; par conséquent, on a bien pour tout v correctement typé :

$$[(\lambda x.u)v] = [((\lambda x.u)y)[v/y]] = [(\lambda x.u)y] \circ [v] = [u] \circ [v] = [u[v/x]]$$

74. La notation Σ introduite ici ne doit pas être confondue avec celle utilisée par Jacobs [88] pour ses *signatures multitypes*. Il existe en effet deux différences fondamentales entre ces deux signatures : d'une part, la signature multitype autorise un même symbole à avoir plusieurs types — de manière similaire à la surcharge de fonctions en programmation — alors que notre signature n'assigne qu'un type par symbole ; et d'autre part, notre signature évalue le type des constantes dans $\mathbb{T}$ alors qu'une signature multitype est limitée à T. Notre signature est alors a priori plus puissante, puisque par duplication de symboles toute signature multitype s'encode dans une des nôtres, mais elles sont en fait moins générales puisqu'elles cachent une dépendance au système de constructeurs : la perspective de Jacobs permet une définition des théories de types qui s'appuie sur n'importe quelle signature multitype, alors que nos signatures sont restreintes à la théorie des types simples.

75. Il s'agit bien ici d'une égalité définitionnelle, et pas d'un simple isomorphisme. Nous utiliserons dans la suite la notation σ^τ pour les types fonction afin d'éviter toute confusion avec la notation générale des morphismes.

De même, la seconde conversion repose sur l'égalité $\lambda(\mathrm{ev}_{\sigma,\tau} \circ ([u] \times \mathrm{id}_\sigma)) = [u]$, où le terme de gauche interprète justement $\lambda x.ux$. Quant aux égalités posées pour l'introduction du type produit et des termes associés, elles se retrouvent dans l'expression des propriétés universelles liées au produit, de façon assez directe.

Sur la base de ces explications, on relèvera en particulier que les termes fermés, qui sont typables dans un contexte vide, constituent les éléments globaux de $\mathscr{C}_{\mathrm{STT}}(\Sigma)$. C'est notamment le cas des symboles dans la signature des constantes $(Q, \mathfrak{t})$: chaque $q \in Q$ définit dans la catégorie classficatrice un morphisme $[q] : 1 \to \mathfrak{t}(q)$. Dans le chapitre 1, nous avons vu que la théorie de types utilisée par Montague était un lambda-calcul utilisant de nombreuses constantes de prédicat de tous ordres, dont une grande part de prédicats du premier ordre comme par exemple $\mathbf{walk} : e \to t$. Considérons la catégorie classificatrice de ce calcul dont les objets de base seront notés E et T par commodité, fixant ainsi la convention noter en majuscules les objets catégoriques correspondant aux types notés par des lettres latines minuscules. Alors, l'interprétation catégorique d'un tel prédicat est un morphisme $[\mathbf{walk}] : 1 \to T^E$, et si on veut appliquer ce prédicat sur l'argument *John*, dont la sémantique $\mathbf{j} : e$ est représentée par le morphisme $[\mathbf{j}] : 1 \to E$, on utilise le morphisme d'évaluation pour former l'interprétation $\mathrm{ev}_{e,t} \circ \langle [\mathbf{walk}], [\mathbf{j}] \rangle : 1 \to T$, qui est bien égale à $[\mathbf{walk}(\mathbf{j})]$ d'après ce qui précède. Cependant, ce recours au morphisme d'évaluation rend la lecture de l'interprétation plus difficile. Intuitivement, on aurait aimé assimiler l'application à une simple composition de morphismes, ce qui est bien évidemment écarté par la présente interprétation. Il existe toutefois un moyen de s'en rapprocher : par η-expansion, $\mathbf{walk}$ est égal $\lambda x.\mathbf{walk}(x)$ dans un contexte vide. Or, ce dernier lambda-terme a par la règle (lam) un jugement $x : e \vdash \mathbf{walk}(x) : t$ correspondant. Que se passe-t-il si l'on utilise l'interprétation quasi-équivalente de ce jugement plutôt que l'interprétation directe de la constante ? Le morphisme correspondant est $[\mathbf{walk}(x)] : E \to T$, et l'on obtient :

$$[\mathbf{walk}(\mathbf{j})] = [\mathbf{walk}(x)[\mathbf{j}/x]] = [\mathbf{walk}(x)] \circ [\mathbf{j}] : 1 \to T$$

Autrement dit, on trouve une forme de correspondance entre application et composition qui satisfait notre intuition. Et elle est loin d'être un simple cas isolé : elle est très générale en vertu de la bijection $\lambda : \mathscr{C}_{\mathrm{STT}}(\Sigma)(E, T) \cong \mathscr{C}_{\mathrm{STT}}(\Sigma)(1, T^E)$ qui caractérise l'exponentielle dans la catégorie classificatrice, et fonctionne par conséquent pour toute application du calcul initial. Du point de vue de la théorie des types, cette correspondance incarne une stratégie précise : limiter l'application à des arguments variables, puis effectuer une substitution sur l'argument précédemment fourni. Cette stratégie étant a priori implémentable dans tout lambda-calcul, il serait tentant d'essayer d'en tirer parti pour former une interprétation plus simple de l'application comme composition. Nous ne creuserons pas davantage cette idée ici, mais nous la gardons de côté pour réutilisation dans les considérations qui accompagneront la construction de nos théorie de types (cf. section 6.1).

Avec cette construction des catégories classificatrices pour la théorie des types simples, nous avons la véritable base des modèles catégoriques. On notera cependant que cette construction est syntaxique : les objets et morphismes sont donnés directement par les types et termes de la théorie sans autre forme d'interprétation. En fait, cette construction est la manière la plus élémentaire de représenter cette théorie des types dans la théorie des catégories. Comment alors proposer des modèles alternatifs, comme le modèle ensembliste de la section précédente ? La clé réside dans la définition de la notion de modèle catégorique, que nous donnons ci-dessous :

Définition 37. *Soit Σ une signature. Un* modèle catégorique *de la théorie des types simples est la donnée d'une catégorie cartésienne fermée $\mathcal{C}$ et d'un foncteur $M : \mathscr{C}_{\mathrm{STT}}(\Sigma) \to \mathcal{C}$ qui préserve cette structure.*

Dire comme ici qu'un foncteur préserve la structure des catégories cartésiennes fermées signifie qu'il préserve les produits d'une part, et les exponentielles d'autre part. La catégorie Set étant justement cartésienne fermée, il est alors possible de retrouver le modèle ensembliste de la section précédente à partir du foncteur $[\![.]\!] : \mathcal{C}\ell_{\mathrm{STT}}(\Sigma) \to \mathsf{Set}$ défini sur les objets par $[\![\sigma]\!] = A_\sigma$, et sur les morphismes comme en section 5.1, avec une extension triviale aux n-uplets. [76] Et naturellement, on comprend vite qu'il est possible de remplacer Set dans cet exemple par n'importe quelle catégorie cartésienne fermée, permettant en conséquence des interprétations plus riches presque sans effort. Comme établi à la fin de la section précédente, cette approche permet de mettre en avant les propriétés du modèle plutôt que le contenu lui-même : pour la théorie des types simples avec produits présentée dans ces pages, ces propriétés sont incarnées par le concept de catégorie cartésienne fermée. Il est d'ailleurs possible de définir CCC, la catégorie ayant pour objets les catégories cartésiennes fermées localement petites et pour morphismes les foncteurs qui préservent cette structure. On retrouve parmi les objets de cette catégorie nos catégories classificatrices (qui sont petites, donc localement petites), de même que Set. Alors, un modèle de la théorie des types simples sur Σ est simplement un morphisme de CCC ayant $\mathcal{C}\ell_{\mathrm{STT}}(\Sigma)$ pour domaine.

Nous terminerons cette discussion sur les modèles catégoriques par deux remarques. La première concerne la théorie équationnelle : comme il a été évoqué plus haut, il y a une distinction entre les égalités qui sont valides dans la théorie des types et celles qui sont valides dans un modèle de cette théorie. Par construction, si $u = v$ est valable dans la théorie des types, alors $[\![u]\!] = [\![v]\!]$ dans tout modèle ; mais la réciproque n'est pas nécessairement vraie, et distingue les théories comme extensionnelles ou intensionnelles. Il est possible d'étendre notre notion de modèle pour définir une logique équationnelle intensionnelle, où des égalités supplémentaires peuvent être ajoutées en hypothèses sans être intégrées à la théorie des types proprement dite, et permet de déterminer si des modèles donnés satisfont ou non ces nouvelles équations. Il est même possible d'étendre encore cette approche pour intégrer une logique complète du premier ordre satisfaisable par des modèles catégoriques.

La seconde remarque porte sur la généralisation de la construction présentée dans cette section : si nous n'avons discuté que de la théorie des types simples pour l'instant, l'élaboration de catégories classificatrices fonctionne de la même manière pour des théories de types utilisant des constructeurs de types supplémentaires. Un exemple très simple est l'introduction des types sommes $+$ et du type vide 0, qui mènent à des interprétations dans des catégories cocartésiennes. D'autres choix de constructeurs voire de règles de typage aboutiront de même à des propriétés supplémentaires ou différentes dans les modèles, donnant lieu à de nouvelles correspondances entre familles de théories des types et classes de catégories. Outre la correspondance établie dans la présente section, démontrée initialement par Lambek [100], on retrouvera notamment une correspondance entre la théorie des types de Martin-Löf [115] et les catégories localement cartésiennes fermées [166], ainsi que celle entre la logique d'ordre supérieure et les topos [62, 101]. Dans la suite de ce chapitre, nous allons nous appuyer sur les idées posées par ces deux remarques pour explorer les propriétés catégoriques nécessaires à des modèles cohérents avec nos exigences linguistiques.

5.3 Vers des modèles adaptés à la sémantique linguistique

Le lambda-calcul utilisé par Montague dans sa proposition initiale [120] est fondé sur une théorie des types qui ressemble beaucoup aux théories précédemment abordées, mais qui dispose

76. Par abus de notation, on écrira $[\![u]\!]$ à la place de $[\![[u]]\!]$ pour tout terme u.

d'une dimension supplémentaire : sa logique. En effet, cette théorie n'est ni plus ni moins qu'un calcul des prédicats étendu en lambda-calcul, c.-à-d. la représentation en termes algébriques (et donc en théorie des types) d'une méthode pour construire des formules logiques. Par conséquent, il est très probable qu'on exige d'un modèle de la théorie en question qu'elle réponde à ces exigences logiques, en implémentant des propriétés adaptées. À la fin de la section précédente, nous avons évoqué la possibilité d'étendre les modèles pour y intégrer une logique complète du premier ordre. Dans cette section, nous commençons donc par expliquer comment une telle logique fonctionne et s'interprète dans ces modèles catégoriques, avant de comparer ce système aux contraintes posées par celui de Montague.

5.3.1 Logique catégorique du premier ordre

Comme précédemment, on supposera avoir un système de types $\mathbb{T}$ formé par un ensemble de types de base T et un système de constructeurs. Nous ajoutons désormais à cette structure une paire $(\mathbb{P}, \gamma)$ où $\mathbb{P}$ est un ensemble de symboles formels appelés *prédicats*, et $\gamma : \mathbb{P} \to \mathbb{T}^*$ est une fonction assignant à chaque symbole une séquence de types correspondant aux types des arguments pour ce symbole. Par abus de notation, nous écrirons $P : \sigma_1, \ldots, \sigma_n$ pour signifier $\gamma(P) = \sigma_1, \ldots, \sigma_n$ — en gardant à l'esprit cependant que les prédicats ne sont pas des termes et que le symbole « , » n'est pas un constructeur de types. Nous ajoutons à ces symboles une série de symboles distingués supplémentaires : pour chaque type $\sigma \in \mathbb{T}$, nous supposons disposer d'un prédicat binaire $=_\sigma : \sigma, \sigma$ (qu'on utilisera en notation infixe) qui représente l'égalité intensionnelle pour le type σ. Ces symboles de prédicats permettent alors de former des *propositions* à partir des termes de la théorie des types. On utilisera pour la formation de ces propositions un séquent similaire à celui du typage pour les termes, de la forme $\Gamma \vdash \varphi$, constitué sur la base des règles suivantes :

$$\frac{\Gamma \vdash u : \sigma \quad \Gamma \vdash u' : \sigma}{\Gamma \vdash u =_\sigma u'} \qquad \frac{\Gamma \vdash u_1 : \sigma_1 \quad \ldots \quad \Gamma \vdash u_n : \sigma_n \quad P : \sigma_1, \ldots, \sigma_n}{\Gamma \vdash P(u_1, \ldots, u_n)}$$

Notons cependant que cette notation est purement syntaxique, dans le sens où elle ne permet que de montrer qu'une proposition est bien construite. On notera également que les propositions ne sont pas typées dans ces séquents : en effet, les propositions sont ici vues comme extérieures aux termes dans la théorie des types. Enfin, on remarquera que les propositions ainsi formées correspondent exactement aux propositions atomiques des logiques propositionnelles. Il est bien alors possible d'ajouter des règles permettant de définir récursivement des propositions plus complexes, utilisant des connecteurs logiques supplémentaires : il suffira de placer en prémisses les conditions montrant que les sous-propositions utilisées en conclusion sont bien formées. Nous donnons ci-dessous les règles nécessaires à la négation, à la conjonction et à la quantification existentielle, en retenant que l'implication et la disjonction se construisent similairement à la conjonction, et que la quantification universelle se construit similairement à l'existentielle ; remarquons au passage que les quantificateurs consomment une variable dans le contexte de typage :

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \neg\varphi} \qquad \frac{\Gamma \vdash \varphi \quad \Gamma \vdash \psi}{\Gamma \vdash \varphi \wedge \psi} \qquad \frac{\Gamma, x : \sigma \vdash \varphi}{\Gamma \vdash \exists x : \sigma.\varphi}$$

Il ne reste alors plus qu'à construire un système déductif pour utiliser ces propostions logiques. Pour ce faire, une autre forme de séquent va être utilisée. En effet, il y a pour un tel système deux informations à rassembler : le contexte de formation d'une part, et les hypothèses logiques d'autre part ; sans parler bien entendu de la conséquence à prouver. De même que les

$$\frac{}{\Gamma \mid \varphi \vdash \varphi}\,(\text{ax}) \qquad \frac{}{\Gamma \mid \Theta \vdash \top}\,(\top_i) \qquad \frac{}{\Gamma \mid \Theta, \bot \vdash \psi}\,(\bot_e)$$

$$\frac{\Gamma \mid \Theta, \varphi \vdash \psi}{\Gamma \mid \Theta \vdash \varphi \Rightarrow \psi}\,(\Rightarrow_i) \qquad \frac{\Gamma \mid \Theta \vdash \varphi \Rightarrow \psi \quad \Gamma \mid \Theta \vdash \varphi}{\Gamma \mid \Theta \vdash \psi}\,(\Rightarrow_e)$$

$$\frac{\Gamma \mid \Theta \vdash \varphi \quad \Gamma \mid \Theta \vdash \psi}{\Gamma \mid \Theta \vdash \varphi \wedge \psi}\,(\wedge_i) \qquad \frac{\Gamma \mid \Theta \vdash \varphi \wedge \psi}{\Gamma \mid \Theta \vdash \varphi}\,(\wedge_{e1}) \qquad \frac{\Gamma \mid \Theta \vdash \varphi \wedge \psi}{\Gamma \mid \Theta \vdash \psi}\,(\wedge_{e2})$$

$$\frac{\Gamma \mid \Theta, \varphi \vdash \xi \quad \Gamma \mid \Theta, \psi \vdash \xi}{\Gamma \mid \Theta, \varphi \vee \psi \vdash \xi}\,(\vee_{li}) \qquad \frac{\Gamma \mid \Theta \vdash \varphi}{\Gamma \mid \Theta \vdash \varphi \vee \psi}\,(\vee_{i1}) \qquad \frac{\Gamma \mid \Theta \vdash \psi}{\Gamma \mid \Theta \vdash \varphi \vee \psi}\,(\vee_{i2})$$

$$\frac{\Gamma, x : \sigma \mid \Theta \vdash \psi \quad x \notin \mathrm{fv}(\Theta)}{\Gamma \mid \Theta \vdash \forall x : \sigma.\psi}\,(\forall_i) \qquad \frac{\Gamma \vdash u : \sigma \quad \Gamma \mid \Theta \vdash \forall x : \sigma.\psi}{\Gamma \mid \Theta \vdash \psi[u/x]}\,(\forall_e)$$

$$\frac{\Gamma \vdash u : \sigma \quad \Gamma \mid \Theta \vdash \psi[u/x]}{\Gamma \mid \Theta \vdash \exists x : \sigma.\psi}\,(\exists_i) \qquad \frac{\Gamma \mid \Theta \vdash \exists x : \sigma.\varphi \quad \Gamma, x : \sigma \mid \Theta', \varphi \vdash \psi \quad x \notin \mathrm{fv}(\Theta', \psi)}{\Gamma \mid \Theta, \Theta' \vdash \psi}\,(\exists_e)$$

$$\frac{\Gamma \vdash u = u' : \sigma}{\Gamma \mid \Theta \vdash u =_\sigma u'}\,(=_i) \qquad \frac{\Gamma \mid \Theta \vdash u =_\sigma u' \quad \Gamma \mid \Theta \vdash u' =_\sigma u''}{\Gamma \mid \Theta \vdash u =_\sigma u''}\,(\text{trans})$$

$$\frac{\Gamma \mid \Theta \vdash u =_\sigma u'}{\Gamma \mid \Theta \vdash u' =_\sigma u}\,(\text{sym}) \qquad \frac{\Gamma \mid \Theta \vdash u =_\sigma u' \quad \Gamma \mid \Theta \vdash \psi[u/x]}{\Gamma \mid \Theta \vdash \psi[u'/x]}\,(\text{rempl})$$

FIGURE 5.1 – Règles de déduction naturelle intuitionniste

contextes sont des séquences de variables typées, nous introduisons des séquences de propositions bien formées que nous appellerons *contextes propositionnels*. Un séquent logique est alors de la forme $\Gamma \mid \Theta \vdash \psi$ où Γ est un contexte, Θ un contexte propositionnel, et ψ une proposition. Naturellement, la formation de ces séquents est contrainte : elle est conditionnée à la bonne formation des propositions dans Θ ainsi que de ψ dans le contexte Γ ; autrement dit, si $\Theta = \varphi_1, \ldots, \varphi_n$, alors les séquents $\Gamma \vdash \varphi_i$ pour $i \in [\![1; n]\!]$ ainsi que $\Gamma \vdash \psi$ doivent être vérifiés grâce aux règles ci-dessus pour que le séquent soit bien formé. Dans la suite, nous supposerons systématiquement que les séquents considérés sont bien formés.

La construction des règles de déduction naturelle sur la base de ces séquents suit alors le schéma classique pour la logique considérée. Nous rassemblons les règles générales pour la logique intuitionniste du premier ordre, établies d'après [88, §4.1], en figure 5.1. [77] Nous y retrouvons tous les connecteurs habituels : les constantes $\top$ et $\bot$, l'implication $\Rightarrow$, la conjonction $\wedge$, la disjonction $\vee$ ainsi que les quantificateurs $\forall$ et $\exists$. On relèvera sans doute l'absence de la négation dans ces règles : tout ce qui y a trait peut en effet être simplement déduit de la définition $\neg\varphi = \varphi \Rightarrow \bot$. On observera également plusieurs règles ayant trait à l'égalité logique : la règle $(=_i)$ permet de convertir la théorie équationnelle en égalités logiques et établit par conséquent que les égalités logiques sont réflexives, les règles (sym) et (trans) imposent la symétrie et la transitivité comme

77. Ces règles doivent évidemment complétées des traditionnelles règles d'affaiblissement, de contraction et d'échange dans les contextes de typage et les contextes propositionnels, ainsi que de la substitution d'une variable par un terme de même type, que nous ne reproduisons pas ici.

attendu pour une égalité, et la règle de remplacement (rempl) permet de propager des égalités dans toute proposition. Cette construction de la logique peut également être complétée par une *axiomatique*, c'est-à-dire une collection de séquents qui sont supposés dérivables sans conditions supplémentaires, et peuvent servir de points de départ pour d'autres preuves. Enfin, signalons qu'il est possible de récupérer une logique classique en ajoutant à ce système la règle de réduction par l'absurde suivante :

$$\frac{\Gamma \mid \Theta, \neg\varphi \vdash \bot}{\Gamma \mid \Theta \vdash \varphi}\,(\text{raa})$$

Venons-en maintenant à l'interprétation de cette logique : comment peut-elle être capturée par des modèles catégoriques ? La réponse des catégoriciens (cf. p. ex. [88, 177]) consiste à interpréter les propositions comme des sous-objets dans le modèle : si $\mathcal{C}\ell$ est la catégorie classificatrice de notre théorie et que $[\![.]\!] : \mathcal{C}\ell \to \mathcal{C}$ en est un modèle quelconque, on définira ainsi l'interprétation d'une proposition bien formée $\Gamma \vdash \varphi$ comme un sous-objet $[\![\varphi]\!] : X \rightarrowtail [\![\Gamma]\!]$. Cependant, la capacité du modèle à interpréter certaines propositions dépend encore une fois de ses propriétés intrinsèques. Dans toute catégorie suffisante pour modéliser notre théorie des types, les prédicats correspondent ainsi à des relations internes (cf. définition 33) dans le modèle : l'interprétation $[\![P]\!]$ de $P : \sigma_1, \ldots, \sigma_n$ est un sous-objet de $[\![\sigma_1]\!] \times \cdots \times [\![\sigma_n]\!]$.[78] Étant donné un contexte Γ, les interprétations qu'il est possible de construire dépendent des propriétés de l'ensemble partiellement ordonné $\mathrm{Sub}([\![\Gamma]\!])$. Par définition, cet ensemble possède toujours un plus grand élément, représenté par $\mathrm{id}_{[\![\Gamma]\!]}$: en terme de propositions, on assimilera ce sous-objet à la constante $\top$ qui caractérise la valeur vraie.

Les autres connecteurs ne sont en revanche interprétables qu'à certaines conditions. Ainsi, la conjonction ne peut être interprétée que si $\mathrm{Sub}([\![\Gamma]\!])$ est un *inf-demi-treillis*, c.-à-d. si toute paire d'éléments a, b admet une borne inférieure $a \cap b$; on aura alors $[\![\varphi \wedge \psi]\!] = [\![\varphi]\!] \cap [\![\psi]\!]$. La quantification existentielle s'appuie quant à elle sur la projection $f : [\![\Gamma, x : \sigma]\!] \to [\![\Gamma]\!]$, et requiert pour son interprétation l'existence d'un adjoint à gauche $\exists_f$ au foncteur de changement de base $f^* : \mathrm{Sub}([\![\Gamma]\!]) \to \mathrm{Sub}([\![\Gamma, x : \sigma]\!])$ induit par f.[79] Pour interpréter la disjonction, l'ensemble des sous-objets doit également posséder les bornes supérieures $a \cup b$ pour tous éléments a, b ; et pour la constante $\bot$, il doit avoir un plus petit élément $0_{[\![\Gamma]\!]}$. L'interprétation de la quantification universelle est accesible si le foncteur de changement de base f^* défini précédemment admet aussi un adjoint à droite $\forall_f$, et l'implication est alors obtenue si $\mathrm{Sub}([\![\Gamma]\!])$ est une algèbre de Heyting, où $a \Rightarrow b$ est le pseudo-complément de a relativement à b.[80] Le cas de l'interprétation de l'égalité intensionnelle est un peu particulier : il requiert l'existence d'égalisateurs. En effet, on souhaite interpréter $u =_\sigma u'$ comme vrai lorsque $[\![u]\!] = [\![u']\!]$, où les deux interprétations sont

78. Rappelons qu'un modèle catégorique dispose du produit même si le système de constructeur n'en comporte pas, cf. [88, §2].

79. Ce foncteur de changement de base est induit par le foncteur de produit fibré introduit au chapitre 4. En effet, pour $X \in \mathsf{obj}(\mathcal{C})$ la catégorie des monomorphismes de codomaine X est une sous-catégorie de la catégorie tranche $\mathcal{C} \downarrow X$, et comme le foncteur de produit fibré préserve les monomorphismes, il induit un autre foncteur sur ces derniers. Or, comme tout foncteur préverse les isomorphismes, ce nouveau foncteur s'accomode parfaitement du passage aux classes d'équivalence qui permet de former les sous-objets.

80. Il est possible de reconnaître dans l'ordre de présentation différentes formes de logiques, qui sont souvent associées à des classes de catégories particulières. On appelle notamment *catégories régulières* les catégories pouvant interpréter les connecteurs $\top$, $\wedge$ et $\exists$, *catégories cohérentes* celles pouvant en plus interpréter $\vee$ et $\bot$, et *catégories de Heyting* celles pouvant interpréter tous les connecteurs introduits ici, et donc la logique intuitionniste. Notons par ailleurs qu'il suffit d'être finiment complète pour interpréter la conjonction (qui s'obtient alors par produit fibré), et qu'il suffit d'être cartésienne fermée munie d'un objet initial pour interpréter $\bot$, puisque tout morphisme $0 \to X$ est alors mono par la proposition 17. Notons enfin, comme discuté p. ex. dans [75], que tout topos est une catégorie de Heyting.

des morphismes $[\![\Gamma]\!] \to [\![\sigma]\!]$, mais par souci de cohérence, nous voudrions obtenir ce résultat sous la forme d'un sous-objet de $[\![\Gamma]\!]$. L'interprétation $[\![u =_\sigma u']\!]$ est alors obtenue en prenant l'égalisateur de $[\![u]\!]$ et $[\![u']\!]$, qui est bien un monomorphisme (et donc représente un sous-objet de $[\![\Gamma]\!]$) d'après la proposition 11. [81]

L'interprétation des conséquences logiques est alors liée à l'ordre des ensembles de sous-objet : on dira qu'un jugement logique $\Gamma \mid \Theta \vdash \psi$ avec $\Theta = \varphi_1, \ldots, \varphi_n$ est *valide* dans le modèle catégorique $\mathcal{C}$ lorsque $[\![\bigcap_{i=1}^n \varphi_i]\!] \leq [\![\psi]\!]$, où $\leq$ est l'ordre sur $\mathrm{Sub}([\![\Gamma]\!])$ et $\cap$ son opérateur de borne inférieure. Pour interpréter les théorèmes de la logique, on assimilera simplement les contextes propositionnels vides à la constante $\top$, de sorte que $\Gamma \mid \varnothing \vdash \psi$ est valide si $[\![\top]\!] \leq [\![\psi]\!]$ dans $\mathrm{Sub}([\![\Gamma]\!])$, ce qui se traduit le cas échéant par $[\![\psi]\!] = [\![\top]\!]$. En particulier, ceci permet bien de récupérer l'égalité intensionnelle de deux termes $\Gamma \vdash u : \sigma$ et $\Gamma \vdash u' : \sigma$: par définition de l'égalisateur, on a $[\![u]\!] \circ [\![u =_\sigma u']\!] = [\![u']\!] \circ [\![u =_\sigma u']\!]$, donc si cette égalité est valide, cela signifie que $[\![u =_\sigma u']\!] = [\![\top]\!] = \mathrm{id}_{[\![\Gamma]\!]}$, d'où $[\![u]\!] = [\![u']\!]$.

Il va sans dire que cette interprétation de la logique est correcte et complète [88, §4.3]. Nous disposons donc désormais d'un moyen utiliser les modèles catégoriques comme modèles d'un système de logique du premier ordre construit autour de nos théories de types. Avons-nous désormais tous les éléments en main pour interpréter correctement le système de Montague ? Presque, mais une différence subsiste, et non des moindres : le système de Montague inclut dans sa théorie un type des propositions logiques explicite, le type t traditionnellement décrit comme le type des valeurs de vérité. Cela signifie que la logique de ce système est internalisée : les formules logiques sont traitées comme des termes de la théorie, et il faut en faire de même des conséquences logiques exprimables à l'aide de ces formules. Nous allons donc dans la suite étudier plus en détail comment l'interprétation de la logique abordée précédemment peut s'adapter à un tel système et maintenir une interprétation cohérente.

5.3.2 Un modèle pour le système montagovien

Reprenons pour point de départ la théorie des types du système de Montague : [82] la *théorie des types simples montagovienne* MSTT est la théorie des types simples construite sur le système de types $\mathbb{T}$ librement engendré par l'ensemble des types de base $\{e, t\}$ et le système de constructeur $\{\to (2), \times(2), 1\,(0)\}$. Le sous-ensemble $\mathbb{P} \subset \mathbb{T}$ des *types de prédicats montagovien* est défini par les grammaires suivantes :

$$\begin{aligned} \Pi &::= e \mid \mathbb{P} \mid \Pi \times \Pi \\ \mathbb{P} &::= t \mid \Pi \to \mathbb{P} \end{aligned}$$

Naturellement, les types de prédicats peuvent être vus comme équivalents à curryfication près : nous ferons l'hypothèse, cohérente avec l'approche traditionnelle de la sémantique montagovienne, que les types de prédicats sont présentés sous leur forme curryfiée par défaut. Alors,

81. Il peut être légitime de se demander si l'égalité $=_\sigma$ est liée au morphisme diagonal $\delta_{[\![\sigma]\!]}$, qui représente exactement la relation d'égalité sur $[\![\sigma]\!]$. C'est en effet le cas, par une propriété d'interchangeabilité entre égalisateurs et produits fibrés alternative à la proposition 14 : pour tous morphismes $f, g : A \to B$, le tiré de $\langle f, g\rangle$ le long de δ_B est isomorphe à l'égalisateur de f et g.

82. Comme dans le chapitre 1, nous éliminons ici toute la complexité inhérente à la logique intensionnelle spécifiquement développée par Montague : concrètement, nous retirons le type des mondes possibles s ainsi que les connecteurs logiques temporels et de nécessité, pour ne garder qu'un système simple en logique d'ordre supérieur. Par ailleurs, nous nous permettons d'enrichir le système de types des constructeurs $\times$ et 1, déjà pour le confort qu'ils apportent pour nos modèles catégoriques, mais aussi parce que nous savons depuis nos conclusions en sous-section 3.3.3 que le type produit sera nécessaire dans notre calcul final.

l'ensemble des constantes de MSTT contient une collection d'entités possédant le type e, des prédicats ayant un type dans $\mathbb{P}$, et les connecteurs logiques suivants :

$$\top, \bot : t \qquad \neg : t \to t \qquad \wedge, \vee, \Rightarrow : t \to t \to t \qquad \exists, \forall : (e \to t) \to t$$

Par ailleurs, pour tout type $\sigma \in \mathbb{T}$, on supposera disposer d'une constante $=_\sigma : \sigma \to \sigma \to t$. On maintiendra pour les connecteurs binaires un emploi infixe. Toutes ces constantes se substituent alors à l'ensemble des symboles prédicats utilisé dans la sous-section précédente.

Soient $\mathcal{C}\ell_M$ la catégorie classificatrice associée à cette théorie et $[\![.]\!] : \mathcal{C}\ell_M \to \mathcal{C}$ un modèle quelconque de cette théorie. Les constantes logiques, comme on a pu le voir dans la section 5.2, sont normalement interprétées comme des éléments globaux dans ce modèle. Cependant, suivant l'idée intuitive développée plus loin dans cette même section, nous allons privilégier leur interprétation en contexte après application à des variables, permettant ainsi d'assimilier l'application dans la théorie à la composition dans le modèle. Par exemple, la négation ne sera pas interprétée d'après le jugement $\vdash \neg : t \to t$, mais d'après $p : t \vdash \neg p : t$. Par abus de notation, on notera tout de même son interprétation comme $[\![\neg]\!] : T \to T$ plutôt que $[\![\neg p]\!]$. De façon similaire, on aura par exemple $[\![\wedge]\!] : T \times T \to T$ ainsi que $[\![\exists]\!] : T^E \to T$. Évidemment, cette remarque ne s'applique pas aux propositions constantes $\top$ (vrai) et $\bot$ (faux), dont les interprétations restent de type $1 \to T$.

Comme t correspond au type des propositions logiques, la construction de ces propositions est alors ramenée à des séries d'applications. Par exemple, pour construire une proposition d'égalité entre deux termes u et u' de type σ, nous procédons selon la dérivation suivante :

$$\dfrac{\dfrac{\dfrac{=_\sigma \in Q}{\Gamma \vdash =_\sigma : \sigma \to \sigma \to t}\,(\text{const}) \qquad \Gamma \vdash u : \sigma}{\Gamma \vdash u =_\sigma : \sigma \to t}\,(\text{app}) \qquad \Gamma \vdash u' : \sigma}{\Gamma \vdash u =_\sigma u' : t}\,(\text{app})$$

Il en va de même pour l'utilisation des prédicats, et pour la construction de toutes les formules utilisant les connecteurs de la logique propositionnelle.

La sémantique montagovienne est souvent présentée comme limitée dans l'utilisation de ses quantificateurs : comme suggéré par l'introduction des connecteurs logiques plus haut, il n'est possible de quantifier que sur le type e, et non pas sur tous les types de $\mathbb{T}$ comme dans la sous-section précédente. Cette caractéristique permet en fait de maintenir la logique montagovienne au premier ordre, puisque e est le seul type du système à ne pas dépendre du type des propositions t. Autoriser la quantification sur d'autres types implique donc de pouvoir quantifier sur des propositions et des prédicats, une caractéristique des *logiques d'ordre supérieur*. En acceptant cette augmentation de puissance, il est alors possible d'étendre le système pour disposer de constantes $\forall_\sigma, \exists_\sigma : (\sigma \to t) \to t$ pour tout type $\sigma \in \mathbb{T}$.[83] La construction d'une formule quantifiée passe dans tous les cas par une lambda-abstraction suivie d'une application :

$$\dfrac{\dfrac{\exists \in Q}{\Gamma \vdash \exists : (e \to t) \to t}\,(\text{const}) \qquad \dfrac{\Gamma, x : e \vdash \varphi : t}{\Gamma \vdash \lambda x.\varphi : e \to t}\,(\text{lam})}{\Gamma \vdash \exists\,(\lambda x.\varphi) : t}\,(\text{app})$$

83. En relisant attentivement l'article de Montague [120], il s'avère qu'utiliser une logique d'ordre supérieure est cohérente avec l'esprit de son système. En effet, Montague n'exclut pas la possibilité de quantifications d'ordre supérieur dans la construction de sa logique intensionnelle [120, §2], et n'hésite pas à formuler une axiomatique qui utilise de telles quantifications [120, §3].

Dans le cas où la quantification est limitée au type e, aucune ambiguïté n'est possible, aussi nous noterons $\exists x.\varphi$ au lieu de $\exists\,(\lambda x.\varphi)$, et de même pour la quantification universelle. Et dans un cadre plus général, on notera $\exists x\!:\!\sigma.\varphi$ pour $\exists_\sigma\,(\lambda x.\varphi)$.

Une conséquence majeure de l'internalisation des propositions logiques est la dualité de l'interprétation de ces propositions. En effet, si $\Gamma \vdash \varphi : t$ est dérivable dans notre théorie de types, deux options s'offrent à nous : en suivant l'approche présentée dans la sous-section précédente, φ peut être interprété comme un sous-objet $X \rightarrowtail [\![\Gamma]\!]$, mais comme φ est aussi un terme, une interprétation sous forme de morphisme $[\![\Gamma]\!] \to T$ est également disponible. Il est alors possible d'observer une correspondance systématique entre ces sous-objets et ces morphismes, puisque chaque connecteur logique a une interprétation comme opération des algèbres de sous-objets et une interprétation comme morphisme issu de la constante correspondante. Cette correspondance peut potentiellement induire dans certains modèles une bijection $\mathrm{Sub}(X) \cong \mathcal{C}(X, T)$, et il n'est pas impossible que cette bijection puisse être naturelle en X dans de tels modèles si les bonnes propriétés sont réunies. Dans les modèles où c'est le cas, cela signifie que le foncteur $\mathrm{Sub}(-)$ est représentable par T, ce qui fait alors de T un classificateur de sous-objet par la proposition 18.

Si nous cantonnons ce raisonnement à quelques modèles dans notre développement du paragraphe précédent, il est en fait possible d'aller plus loin : comme montré notamment par Jacobs [88, §5.3], tout modèle de la logique d'ordre supérieur qui interprète les propositions comme des sous-objets [84] interprète le type des propositions comme un tel classificateur de sous-objets. Et comme les modèles de la théorie des types simples sont des catégories cartésiennes fermées, nous en venons au constat suivant :

Proposition 31. *Les modèles de la théorie des types simples montagovienne sont des topos, et le type des propositions t y est interprété par leurs classificateurs de sous-objets.*

Ce résultat est cohérent avec les équivalences démontrées entre les topos et les logiques d'ordre supérieur [101], la partie la plus importante ici étant de se rendre compte que la sémantique montagovienne est bel et bien d'ordre supérieur, par constrate avec la présentation traditionnelle qui limite la quantification au seul type des entités e. Le lien entre modèles de la langue naturelle et topos n'est cependant pas nouveau : on le retrouve notamment chez Asher [4, §4.5 et §5.3] qui exploite notamment la complète finitude des topos pour modéliser les objets pointés sous forme de produits fibrés pour son système TCL. [85] En revanche, à notre connaissance, il ne semble pas que Asher se soit intéressé à l'interprétation du type t des propositions comme classificateur de sous-objet et aux conséquences que cette interprétation pourrait avoir. Par extension, les modèles catégoriques des théories de types dédiées à la linguistique ont été somme toute assez peu étudiés, [86] et encore moins le rôle des topos dans cette perspective.

En tant que modèle général pour la logique d'ordre supérieur, les topos offrent une grande puissance d'interprétation ouvrant la voie à la construction de systèmes plus expressifs. Nous avons justement discuté au chapitre 1 de la nécessité de renforcer le pouvoir expressif du système de Montague par l'ajout de caractéristiques nouvelles : ontologie de types, constructeurs, coercions, etc. Si les topos peuvent facilement interpréter la théorie montagovienne originale, il est

84. Il est en effet possible de donner des interprétations alternatives des propositions, dans des catégories différentes suivant la construction encore plus générale de *fibration*, largement étudiée dans le livre de Jacobs. Nous ne développerons pas cette notion dans le cadre de cette thèse, bien que le sujet mériterait d'être exploré davantage.

85. Nous reviendrons à cette application spécifique en section 6.3.

86. Pour quelques applications de la théorie des catégories dans le domaine de la linguistique informatique, on pourra consulter les travaux de La Palme Reyes [99] ainsi que de Sadrzadeh, Kartsaklis, Coecke et leurs collaborateurs [44, 160].

plutôt naturel de penser que la puissance des topos pourrait être exploitée afin d'étendre cette théorie sans pour autant modifier la classe des modèles catégoriques possibles. Avant de commencer à explorer plus en détail cette perspective au chapitre 6, la section suivante se propose de fournir davantage d'intuition sur la manière dont les modèles de topos peuvent influencer l'amélioration des théories de types sous-jacentes.

5.4 Inverser la dynamique : des modèles aux théories

L'introduction aux modèles catégoriques des théories de types au fil de ce chapitre s'est faite dans une certaine direction : étant donnée une théorie des types, nous avons cherché à construire une représentation catégorique de celle-ci et d'étudier les propriétés de ses modèles, menant ainsi à la caractérisation de classes de catégories pouvant modéliser la théorie, comme les catégories cartésiennes fermées pour la théorie des types simple ou les topos pour la logique d'ordre supérieure. Il est cependant possible de se poser la question inverse : étant donnée une classe de catégories, quelles seraient les théories de types y admettant des modèles ?

La question est loin d'être anodine, car les paramètres qu'il est possible de modifier dans une théorie de types sont nombreux : types, constructeurs, règles de typage, règles équationnelles, règles de déduction logique... Bien sûr, certains de ces paramètres contraignent leurs modèles : ainsi, l'ensemble des types de base forcera l'existence d'un certain nombre d'objets indépendants dans le modèle final, et certaines règles équationnelles ou logiques peuvent forcer l'égalité de certains morphismes, et ainsi de suite. Mais, comme sous-entendu avec l'existence des théories équationnelles intensionnelles et extensionnelles, certaines composantes d'une sous-classe des modèles possibles peuvent exister sans être nécessairement accessibles comme des interprétations de la théorie initialement choisie. La présente section se propose donc d'explorer les possibilités de modifier une théorie tout en maintenant les propriétés universelles de ses modèles, afin d'ouvrir la voie vers des extensions de la théorie de Montague.

5.4.1 Reformulations admissibles

Comme affirmé juste au-dessus, certaines composantes d'un modèle donné peuvent très bien être inaccessibles en tant qu'interprétation d'une théorie donnée. Rappellons notamment que la théorie équationnelle intensionnelle d'une théorie de types est l'ensemble des égalités qui utilisent le connecteur $=_\sigma$ introduit en sous-section 5.3.1 et qui sont prouvées à partir de la logique catégorique sur la base d'une axiomatique bien choisie et sans autres hypothèses. Si $\mathcal{F}$ désigne la classe des modèles de la théorie initiale, il existe alors une sous-classe $\mathcal{F}'$ de $\mathcal{F}$ des modèles qui satisfont effectivement cette axiomatique. Si cette théorie initiale n'est pas extensionnelle, c.-à-d. s'il existe des égalités $u =_\sigma u'$ valides qui ne sont pas accessibles comme des conversions sous la forme $u = u' : \sigma$, alors il existe en particulier des modèles de cette théorie qui ne sont pas dans $\mathcal{F}'$, rendant l'inclusion $\mathcal{F}' \subseteq \mathcal{F}$ stricte. Supposons maintenant que nous intégrons toutes les égalités de l'axiomatique comme des conversions : on obtient alors une nouvelle théorie des types qui est exactement une théorie extensionnelle, avec $u = u' : \sigma$ si et seulement si $u =_\sigma u'$ est valide, et la classe des modèles de cette nouvelle théorie est alors réduite à $\mathcal{F}'$.

Dans cet exemple, jouer avec ce qui relève de la théorie elle-même ou de sa logique permet de raffiner la classe des modèles possibles vers une de ses sous-classes. Dans l'hypothèse où la classe initiale était caractérisée par certaines propriétés universelles, cela signifie que les modèles de la nouvelle théorie, en tant que sous-classe de la première, possèdent également ces propriétés. À l'inverse, si l'on supprimait des égalités dans la théorie équationnelle, on risquerait d'élargir la classe des modèles possibles, et possiblement de perdre des caractérisations communes à tous les

modèles. Par exemple, retirer la règle (β) de β-conversion de la théorie des types simples telle que présentée en section 5.2 aurait pour conséquence d'autoriser des modèles qui ne seraient pas nécessairement des catégories cartésiennes fermées.

De manière générale, notre objectif sera donc d'étendre notre théorie de types de façon à maintenir les propriétés universelles de ses modèles, c.-à-d. en raffinant plus qu'en élargissant la classe des modèles. On retrouve dans cette approche l'idée d'*extension conservatrice* d'une théorie : il est attendu notamment que chaque terme construit avec les éléments présents dans la théorie initiale ait le même type dans la théorie initiale et dans la théorie modifiée, et que les égalités basées sur ces termes soient aussi les mêmes dans les deux théories. Certaines modifications de la théorie initiale respectent cette condition sans difficulté : c'est par exemple le cas lors de l'ajout de constructeurs supplémentaires, à la condition suffisante que les règles d'introduction et d'élimination des termes associés n'impliquent pas des constructeurs déjà existants ; c'est également le cas pour le simple ajout de types de base additionnels et de constantes les utilisant, puisqu'ils ne sont pas impliqués dans la construction des termes de la théorie initiale. Du point de vue des modèles, cela revient à ajouter ou exploiter des propriétés universelles (qui caractérisent généralement les constructeurs les plus répandus) ou tout simplement des objets et des morphismes. Il faut néanmoins faire attention aux égalités qui peuvent résulter de ces ajouts, car c'est sur ces dernières que des différences peuvent apparaître entre les théories.

Dans le cas qui nous intéresse, la possibilité d'étendre la théorie des types de Montague repose sur l'idée que les topos qui lui servent de modèles sont loin d'être exploités au maximum de leur potentiel, dans le sens où certaines de leurs propriétés intrinsèques ne sont pas utilisées du tout dans le cadre de la modélisation de cette théorie. Sans le dire, nous avons déjà exploité un phénomène similaire lors des discussions de ce chapitre : la théorie des types simples, qui ne s'appuie que sur des types de base quelconques et le constructeur de types fonctions, a en effet pour modèles les catégories cartésiennes fermées — et cela est en fait indépendant de si la théorie inclut ou non les types produits et les paires de termes. L'introduction des constructeurs de types $\times$ et 1, des paires, des projections et des égalités associées étend ainsi la théorie des types simples de façon conservatrice et sans modifier ses modèles, qui sont déjà cartésiens. C'est pourquoi nous nous sommes permis de les introduire directement dans notre théorie, sans perte de généralité.

Nous allons cependant considérer des transformations légèrement plus générales afin de nous adapter aux ccaractéristiques de la modélisation sémantique linguistique qui guident notre construction d'une théorie des types. En ce sens, nous ne nous reposerons pas directement sur la notion d'extension convervatrice des théories de types, telle que définie par exemple dans [192]. Nos besoins incluent en effet d'une part la possibilité de modifier certains constructeurs de types (nous défendrons ce besoin en section 6.1), et d'autre part l'impératif de conserver les propriétés des modèles catégoriques. Nous introduisons pour cela une notion plus flexible de *reformulation admissible* pour les théories de types, et en profitons pour établir quelques propriétés préliminaires. Pour les besoins de ces définitions très générales, nous définissons une théorie de type $\mathscr{T}$ comme un quintuplet $(\mathbb{T}, Q, \mathfrak{t}, \Lambda, \mathcal{R})$ où $\mathbb{T}$ est un système de types, $(Q, \mathfrak{t})$ une signature de constantes, Λ un ensemble de termes (avec $Q \subseteq \Lambda$) et $\mathcal{R}$ l'ensemble des règles permettant d'établir les jugements de typage et d'égalité. On notera $\Gamma \vdash_{\mathscr{T}} u : \sigma$ et $\Gamma \vdash_{\mathscr{T}} u = u' : \sigma$ si les jugements $\Gamma \vdash u : \sigma$ et $\Gamma \vdash u = u' : \sigma$ sont dérivables à l'aide des règles de la théorie $\mathscr{T}$.

Définition 38. *Soient $\mathscr{T}_1$ et $\mathscr{T}_2$ deux théories de types. Une* reformulation *de $\mathscr{T}_1$ à $\mathscr{T}_2$ est la donnée d'un triplet $\theta = (\theta_1, \theta_2, \theta_3)$ tel que :*

— *$\theta_1 : \mathbb{T}_1 \to \mathbb{T}_2$ est une fonction ;*

— *$\theta_2 : Q_1 \to Q_2$ est une fonction telle que $\mathfrak{t}_2(\theta_2(q)) = \theta_1(\mathfrak{t}_1(q))$;*

— $\theta_3 : \Lambda_1 \to \Lambda_2$ *est une fonction qui coïncide avec* θ_2 *sur* Q_1 *et telle que pour tout terme* $u \in \Lambda_1$, $\mathrm{fv}(\theta_3(u)) \subseteq \mathrm{fv}(u)$;
— *si* $\Gamma \vdash_{\mathscr{T}_1} u : \sigma$ *alors* $\theta(\Gamma) \vdash_{\mathscr{T}_2} \theta_3(u) : \theta_1(\sigma)$ *où, pour* $\Gamma = x_1 : \sigma_1, \dots x_n : \sigma_n$, *on définit* $\theta(\Gamma) = x_1 : \theta_1(\sigma_1), \dots, x_n : \theta_1(\sigma_n)$;
— *si* $\Gamma \vdash_{\mathscr{T}_1} u = u' : \sigma$ *alors* $\theta(\Gamma) \vdash_{\mathscr{T}_2} \theta_3(u) = \theta_3(u') : \theta_1(\sigma)$.

On relèvera que cette notion de reformulation, qui n'est autre qu'un foncteur entre catégories classificatrices qui préserve les produits, n'utilise pas d'homomorphisme pour les types ou les termes, ce qui permet de relever certaines limites qu'une telle contrainte pourrait imposer ; néanmoins, il est attendu que la transformation des termes soit influencée par les conditions de préservation des termes typables, et c'est pourquoi en particulier cette transformation interdit d'introduire des variables libres supplémentaires. De manière plus générale, il sera relativement commun que les reformulations induisent des transformations de dérivation de typage, où certaines séquences d'applications de règles sont remplacées par d'autres utilisant au moins les mêmes prémisses. Nous laissons cependant davantage de liberté sur la transformation des types. On notera $\theta : \mathscr{T}_1 \to \mathscr{T}_2$ pour indiquer que θ est une reformulation de $\mathscr{T}_1$ à $\mathscr{T}_2$, et on remarquera que ces reformulations peuvent être composées et qu'il existe pour toute théorie une reformulation identité sur celle-ci.

Exemple 11. *Lorsque les trois fonctions* θ_1, θ_2 *et* θ_3 *sont des identités, la reformulation est tout simplement une* extension *de la théorie des types initiale* (cf. [192]).

Exemple 12. *Si les théories considérées sont des lambda-calculs linéaires et que les composantes sont des homomorphismes, alors la structure formée par ces deux théories et la reformulation en question est une grammaire catégorielle abstraite* (cf. [49]).

Exemple 13. *Soit* $\mathrm{MSTT} = (\mathbb{T}, Q, \mathfrak{t}, \Lambda, \mathcal{R})$ *la théorie des types simples montagovienne telle qu'introduite en sous-section 5.3.2. Une liste de types de* $\mathbb{T}$ *est définie comme un mot dans le monoïde* $(\mathbb{T}^*, \cdot)$ *où* * *est l'étoile de Kleene et* $\cdot$ *l'opération associative de concaténation ; la liste vide sera désignée par* ε. *La fonction d'applatissement* $\mathrm{fl} : \mathbb{T} \to \mathbb{T}^*$ *est définie par* $\mathrm{fl}(\sigma) = \sigma$ *si* σ *n'est pas un produit et* $\mathrm{fl}(\sigma \times \tau) = \mathrm{fl}(\sigma) \cdot \mathrm{fl}(\tau)$ *sinon. Alors, pour tout* $\ell \in \mathbb{T}^*$, *définissons la fonction* $f_\ell : \mathbb{T} \to \mathbb{T}$ *comme suit :*

$$f_\ell(\sigma \to \tau) = f_{\ell'}(\tau) \quad \text{où } \ell' = \ell \cdot \mathrm{fl}(f_\varepsilon(\sigma))$$

$$f_\ell(\sigma \times \tau) = \begin{cases} f_\varepsilon(\sigma) \times f_\varepsilon(\tau) & \text{si } \ell = \varepsilon \\ \xi_1 \times \cdots \times \xi_n \to (f_\varepsilon(\sigma) \times f_\varepsilon(\tau)) & \text{si } \ell = \xi_1, \dots, \xi_n \end{cases}$$

$$f_\ell(b) = \begin{cases} b & \text{si } \ell = \varepsilon \\ \xi_1 \times \cdots \times \xi_n \to b & \text{si } \ell = \xi_1, \dots, \xi_n \end{cases} \qquad \text{pour } b \in \{e, t, 1\}$$

et posons $\theta_1 = f_\varepsilon$. *Soit alors* MSTT' *la théorie exactement similaire à* MSTT *excepté pour* $\mathfrak{t}$ *qui est remplacé par* $\mathfrak{t}' = \theta_1 \circ \mathfrak{t}$. *Posons* θ_2 *l'identité sur* Q *et définissons ensuite la fonction* $\theta_3 : \Lambda \to \Lambda$ *telle que :*

$$\begin{aligned} \theta_3(*) &= * \\ \theta_3(x) &= x \text{ pour } x \in V \\ \theta_3(q) &= q \text{ pour } q \in Q \\ \theta_3(\langle u, v \rangle) &= \langle \theta_3(u), \theta_3(v) \rangle \\ \theta_3(\pi_i u) &= \pi_i(\theta_3(u)) \quad \text{pour } i \in \{1, 2\} \\ \theta_3(\lambda x_1 \dots \lambda x_n . u) &= \lambda x . (\theta_3(u)[\pi_1 x / x_1, \dots, \pi_n x / x_n]) \text{ pour } u \notin \lambda V . \Lambda \\ \theta_3((u\, v_1) \dots v_n) &= \theta_3(u) \langle \theta_3(v_1), \dots, \theta_3(v_n) \rangle \text{ pour } u \notin \Lambda\Lambda \end{aligned}$$

où les deux dernières conditions imposent respectivement que u n'est pas une lambda-abstraction dans le premier cas, et n'est pas une application dans le second cas. Alors, le triplet $(\theta_1, \theta_2, \theta_3)$ induit une reformulation $\mathrm{dcr} : \mathrm{MSTT} \to \mathrm{MSTT}'$ *appelée* décurryfication. *Nous laissons le soin au lecteur de vérifier que les jugements dérivables sont transformés comme attendus et de déterminer quelles transformations de dérivations de typage sont induites par cette reformulation.*

Nous définissons maintenant un cas particulier de reformulation qui prend en compte les contraintes que nous souhaitons imposer à nos modifications des théories de types, à savoir un caractère conservatif dans les typages et égalités possibles ainsi que la préservation des propriétés des modèles catégoriques associés.

Définition 39. *Une reformulation $\theta : \mathscr{T}_1 \to \mathscr{T}_2$ est dite* admissible *si les propriétés suivantes sont satisfaites :*

- *pour tout terme $u \in \Lambda_1$, s'il existe Γ et τ tels que $\Gamma \vdash_{\mathscr{T}_2} \theta(u) : \tau$ et $\tau \in \mathrm{im}(\theta)$, alors il existe Δ et σ tels que $\Delta \vdash_{\mathscr{T}_1} u : \sigma$, $\Gamma \subseteq \theta(\Delta)$ et $\theta(\sigma) = \tau$;*
- *pour tout termes $u, u' \in \Lambda_1$, s'il existe Γ et τ tels que $\Gamma \vdash_{\mathscr{T}_2} \theta(u) = \theta(u') : \tau$ et $\tau \in \mathrm{im}(\theta)$, alors il existe Δ et σ tels que $\Delta \vdash_{\mathscr{T}_1} u = u' : \sigma$, $\Gamma \subseteq \theta(\Delta)$ et $\theta(\sigma) = \tau$;*
- *tout modèle catégorique de $\mathscr{T}_2$ est aussi un modèle de $\mathscr{T}_1$.*

Par abus de langage, on dira qu'une théorie est une reformulation admissible d'une autre s'il existe une telle reformulation de la seconde à la première. La condition $\tau \in \mathrm{im}(\theta)$ est une contrainte qui permet de s'assurer que l'introduction de nouveaux types de base ou de constructeurs n'empêche pas une théorie d'être une reformulation admissible d'une autre, même lorsque de nouveaux typages utilisant ces types apparaissent pour des termes dans l'image de la théorie initiale. Quant à la condition d'inclusion entre les contextes, elle rappelle que l'image d'un terme par une reformulation peut disposer de moins de variables libres que le terme initial, mais qu'aucune nouvelle variable libre n'y est utilisée. Il y aurait beaucoup à dire sur la théorie générale des théories de types et reformulations esquissée ici, mais son étude en profondeur ne rentre pas dans le cadre de la présente thèse. En particulier, il n'est pas impossible que la troisième condition dans cette définition soit superflue ou puisse être capturée par des conditions plus faibles, mais nous estimons que la définition telle que donnée ici sera suffisante pour nos objectifs.

Exemple 14. *Si une reformulation admissible est en plus une extension, c.-à-d. que ses composantes sont des identités, alors elle est une extension conservatrice.*

Exemple 15. *La décurryfication* $\mathrm{dcr} : \mathrm{MSTT} \to \mathrm{MSTT}'$ *introduite ci-dessus n'est* pas *admissible. En effet, les termes $\lambda x \lambda y.y : e \to e \to t$ et $\lambda z.\pi_1 z : e \times e \to t$ sont distincts dans* MSTT, *mais sont envoyés par la décurryfication sur le même terme $\lambda z.\pi_1 z$. En particulier, par réflexivité on peut prouver :*

$$\vdash_{\mathrm{MSTT}'} \mathrm{dcr}(\lambda x.\lambda y.y) = \mathrm{dcr}(\lambda z.\pi_1 z) : e \times e \to t$$

mais il n'est pas possible de prouver une égalité entre les termes initiaux dans MSTT.

Dans les exemples suivants, nous abordons deux cas très importants de reformulations admissibles de MSTT : l'ajout de types de base, et l'introduction de constructeurs accessibles par application des propriétés des topos comme modèles de cette théorie.

Exemple 16. *Soit B un ensemble de types de base tel que $B \cap \{e, t\} = \varnothing$, et soit* $\mathrm{MSTT} \cup B$ *la théorie obtenue par l'ajout de B aux types de base du système de types de* MSTT, *par l'ajout*

éventuel de constantes utilisant ces nouveaux types, ainsi que par l'ajout éventuel d'égalités sur des termes utilisant ces nouvelles constantes, le reste étant engendré comme dans MSTT. *Alors,* MSTT $\cup B$ *est une reformulation admissible et une extension de* MSTT. *En particulier, les modèles de cette reformulation sont les modèles de* MSTT *possédant des objets supplémentaires pour interpréter les types dans* B.

Exemple 17. *D'après la proposition 19, les topos sont finiment cocomplets, ce qui signifie notamment qu'ils admettent un objet initial* 0 *ainsi que tous les coproduits. Soit* MSTT_+ *l'extension de* MSTT *ajoutant* $+\,(2)$ *et* $0\,(0)$ *au système de constructeurs, étendant les termes avec :*

$$\Lambda ::= \ldots \mid \aleph \mid \iota_1\Lambda \mid \iota_2\Lambda \mid \texttt{match}\ \Lambda\ \texttt{with}\ [\iota_1 V \rightsquigarrow \Lambda \mid \iota_2 V \rightsquigarrow \Lambda],$$

les règles de typage avec :

$$\frac{}{\Gamma, x : 0 \vdash \aleph : \sigma}\,(\text{abort}) \qquad \frac{\Gamma \vdash u : \sigma}{\Gamma \vdash \iota_1 u : \sigma + \tau}\,(\text{inj}_1) \qquad \frac{\Gamma \vdash u : \tau}{\Gamma \vdash \iota_2 u : \sigma + \tau}\,(\text{inj}_2)$$

$$\frac{\Gamma \vdash u : \sigma + \tau \qquad \Gamma, x : \sigma \vdash v : \xi \qquad \Gamma, y : \tau \vdash v' : \xi}{\Gamma \vdash \texttt{match}\ u\ \texttt{with}\ [\iota_1 x \rightsquigarrow v \mid \iota_2 y \rightsquigarrow v'] : \xi}\,(\text{match})$$

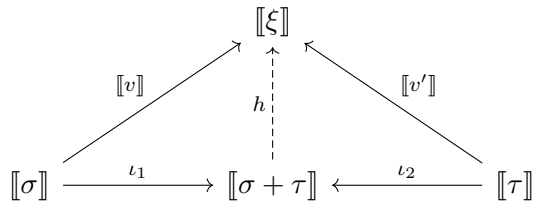
et les règles d'égalité avec :

$$\frac{\Gamma, x : 0 \vdash u : \sigma}{\Gamma, x : 0 \vdash u = \aleph : \sigma} \qquad \frac{\Gamma \vdash u : \sigma \qquad \Gamma, x : \sigma \vdash v : \xi \qquad \Gamma, y : \tau \vdash v' : \xi}{\Gamma \vdash \texttt{match}\ \iota_1 u\ \texttt{with}\ [\iota_1 x \rightsquigarrow v \mid \iota_2 y \rightsquigarrow v'] = v[u/x] : \xi}$$

$$\frac{\Gamma \vdash u' : \tau \qquad \Gamma, x : \sigma \vdash v : \xi \qquad \Gamma, y : \tau \vdash v' : \xi}{\Gamma \vdash \texttt{match}\ \iota_2 u'\ \texttt{with}\ [\iota_1 x \rightsquigarrow v \mid \iota_2 y \rightsquigarrow v'] = v'[u'/y] : \xi}$$

$$\frac{\Gamma \vdash u : \sigma + \tau \qquad \Gamma, z : \sigma + \tau \vdash v : \xi}{\Gamma \vdash \texttt{match}\ u\ \texttt{with}\ [\iota_1 x \rightsquigarrow v[(\iota_1 x)/z] \mid \iota_2 y \rightsquigarrow v[(\iota_2 y)/z]] = v[u/z] : \xi}$$

Alors, la reformulation MSTT $\to$ MSTT_+ *est admissible. D'un point de vue catégorique, on interprète le type* 0 *comme l'objet initial, tout type* $\sigma + \tau$ *comme un coproduit dont les injections* ι_1 *et* ι_2 *interprètent les termes correspondants, le terme* $\aleph : \sigma$ *comme l'unique morphisme* $0_{[\![\sigma]\!]} : 0 \to [\![\sigma]\!]$ *(ce qui est cohérent avec les règles ci-dessus car pour tout objet* A *du modèle, on a l'isomorphisme* $A \times 0 \cong 0$*), et le terme* $z : \sigma + \tau \vdash \texttt{match}\ z\ \texttt{with}\ [\iota_1 x \rightsquigarrow v \mid \iota_2 y \rightsquigarrow v'] : \xi$ *pour* $x : \sigma \vdash v : \xi$ *et* $y : \tau \vdash v' : \xi$ *comme le morphisme* $h : [\![\sigma + \tau]\!] \to [\![\xi]\!]$ *obtenu par propriété universelle du coproduit :*



Or, comme les topos sont finiment cocomplets et distributifs, tout modèle de MSTT_+ *est aussi un topos ; et on a de plus la propriété que tout modèle de* MSTT *est un modèle* MSTT_+. *De ce point de vue, on peut considérer cette extension comme « gratuite », car elle étend naturellement la théorie sans influencer les modèles possibles.*

Il est bien entendu possible de combiner les deux extensions présentées dans les exemples précédents, et il pourrait même être possible, avec un peu plus de travail, de montrer que ces extensions commutent. Nous n'irons cependant pas plus loin dans l'exploration des reformulations : dans le cadre de cette thèse, ces dernières nous serviront d'outils permettant de mesurer que nos modifications de MSTT sont toujours cohérentes avec l'état initial de ce dernier, et en particulier qu'il n'y a pas de perte en expressivité. Comme nous venons de le voir, la puissance expressive des topos autorise des modifications de la théorie des types qui entrent largement dans ce cadre : dans les sections suivantes, nous poursuivons donc notre exploration des propriétés exploitables des topos afin de poser des bases solides pour la modélisation des phénomènes sémantiques linguistiques.

5.4.2 La construction des ontologies

La discussion de la section 5.3 permet de souligner à quel point le type t des propositions est différent des autres types de base : les contraintes imposées par la logique interne de la théorie lui confèrent en effet des propriétés uniques, qui s'incarnent notamment dans son interprétation comme classificateur de sous-objets dans les modèles catégoriques. Le type e, en comparaison, ne présente aucune de ces contraintes, et son interprétation consiste en un simple objet sans la moindre hypothèse additionnelle. Cette observation invite ainsi à traiter t comme un type spécial, à l'instar des constructeurs 0 et 1 qui sont eux aussi d'arité nulle et qui jouissent de propriétés elles aussi très singulières. Pour appuyer cette perspective, nous apportons à nos théories de types, et en particulier MSTT, la modification suivante : le type t sera désormais traité comme un constructeur d'arité nulle.

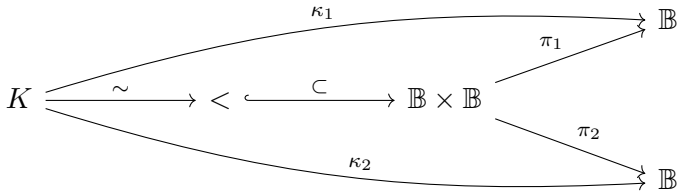
Que faire alors des types de base restants ? La finalité linguistique et sémantique que nous donnons à nos théories de types, dans la droite des lignées des considérations historiques (cf. chapitre 2), nous invitent à considérer ces types comme représentant les entités. Comme notre de point de départ est MSTT, nous partons d'un unique type e pour représenter ces entités. Les considérations sémantiques observées dans les premiers chapitres nous suggèrent d'élargir cette représentation sous la forme d'une classification des entités, que nous introduisons au travers de la notion d'ontologie de types telle que définie en section 1.4 : rappelons qu'il s'agit plus précisément d'un triplet $(\mathbb{B}, \leq, e)$ où $\mathbb{B}$ est un ensemble de types de base partiellement ordonné par $\leq$ et muni d'un plus grand élément e. Dès lors, nous renforçons nos hypothèses sur les théories de types en remplaçant l'ensemble des types de base par une ontologie de types, ce qui constitue la deuxième modification que nous apportons à nos théories.

Dans le cas de MSTT, ces deux modifications ne changent rien : il s'agit pour l'instant d'exactement la même théorie à ceci près que le système de types n'est plus construit à partir de l'ensemble de types de base $\{e, t\}$ et du système de constructeurs $\{\times(2), \to(2), 1(0)\}$, mais à partir de l'ontologie $(\{e\}, e, =)$ [87] et des constructeurs $\{\times(2), \to(2), 1(0), t(0)\}$. Comme le même système de types est produit à partir de ces deux points de vue, la théorie complète n'est donc absolument pas modifiée. Notre nouvel objectif sera dès lors de développer l'ontologie de types, et de déterminer comment ce développement peut être interprété dans les modèles catégoriques — et de vérifier notamment que ce développement est une reformulation admissible, en mettant l'accent sur la nécessité de pouvoir l'interpréter dans des topos.

Commençons par rappeler que la relation de sous-typage entre deux types a et b signifie que tout terme de type a est un argument acceptable pour toute fonction qui accepte des arguments

87. L'égalité utilisée pour cette ontologie est la relation définie sur $\{e\} \times \{e\}$ par $e = e$, elle n'a aucun lien avec la relation d'égalité sur les termes ni avec le prédicat d'égalité logique. À ce stade de notre développement, cette relation ontologique est totalement ignorée par la théorie MSTT.

de type b. Par conséquent, cette relation ne se manifeste qu'au moment de la composition et au regard des fonctions (ou des prédicats) présents dans le système. Tout comme Luo [107], et pour les raisons qu'il évoque, nous retenons une approche *coercive* du sous-typage, ce qui veut dire que toute intervention du sous-typage sur un terme de notre théorie sera marquée explicitement par l'application d'un terme spécifique appelé coercion. Jusqu'à nouvel ordre, nous ne retiendrons pas cependant l'approche définitionnelle de coercion comme abréviation caractéristique de son système de sous-typage coercitif, et noterons toujours la coercion lorsqu'elle a lieu. [88] Pour introduire formellement ces coercions explicites dans notre système, nous prenons comme point de départ une ontologie de types quelconque $(\mathbb{B}, e, \leq)$. Par définition, il est possible de voir la relation $\leq$ comme un sous-ensemble de $\mathbb{B} \times \mathbb{B}$; nous considérons de même la relation d'ordre stricte $<$ associée. On considère alors un ensemble K de symboles, dits *symboles de coercions de base*, isomorphe à $<$, et on définit deux fonctions $\kappa_1, \kappa_2 : K \to \mathbb{B}$ par les composées suivantes dans Set :



Afin de pouvoir faire intervenir ces symboles dans notre théorie de types, nous introduisons un nouveau jugement dans notre système, de la forme $\vdash k : \sigma < \tau$ où k est un symbole de coercion, et σ et τ sont des types dans $\mathbb{T}$. Étant donné que seuls les types de bases sont comparables pour l'instant, nous n'aurons pour commencer qu'une seule règle permettant d'introduire ces jugements :

$$\frac{c \in K}{\vdash c : \kappa_1(c) < \kappa_2(c)}\,(\text{base})$$

Pour une utilisation concrète dans notre théorie de types, nous étendons la définition des termes pour introduire les termes coercés :

$$\Lambda ::= \ldots \mid K\Lambda$$

Néanmoins, afin de respecter l'idée que le sous-typage ne doit intervenir qu'au moment de l'application d'une fonction, il ne sera pas possible de typer directement des termes coercés : il apparaîtront toujours dans le contexte d'une application. Nous reprenons pour assurer cela l'idée de la règle d'application coercive de Luo et ses collègues [107, 110] : [89]

$$\frac{\Gamma \vdash u : \sigma \to \tau \qquad \Gamma \vdash v : \sigma' \qquad \vdash c : \sigma' < \sigma}{\Gamma \vdash u(cv) : \tau}\,(\text{rac})$$

On remarquera que les coercions présentent beaucoup de similarités avec les fonctions, ce qui n'est pas étonnant au vu de notre compréhension des coercions comme des applications permettant de changer le type de leurs arguments. Néanmoins, nous tenons à garder les coercions comme

88. Nous reconnaissons cependant que l'approche de Luo sur cette question, en plus d'offrir un aspect pratique bien plus confortable, répond pleinement à l'idée intuitive que l'on peut se faire du sous-typage comme une coercion « naturelle », à l'opposé d'autres types de coercion linguistique. Ultimement, une fois notre compréhension générale des coercions bien installée, nous serons en mesure d'autoriser l'abréviation pour le cas particulier du sous-typage.

89. Il existe néanmoins une restriction à imposer sur cette règle pour éviter que les termes coercés soient accessibles directement comme des termes typés : la fonction considérée ne doit pas pouvoir se β-réduire à l'un de ces arguments. En effet, si f est une telle fonction qui se β-réduit à son i-ème argument, il serait possible de construire $fu_1 \ldots u_{i-1}(cv)u_{i+1} \ldots u_n$ qui se réduit en $cv : \sigma$, que nous souhaitons éviter. Dans le chapitre suivant, nous étudierons un système qui nous permettra d'écarter ce cas particulier.

séparées du reste des termes fonctionnels afin de pouvoir contrôler davantage les contextes dans lesquels elles peuvent être utilisées, tout en soulignant leurs propriétés particulières. Parmi ces dernières se retrouve notamment l'impératif de *cohérence*, une condition affirmant l'unicité des coercions : étant donnés deux types σ et τ, il n'existe au plus qu'une seule coercion $\sigma < \tau$. Cette propriété est pour l'instant vérifiée dans notre système puisque K est isomorphe à $<$, qui ne contient qu'un élément pour chaque paire de types strictement comparables. Mais en prévision des futures coercions qui enrichiront notre théorie, il nous paraît nécessaire de consolider ces impératifs avec des règles supplémentaires. On commencera notamment par remarquer que l'ensemble $<$ supporte une opération partielle de composition $\circ$ telle que $(b, c) \circ (a, b) = (a, c)$, par définition de la transitivité. Si nous choisissons l'isomorphisme $K \cong <$ de sorte à être également un homomorphisme qui reflète cette opération, alors K possède lui-même une composition, noté $\circ$ également. Introduisons alors un jugement d'égalité de coercions $\vdash k = k' : \sigma < \tau$; la composition et la cohérence sont supportées par les règles suivantes :

$$\frac{\vdash c : \sigma < \tau \quad \vdash c' : \tau < \xi}{\vdash c' \circ c : \sigma < \xi}\,(\text{comp}) \qquad \frac{\vdash c : \sigma < \tau \quad \vdash c' : \sigma < \tau}{\vdash c = c' : \sigma < \tau}\,(\text{coh})$$

En particulier, ces règles assurent que si $k, k', k'' \in K$ avec $k : a < b$, $k' : b < c$ et $k'' : a < c$, alors on peut dériver $k'' = k' \circ k$. Naturellement, les règles d'égalité sont complétées par la transitivité et la symétrie, faisant de la relation $=$ sur les coercions une relation d'équivalence. Enfin, même si nous voulons éviter de manipuler directement des termes coercés, il est possible de définir des conversions entre eux, qui peuvent ensuite se propager par passage au contexte. La conversion canonique est alors la suivante :

$$\frac{\Gamma \vdash v = v' : \sigma \quad \vdash c = c' : \sigma < \tau}{\Gamma \vdash cv = c'v' : \tau}$$

Si la similarité entre fonctions et coercions peut néanmoins nous servir, c'est pour déterminer en partie l'interprétation catégorique de ces dernières. En effet, l'interprétation catégorique doit assigner une interprétation directe à toutes les coercions afin de pouvoir interpréter les termes dans lesquels ces coercions apparaissent. Or, les termes de la forme $K\Lambda$ sont très similaires à des applications : nous pouvons nous servir de cette similarité pour interpréter les coercions comme des fonctions. Cependant, nous allons plutôt recourir à l'astuce introduite en section 5.2 et considérer pour toute coercion $c : a < b$ l'interprétation $[\![c]\!]$ comme un morphisme $A \to B$, de sorte que l'on ait $[\![cu]\!] = [\![c]\!] \circ [\![u]\!]$ pour tout terme u.

Cependant, les coercions de sous-typage présentent des propriétés particulières en comparaison des autres fonctions, du fait de leur fonction explicitée dans la section 3.3.2. Une propriété doit notamment nous apparaître ici : passer d'un type à un sous-type préserve le critère d'identité associé à ce type. Notre théorie de types contient justement le critère d'identité de chaque type σ sous la forme du prédicat $=_\sigma : \sigma \to \sigma \to t$. Cette idée de préservation peut facilement se traduire en règle logique, que nous donnons ci-dessous :

$$\frac{\vdash c : \sigma < \tau \quad \Gamma \mid \Theta \vdash cu =_\tau cu' : t}{\Gamma \mid \Theta \vdash u =_\sigma u' : t}$$

Cette règle peut être appliquée de telle sorte à montrer la validité du jugement logique $\Gamma \mid cu =_\tau cu' \vdash u =_\sigma u'$ dans tout modèle, pour tout termes u et u'. Traduite en notions catégoriques, cette proposition signifie alors que l'égalité $[\![c]\!] \circ [\![u]\!] = [\![c]\!] \circ [\![u']\!]$ implique l'égalité $[\![u]\!] = [\![u']\!]$. Cette propriété est bien entendue restreinte aux interprétations de la théorie initiale, mais en la

généralisant à tous morphismes, dans l'esprit de notre analyse du type t comme classificateur de sous-objet, nous pouvons imposer une propriété encore plus forte : l'interprétation d'une coercion de sous-typage sur les types de base serait alors un monomorphisme. [90]

À partir de ce point, nous allons donc raffiner l'ensemble de nos modèles possibles pour ne conserver que ceux dans lesquels les coercions sont bien interprétées comme des monomorphismes. On notera alors que la nouvelle théorie ainsi obtenue est a priori une reformulation admissible, voire une extension conservatrice. [91] Il est même possible de renforcer les hypothèses catégoriques en faisant le lien entre l'ontologie de types et les modèles : en effet, en tant qu'ensemble partiellement ordonné l'ontologie de types peut être exprimée comme une petite catégorie $\mathcal{B}$ dont les objets sont donnés par $\mathsf{obj}(\mathcal{B}) = \mathbb{B}$ et dont les morphismes sont donnés par les éléments de K étendus de symboles pour l'identité ; pour tous objets a et b, on a ainsi :

$$\mathcal{B}(a,b) = \begin{cases} \{k\} & \text{si } a < b, \text{ avec } k \in K \text{ tel que } \kappa_1(k) = a \text{ et } \kappa_2(k) = b, \\ \{\mathrm{id}_a\} & \text{si } a = b, \\ \varnothing & \text{sinon.} \end{cases}$$

La composition sur $\mathcal{B}$ étend alors la composition sur K comme attendu. Il est alors clair que tout morphisme de $\mathcal{B}$ est un monomorphisme, puisque les *hom-sets* contiennent au plus un élément. Afin de cadrer au mieux notre interprétation, nous exigeons désormais que tout modèle $\mathcal{C}$ de notre théorie de types admette l'existence d'un plongement $O_{\mathcal{C}} : \mathcal{B} \to \mathcal{C}$ qui préserve les monomorphismes. [92]

Cette reconnaissance du lien entre sous-typage et monomorphisme — du moins, pour l'instant, au niveau des types de base — ouvre l'accès à de puissantes propriétés dans les modèles par l'interaction de ces relations avec le classificateur de sous-objet. Afin d'examiner de plus près ces propriétés et de voir comment elles peuvent influencer les règles de notre théorie, nous nous plaçons jusqu'à la fin de ce chapitre dans un modèle $\mathcal{C}$ quelconque de notre théorie actuelle : il s'agit naturellement d'un topos dont on notera Ω le classificateur de sous-objet, et pour lequel on supposera l'existence d'un plongement comme défini ci-dessus. Par abus de langage, on appellera alors *prédicat* tout morphisme de $\mathcal{C}$ de codomaine Ω. Un premier résultat qu'il est possible de remarquer est alors l'existence, pour tout type a de l'ontologie, d'un prédicat toujours vrai sur ce type : si A est son interprétation, posons $\mathrm{true}_A = \top \circ !_A : A \to \Omega$. Ce prédicat joue un rôle essentiel dans les propriétés du classificateur de sous-objet, puisqu'il apparaît en bas des produits fibrés caractéristiques de sa propriété centrale, et trouve déjà une correspondance dans la théorie de types sous la forme du terme $\lambda x.\top : a \to t$. Ce prédicat particulier se retrouve également dans UTT sous le nom de *forme prédicationnelle* du type a [34], notamment utilisé pour la modélisation de la négation.

De manière plus générale, une autre famille de prédicats va retenir notre attention : les prédicats caractéristiques du sous-type d'un type ontologique donnée. En effet, si $a < b$ dans notre ontologie, alors il existe un monomophisme $c : A \rightarrowtail B$ dans $\mathcal{C}$ qui engendre un sous-objet de B. Par la propriété du classificateur, il existe alors un prédicat $\chi_A^B : B \to \Omega$ qui n'est satisfait

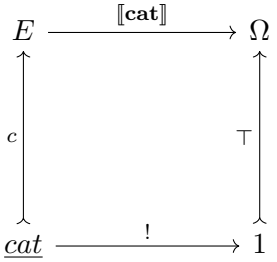
90. Une question plus difficile, que nous ne prendrons pas le temps d'aborder dans cette thèse mais qui mérite d'être soulevée, serait de vérifier s'il existe bien des modèles satisfaisant toutes les règles précédentes et dans lesquelles l'interprétation des coercions sur les types de base ne sont *pas* des monomorphismes. Nous supposons ici, sans perte de généralité, que c'est le cas.

91. Ceci a été démontré dans le cadre du sous-typage coercif pour UTT par [110].

92. En particulier, il existe un plongement $O_{\mathcal{C}\ell}$ de $\mathbb{B}$ dans la catégorie classificatrice, et il est attendu que pour un foncteur d'interprétation $[\![.]\!] : \mathcal{C}\ell \to \mathcal{C}$, on ait $O_{\mathcal{C}} = [\![.]\!] \circ O_{\mathcal{C}\ell}$. Rappelons que par convention, pour un modèle $\mathcal{C}$ fixé et pour tout type de base a, on notera A pour signifier $O_{\mathcal{C}}(a)$. De même, nous introduisons désormais la simplification de notation $c := O(c)$ pour tout $c \in K$, de sorte qu'on notera directement c plutôt que $[\![c]\!]$.

que sur les termes de type a, à travers l'égalité $\chi^B_A \circ c = \text{true}_A$. L'existence d'un tel prédicat dans les modèles peut avoir une conséquence importante dans les théories de types : elle autorise en effet une notion de typage interne, à travers un prédicat $P_a : e \to t$ tel que $\Gamma \mid \varnothing \vdash P_a(u) : t$ est valide si et seulement s'il existe un terme v tel que $\Gamma \vdash v : a$ et $\Gamma \vdash u = cv : e$ sont dérivables. Ce lien entre types ontologiques et prédicats est notamment au cœur d'un débat autour de l'interprétation des noms communs : là où la tradition montagovienne les traite comme des prédicats, Luo et Chatzikyriakidis [108, 34] proposent d'associer à chaque nom commun son propre type de base, une tentative de médiation ayant été suggérée par Retoré [149] et exploitant justement l'équivalence ci-dessus. Comme développé dans [9], nous pensons surtout que cette équivalence permet de tisser un lien fort entre les deux approches, mais que les noms communs ne peuvent pas suffire par eux-mêmes à définir l'ontologie de types. Il est néanmoins important de noter que toute tentative d'exploiter cette propriété devra être soigneusement étudiée afin de prévenir l'indécidabilité du typage dans la théorie des types, comme souligné dans [34].

Dans le paragraphe précédent, la propriété du classificateur de sous-objet a été exploitée dans le sens sous-objets vers prédicats, mais il est également possible de considérer l'inverse : alors, cette propriété peut permettre de générer des types admissibles dans notre théorie à partir des prédicats que l'on suppose. Prenons le cas concret du mot *chat* et de son interprétation comme la constante **cat** : $e \to t$. Par définition, il existe alors dans $\mathcal{C}$ un objet $\underline{cat}$ tel que le carré ci-dessus soit un produit fibré :



Ce nouvel objet induit alors un nouveau type `cat` qu'il serait envisageable d'ajouter à notre ontologie s'il s'avère pertinent. Ce type, comme on peut s'y attendre, regroupe exactement les entités qui sont des chats, et est par construction un sous-type de e. Ceci démontre notamment l'adéquation entre notre modélisation du sous-typage et l'intention linguistique que l'on place dans cette relation. Cette adéquation est d'autant plus frappante lorsque l'on interprète à travers notre modèle des énoncés analytiques, par exemple « tous les chats sont des animaux ». En effet, l'interprétation traditionnelle d'une telle phrase est la formule logique $\forall x.\mathbf{cat}(x) \Rightarrow \mathbf{animal}(x)$, où **animal** est aussi un prédicat $e \to t$. Introduire cette formule comme axiome dans la logique de notre théorie a pour conséquence la validité du jugement $x : e \mid \mathbf{cat}(x) \vdash \mathbf{animal}(x) : t$. Or, chacun de ces prédicats se convertit en un sous-objet de E, représentant les types `cat` et `animal` respectivement : alors, la validité de ce jugement se traduit par l'inégalité correspondante dans $\text{Sub}(E)$; autrement dit, le modèle soutient la relation de sous-typage `cat` $\leq$ `animal`, et ce uniquement à partir de l'axiome initial.

Ainsi, les propriétés des topos nous permettent de définir une ontologie de type suivant un certain nombre de schéma analytiques si on le souhaite, offrant par là même un grand nombre de possibilités pour construire des ontologies cohérentes. Cependant, cette construction ne répond pas tout à fait aux critères de construction que nous avons retenus de nos discussions linguistiques et philosophiques, en particulier au regard des théories de Sommers. Dans la dernière sous-section pour ce chapitre, nous allons donc regarder de plus près les conditions imposées par cette théorie sur nos ontologies, et étudier les interactions de ces conditions avec notre théorie et nos modèles.

5.4.3 Ontologies sommersiennes et logique intuitionniste

Si l'on suit attentivement la théorie de Sommers telle que décrite aux sections 2.5 et 3.3, notre ontologie de types pour la linguistique est contrainte par certaines propriétés structurelles qui lui imposent la forme d'un arbre. Rappelons que tout ensemble muni d'une relation binaire $(X, \prec)$ peut être vu comme un graphe orienté dont les sommets sont les éléments de X et pour lequel il existe une arête (x, y) si et seulement si $x \prec y$. Si $\leq$ est une relation d'ordre sur un ensemble X, on définit sa *réduction réflexive transitive* comme la relation $\lhd$ définie pour $x, y \in X$ par $x \lhd y$ si et seulement si y est un élément minimal de l'ensemble $\{z \in X \mid x \leq z \text{ et } x \neq z\}$. Nous utilisons cette construction pour définir formellement des ontologies qui respectent la forme d'arbre :

Définition 40. *Une ontologie de types* $(\mathbb{B}, e, \leq)$ *est dite* sommersienne *si le graphe engendré par la réduction réflexive transitive de* $\leq$ *sur* $\mathbb{B}$ *(c.-à-d. son diagramme de Hasse) est un arbre.*

Naturellement, lorsqu'une ontologie est sommersienne, l'arbre associé a pour racine le type e. La forme d'arbre ainsi obtenue pour la hiérarchie des types ontologiques a alors l'avantage d'être simple à visualiser, en plus de resepcter les contraintes linguistiques observées par Sommers. La vision des types ontologiques est d'ailleurs d'autant plus aisée que cette hiérarchie suppose une certaine exhaustivité : en effet, nous savons que par construction (cf. sous-section 3.3.2) les types qui caractérisent les mots occupant les positions syntaxiques d'arguments constituent les feuilles de l'arbre hiérarchique. Cela implique alors que pour chaque type a de l'ontologie, l'ensemble des mots-arguments régis par le type a est exactement l'union des mots correspondant aux feuilles descendant de a.

Une question légitime est alors de déterminer comment cette hiérarchie se reflète dans nos modèles catégoriques : en effet, les modèles ont aussi leur propres contraintes, qui pourraient ne pas nécessairement s'accorder avec celles de l'ontologie, et il convient de vérifier que si incompatibilité il y a entre lesdites contraintes, les conséquences de cette incompatibilité ne remettent pas en cause la précision de notre modélisation. Pour ce faire, nous allons dans la suite nous intéresser à la manière de prolonger une ontologie sommersienne en algèbre de Heyting, puisque qu'une telle algèbre sera alors nécessairement contenue dans l'algèbre $\mathrm{Sub}(E)$, quel que soit le modèle catégorique choisi — ce qui permettra en particulier d'étudier le plongement associé. Nous nous appuierons pour cette construction sur des éléments basiques de la théorie des ordres, comme introduite p. ex. dans [47].

Soit $(\mathbb{B}, e, \leq)$ notre ontologie de départ, qu'on supposera sommersienne. La première étape de cette construction consiste à compléter cette ontologie par un opérateur d'union sur les types permettant d'accéder à l'intégralité des bornes supérieures possibles. Soit $\cup$ un symbole d'union d'arité 2, posons alors $\mathfrak{u}(\mathbb{B})$ l'ensemble librement engendré par l'ajout de cet opérateur à $\mathbb{B}$, c.-à-d. l'ensemble défini par la grammaire :

$$\mathfrak{u}(\mathbb{B}) ::= \mathbb{B} \mid \mathfrak{u}(\mathbb{B}) \cup \mathfrak{u}(\mathbb{B})$$

Posons alors $\preceq$ la plus petite relation d'ordre sur $\mathfrak{u}(\mathbb{B})$ respectant les axiomes suivants :

$$\begin{aligned}
&(0) \quad \forall a, b \in \mathbb{B}.\ a \leq b \Rightarrow a \preceq b \\
&(i) \quad \forall a, b \in \mathfrak{u}(\mathbb{B}).\ a \preceq a \cup b \\
&(i') \quad \forall a, b \in \mathfrak{u}(\mathbb{B}).\ b \preceq a \cup b \\
&(ii) \quad \forall a, b, c \in \mathfrak{u}(\mathbb{B}).\ (a \preceq c) \wedge (b \preceq c) \Rightarrow a \cup b \preceq c
\end{aligned}$$

Cette relation d'ordre impose naturellement un certain nombre d'égalités entre éléments de $\mathfrak{u}(\mathbb{B})$, qui lui donnent une structure théoriquement plus simple de sup-demi-treillis, c.-à-d. un ensemble ordonné où toute paire d'élément admet une borne supérieure.

Lemme 3. $(\mathfrak{u}(\mathbb{B}), \preceq, e, \cup)$ *est un sup-demi-treillis de plus grand élément* e*.*

Démonstration. Nous ne donnons ici que les grandes lignes de la preuve afin de ne pas alourdir notre propos. Être un sup-demi-treillis suppose un certain nombre d'hypothèses, à commencer par montrer que l'opérateur $\cup$ est associatif, commutatif et idempotent : toutes ces propriétés sont conséquences des trois derniers axiomes. De même, on montre facilement que pour tous éléments $a, b \in \mathfrak{u}(\mathbb{B})$, on a $a \preceq b$ si et seulement si $a \cup b = b$, ce qui suffit à caractériser $\cup$ comme opérateur de borne supérieure. Enfin, en remarquant que tout élément de $\mathfrak{u}(\mathbb{B})$ peut s'écrire $\bigcup_{i=1}^{n} a_i$ avec $a_i \in \mathbb{B}$ et en exploitant le fait que e est le plus grand élément de $\mathbb{B}$ ainsi que l'inclusion de $\leq$ dans $\preceq$ d'après le premier axiome, on en conclut que e est bien le plus grand élément de notre nouvel ensemble. □

La seconde étape est de construire une fermeture de $\mathfrak{u}(\mathbb{B})$ vers le bas. Pour cela, nous introduisons un symbole d'intersection $\cap$ d'arité 2 ainsi qu'un nouvel objet 0 destiné à représenter le plus petit élément de notre future algèbre. Posons alors $\mathfrak{h}(\mathbb{B})$ l'ensemble librement engendré par l'ajout ce symbole et de cet élément à l'ensemble $\mathfrak{u}(\mathbb{B})$. Soit alors $\leq'$ la plus petite relation d'ordre sur $\mathfrak{h}(\mathbb{B})$ qui contient $\preceq$ et qui respecte les propriétés imposées par les axiomes suivants :

$$
\begin{aligned}
&(iii) \quad \forall a \in \mathfrak{h}(\mathbb{B}).\ 0 \leq' a \\
&(iv) \quad \forall a, b \in \mathbb{B}.\ a \cap b = \begin{cases} a & \text{si } a \leq b \\ b & \text{si } b \leq a \\ 0 & \text{sinon} \end{cases} \\
&(v) \quad \forall a, b, c \in \mathfrak{h}(\mathbb{B}).\ (a \cup b) \cap c = (a \cap c) \cup (b \cap c) \\
&(v') \quad \forall a, b, c \in \mathfrak{h}(\mathbb{B}).\ c \cap (a \cup b) = (c \cap a) \cup (c \cap b)
\end{aligned}
$$

Lemme 4. $(\mathfrak{h}(\mathbb{B}), \leq', e, 0, \cup, \cap)$ *est un treillis borné, complet et distributif.*

Démonstration. Nous esquissons là encore les grandes lignes de la preuve. Les axiomes (iv), (v) et (v') permettent de prouver par induction que $\cap$ est bien associatif, commutatif et idempotent. On prouve de même que $a \leq' b$ si seulement si $a \cap b = a$ pour tous $a, b \in \mathfrak{h}(\mathbb{B})$, ce qui suffit à montrer la structure de treillis. L'axiome (iii) définit clairement 0 comme plus petit élément de $\mathfrak{h}(\mathbb{B})$ et le statut de e reste inchangé, d'où la propriété bornée. La complétude s'obtient directement par construction en remarquant que l'ensemble $\mathbb{B}$ initial est fini par hypothèse, ce qui rend $\mathfrak{u}(\mathbb{B})$ et $\mathfrak{h}(\mathbb{B})$ également finis : les bornes supérieure et inférieure de chaque partie de ce dernier ensemble sont donc aisées à construire. Enfin, la distributivité est une conséquence des axiomes (v) et (v') et des propriétés de $\cap$ comme borne inférieure. □

On relèvera notamment que par construction, tous les types ontologiques initialement incomparables sont considérés comme disjoints dans ce treillis, propriété qui reflète bien le comportement attendu pour ces types. On remarquera également que cette propriété implique que tout élément de $\mathfrak{h}(\mathbb{B})$ de la forme $a \cap b$ est égal à un élément de $\mathfrak{u}(\mathbb{B}) \cup \{0\}$: l'ajout de l'opérateur d'intersection n'introduit donc pas de nouvel élément à notre ensemble, à l'exception de 0 lui-même.[93] Nous ne sommes dès lors plus qu'à une étape d'avoir étendu notre ontologie

93. Cette affirmation souligne qu'au cours de notre développement, nous avons implicitement quotienté nos différents ensembles par la relation d'égalité générée une fois la relation ordre correspondante clairement définie.

initiale en algèbre de Heyting. Le résultat suivant est une conséquence directe de la propriété de complétude de notre treillis :

Lemme 5. *Pour tous éléments $a, b \in \mathfrak{h}(\mathbb{B})$, l'ensemble $\{x \in \mathfrak{h}(\mathbb{B}) \mid a \cap x \leq' b\}$ admet une borne supérieure.*

Ce dernier lemme signifie notamment qu'il est possible de définir une opération adjointe à l'intersection sous la forme d'un symbole $\rightrightarrows$ d'arité 2 défini par l'axiome suivant :

$$(vi) \quad \forall a, b \in \mathfrak{h}(\mathbb{B}).\ a \rightrightarrows b = \bigcup \{x \in \mathfrak{h}(\mathbb{B}) \mid a \cap x \leq' b\}$$

où $\bigcup$ dénote la borne supérieure — comme union de tous les éléments de l'ensemble. Par passage aux classes d'équivalence pour l'égalité, l'enrichissement de $\mathfrak{h}(\mathbb{B})$ par ce symbole n'introduit pas de nouvel élément dans le treillis. Il résulte alors du lemme ci-dessus que la structure formée par $(\mathfrak{h}(\mathbb{B}), \leq', e, 0, \cup, \cap, \rightrightarrows)$ est bien une algèbre de Heyting. Ceci nous permet au passage d'introduire une notation de pseudo-complémentation sur $\mathfrak{h}(\mathbb{B})$, en posant $\neg a = a \rightrightarrows 0$ pour tout a.

Rappelons cependant que notre vision de l'ontologie sommersienne impose une certaine forme d'exhaustivité, où chaque type est perçu comme étant exactement l'union de ses types enfants. Comment imposer formellement cette vision à notre algèbre, et quelles conséquences peut-elle avoir ? Étant donné un type ontologique $a \in \mathbb{B}$ de notre ontologie sommersienne, notons $C(a) \subset \mathbb{B}$ l'ensemble de ses enfants et $S(a) \subset \mathbb{B}$ l'ensemble de ses frères dans l'arbre associé. Par ailleurs, si $a \neq e$, notons $p(a)$ le parent direct de a dans cet arbre.[94] Alors, formaliser la vision linguistique décrite ci-dessus pour nos types ontologiques peut s'incarner dans un axiome que nous appelerons *axiome d'exhaustivité*, que nous décrivons a priori dans $\mathfrak{u}(\mathbb{B})$ mais qui s'étend à $\mathfrak{h}(\mathbb{B})$ par isomorphisme de leurs quotients par l'égalité, et qui s'incarne dans la formule suivante :

$$\forall a \in \mathbb{B}.\ a = \bigcup C(a)$$

La propagation de cet axiome à $\mathfrak{h}(\mathbb{B})$ et à l'algèbre de Heyting qui s'articule autour a des conséquences importantes, faisant notamment apparaître une propriété qui la distingue fortement des autres algèbres de ce genre.

Théorème 2. *Si $\mathfrak{h}(\mathbb{B})$ satisfait l'axiome d'exhaustivité, alors $(\mathfrak{h}(\mathbb{B}), \leq', e, 0, \cup, \cap, \neg)$ est une algèbre booléenne.*

Démonstration. Il suffit de montrer que pour tout $a \in \mathfrak{h}(\mathbb{B})$, on a l'égalité $a \cup \neg a = e$. Montrons dans un premier temps que cela est vrai dans le cas particulier où $a \in \mathbb{B}$. Posons l'ensemble $F(a) = \{x \in \mathbb{B} \mid a \leq x\}$. Comme notre ontologie est un arbre, $F(a)$ correspond exactement à la branche formant le chemin entre a et e. Posons ensuite $N(a) = \bigcup_{x \in F(a)} S(x)$, puis $b = \bigcup N(a)$ dans $\mathfrak{h}(\mathbb{B})$. Par construction, tout élément $y \in N(a)$ vérifie $a \cap y = 0$ puisque les frères des ancêtres de a ne sont pas comparables à a. Par distributivité, on vérifie alors $a \cap b = 0$, ce qui implique $b \leq' \neg a$. Or, par commutativité et associativité de $\cup$ et par application de l'axiome d'exhaustivité, on vérifie la récurrence suivante :

$$\begin{aligned} a \cup b &= (a \cup \bigcup S(a)) \cup \bigcup S(p(a)) \cup \cdots \\ &= p(a) \cup \bigcup S(p(a)) \cup \cdots \\ &= p\big(p(a)\big) \cup \cdots \\ &= e \end{aligned}$$

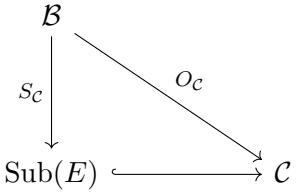
94. Relevons que ces trois définitions sont liées par la relation $C\big(p(a)\big) = \{a\} \cup S(a)$.

On en déduit alors, par les propriétés générales de $\cup$ (notamment en application de l'axiome (ii) ci-dessus), l'égalité $a \cup \neg a = e$. Soit maintenant $a \in \mathfrak{h}(\mathbb{B})$ quelconque. Par construction de notre ensemble, il existe nécessairement une famille $b_1, \ldots, b_n$ d'éléments de $\mathbb{B}$ telle que $a = \bigcup_{i=1}^{n} b_i$. Alors, par propriété des algèbres de Heyting (cf. p. ex. [75, §8]), $\neg a = \bigcap_{i=1}^{n} \neg b_i$. Pour tout $i \in [\![1, n]\!]$, posons alors $c_i = \bigcup N(b_i)$ comme précédemment : on a alors $\bigcap_{i=1}^{n} c_i \leq' \neg a$, et par distributivité, on a :

$$a \cup \bigcap_{i=1}^{n} c_i = \bigcup_{i=1}^{n} b_i \cup \bigcap_{i=1}^{n} c_i = \bigcap_{i=1}^{n} \left(\bigcup_{j=1}^{n} b_j \cup c_i \right) = \bigcap_{i=1}^{n} \left(\bigcup_{\substack{j=1 \\ j \neq i}}^{n} b_j \cup \underbrace{(b_i \cup c_i)}_{=e} \right) = e$$

On en conclut alors $a \cup \neg a = e$ pour tout $a \in \mathfrak{h}(\mathbb{B})$. □

Venons-en à présent à la manière dont les modèles catégoriques reflètent cette structure ontologique étendue. À la fin de la sous-section précédente, nous avons introduit un plongement $O_{\mathcal{C}} : \mathcal{B} \to \mathcal{C}$ de l'ontologie des types vers un modèle $\mathcal{C}$ quelconque. Comme l'ontologie en question est un ensemble partiellement ordonné de plus grand élément e et ne crée que des monomorphismes, ce plongement se factorise en fait à travers $\mathrm{Sub}(E)$, qui par propriété générale des topos est lui-même une algèbre de Heyting. En effet, l'algèbre $\mathrm{Sub}(E)$ peut être vue comme une sous-catégorie de $\mathcal{C}$ où chaque sous-objet est envoyé sur un objet de $\mathcal{C}$ et chaque morphisme représentant l'ordre sur $\mathrm{Sub}(E)$ est envoyé vers un représentant de la classe d'équivalence des monomorphismes entre les objets correspondants. En voyant cette algèbre comme une catégorie (dans le même esprit que la catégorie $\mathcal{B}$), la factorisation en question se décrira en posant $S_{\mathcal{C}}$ la restriction de $O_{\mathcal{C}}$ à ce codomaine, de telle sorte que les foncteurs du diagramme suivant commutent dans Cat :



En tant qu'algèbre de Heyting, $\mathrm{Sub}(E)$ contient toutes les unions et intersections des objets représentant les types ontologiques. Par ailleurs, $\mathcal{C}$ contient tous les prédicats du langage sous forme de morphismes entre les types ontologiques et le classificateur de sous-objet Ω. Comme vu précédemment, cela signifie que $\mathrm{Sub}(E)$ contient autant de sous-objets pour un objet $A \leq E$ qu'il n'y a de prédicats du type $a \to t$ avec $a \in \mathbb{B}$ dans la théorie des types initiale, rendant la structure de $\mathrm{Sub}(E)$ bien plus complexe que celle de $\mathfrak{h}(\mathbb{B})$.

Une des conséquences de cette complexification est la potentielle incapacité de $\mathrm{Sub}(E)$ à refléter le comportement de l'algèbre $\mathfrak{h}(\mathbb{B})$. En effet, nous avons vu plus haut que dans cette dernière deux types ontologiques incompatibles $a, b \in \mathbb{B}$ satisfont par construction $a \cap b = 0$. Cette propriété se propage naturellement à la catégorie classificatrice de notre théorie de types, en montrant notamment qu'il n'existe pas de terme u pouvant satisfaire $\Gamma \vdash u : a$ et $\Gamma \vdash u : b$, à moins que $u = \aleph$.[95] Mais dans le cas plus général, il n'y a aucune raison que l'isomorphisme

95. La preuve de ce fait repose sur l'idée que le système de types considéré ici ne contient pas d'éléments capables de générer un terme complexe ayant un type dans $\mathbb{B}$, à part si l'on étend notre typage aux termes coercés. Mais même dans ce cas, cela signifie qu'un terme satisfaisant les deux jugements ci-dessus est soit une variable, soit une constante, soit un terme coercé, soit $\aleph$. Les deux premiers cas sont exclus puisque chaque constante a un type unique et que si c'est une variable, elle ne peut recevoir qu'un seul type dans le contexte Γ commun aux deux

$A \cap B \cong 0$ soit valide dans un modèle catégorique quelconque : il n'y aura juste aucun antécédent de cet objet dans la catégorie classificatrice.

Ce phénomène n'a a priori aucune incidence sur l'interprétation de notre théorie des types, puisque cet objet n'est pas accessible. S'il pose un problème, celui-ci serait davantage d'ordre philosophique. Que représentent les objets de nos modèles catégoriques ? Une réponse intuitive se base sur la catégorie Set comme l'un des modèles possibles : dans une interprétation ensembliste, les éléments de l'ensemble correspondant à un type ontologique donné peuvent être vus comme représentant des entités du monde réel, sur lesquels on pourra affirmer ou nier des propriétés (ces propriétés étant également regroupées en ensembles, soit dit en passant). Mais, comme nous avons pu le voir dans le chapitre 3, cette interprétation dépasse le cadre du réel, puisqu'il est possible d'affubler des entités de caractéristiques imaginaires, par exemple pour parler des licornes. Une idée persiste néanmoins, qui est la représentation des interprétations des types ontologiques comme des viviers d'unités conceptuelles qui peuvent servir de référents dans un monde choisi. Par hypothèse linguistique, suivant la théorie de Sommers, nous reconnaissons ces viviers comme soit inclus l'un dans l'autre, signifiant que l'un offre une perspective plus large qu'un autre sur des objets similaires ; soit disjoints, dans le sens où si A et B sont disjoints, si x provient de l'objet A, et si y provient de l'objet B, alors il n'existe aucun monde dans lequel x et y peuvent désigner la même entité. [96]

Notons cependant que ces viviers d'entités n'ont aucune raison d'être des ensembles : il peut s'agir plus généralement de classes, sans obligation d'avoir la moindre propriété prédéfinie ni même d'être mathématiquement caractérisable. Au contraire, cette vision paraît plus à même de rendre compte de nombreux problèmes inhérents à la langue elle-même, par exemple en ce qui concerne les descriptions sans référents [54], les énoncés vagues [25], voire la distinction entre entités dénombrables et masses [37, 59]. Or, l'une des vertus de la théorie générale des topos est justement de proposer un univers de systèmes proches des ensembles, sans être complètement des ensembles non plus. Il n'y a donc aucune contradiction à considérer nos types comme des classes abstraites d'unités sémantiques permettant de générer des référents dans des mondes possiblement hypothétiques. Il résulte de cette interprétation que tout objet du modèle représente une classe d'entités présentant des propriétés similaires et pouvant être référées dans un monde donné. Ceci nous amène à entrevoir pourquoi accepter la propriété $A \cap B \ncong 0$ pour des types a et b initialement disjoints peut poser un problème philosophique : cela reviendrait en effet à accepter l'existence, dans un monde possible, d'un référent pouvant répondre aux caractéristiques des deux types à la fois.

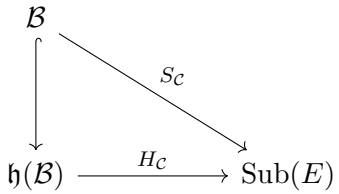
Pourtant, nous savons depuis la section 1.2 qu'un certaine classe de référents satisfaisant des propriétés proches existe : il s'agit des objets pointés qui, comme discuté en sous-section 3.3.3, n'appartiennent pas à l'ontologie. L'objet $A \cap B$ est-il alors un bon candidat pour interpréter les types pointés ? Suivant l'analyse d'Asher [4, §5.2.1], la réponse est en fait négative, pour la raison que $A \cap B$ est essentiellement inclus dans A et B, ce qui signifie qu'une entité $x \in A \cap B$ définit deux entités $y \in A$ et $y' \in B$ pour lesquelles il existe un seul même référent dans un monde possible, ce qui contredit l'hypothèse énoncée plus haut sur les types disjoints. En d'autres termes,

jugements. Quant aux termes coercés, ils se ramènent à l'un des trois autres cas en remontant les coercions, pour les mêmes raisons que plus haut et sachant que si notre ontologie est sommersienne, alors elle ne contient pas de sous-type commun à a et b. Le terme spécial $\aleph$ reste donc notre unique possibilité, et implique au passage que Γ contient une variable $z : 0$.

96. Il y a dans cette vision une légère dérivation entre les mots et la distinction entre arguments et prédicats : les entités sont globalement issues des noms, mais dans notre approche, les noms génèrent également des prédicats. Cette dualité, qui est au cœur du débat entre types et prédicats pour les noms communs, n'est cependant pas paradoxale : la valeur de prédication d'un nom vient en effet de l'assimilation entre le nom lui-même et sa composition avec le verbe *être*. C'est principalement sur cette assimilation que Sommers construit son ontologie.

non seulement on ne peut pas utiliser ce type pour modéliser les objets pointés, mais en plus on ne peut pas supposer que ce type est utilisable tout court, sous peine d'introduire de l'incohérence dans notre interprétation sémantique. Dans notre optique de renverser la dynamique entre modèles et théorie, permettant aux premiers d'influencer la structure de la seconde, l'existence d'un type intersection non-zéro laisse la possibilité d'étendre notre théorie pour utiliser ce type, ce qui irait à l'encontre de nos impératifs linguistiques.

Mais au-delà de cet impératif, il n'est théoriquement pas nécessaire d'imposer cette propriété comme une contrainte sur nos modèles. On se contentera plus justement de parler de *préférence*, sans exclure la possibilité que l'on puisse choisir des modèles qui ne la satisfont pas. Néanmoins, il pourra être utile de caractériser formellement cette préférence pour future référence, ce que nous nous proposons de faire dans la définition 41 ci-dessous. En préambule, nous posons $\mathfrak{h}(\mathcal{B})$ la catégorie correspondant aux classes d'équivalence par l'égalité de $\mathfrak{h}(\mathbb{B})$ en tant qu'ensemble partiellement ordonné, de la même manière que nous avons construit $\mathcal{B}$ plus haut. Alors, comme $\mathcal{B}$ est évidemment une sous-catégorie de cette nouvelle structure, le plongement $S_{\mathcal{C}} : \mathcal{B} \to \mathrm{Sub}(E)$ se prolonge naturellement en un nouveau plongement $H_{\mathcal{C}} : \mathfrak{h}(\mathcal{B}) \to \mathrm{Sub}(E)$ tel que $H_{\mathcal{C}}(a \cup b) = H_{\mathcal{C}}(a) \cup H_{\mathcal{C}}(b)$ et $H_{\mathcal{C}}(0) = 0$, et bien sûr tel que le diagramme suivant commute :



Définition 41. *On dira qu'un modèle* $\mathcal{C}$ reflète l'ontologie *si le plongement* $H_{\mathcal{C}} : \mathfrak{h}(\mathcal{B}) \to \mathrm{Sub}(E)$ *est un homomorphisme d'algèbres de Heyting.*

La clé de cette définition est qu'elle impose notamment la condition $H_{\mathcal{C}}(a \cap b) = H_{\mathcal{C}}(a) \cap H_{\mathcal{C}}(b)$ pour tous types $a, b \in \mathbb{B}$: en particulier, si a et b sont des types incomparables, cette condition mène donc à $H_{\mathcal{C}}(a) \cap H_{\mathcal{C}}(b) \cong 0$ dans $\mathrm{Sub}(E)$, qui est bien la caractéristique recherchée. Mais cette propriété de réflection de l'ontologie a une autre vertu : elle permet également de refléter l'axiome d'exhaustivité, faisant le cas échéant de $\mathrm{Sub}(E)$ une algèbre booléenne. Cette propriété ouvre alors une autre question : si l'on impose l'axiome d'exhaustivité et que notre modèle reflète l'ontologie, n'obtient-on pas un modèle trop restreint ? Une des premières craintes serait bien sûr de se retrouver borné à une interprétation en logique classique et de perdre la puissance de la logique intuitionniste. Mais cette crainte peut être rapidement écartée : en effet, la logique interne d'un topos est classique si et seulement si le topos est lui-même classique, c.-à-d. lorsque le classificateur de sous-objet est assimilable à un ensemble à deux éléments par un isomorphisme $1 + 1 \cong \Omega$, propriété qui, d'après p. ex. Goldblatt [75, §7.3], est équivalente à ce que $\mathrm{Sub}(\Omega)$ soit une algèbre booléenne. À l'inverse, pour que la logique interne du topos soit purement intuitionniste, deux conditions sont nécessaires : le topos doit être non bivalent, et $\mathrm{Sub}(\Omega)$ doit être non-booléenne. Mais cette dernière propriété n'impose pas que $\mathrm{Sub}(A)$ soit non-booléenne pour tout objet A du modèle, ce qui signifie qu'il est tout à fait possible d'avoir une algèbre ontologique $\mathrm{Sub}(E)$ booléenne tout en interprétant notre sémantique dans une logique intuitionniste. [97]

97. On notera d'ailleurs que la réciproque est fausse : si $\mathrm{Sub}(\Omega)$ est booléenne, alors pour tout objet A l'algèbre $\mathrm{Sub}(A)$ est booléenne. Par conséquent, restreindre l'interprétation sémantique à la logique classique incite très fortement à utiliser l'axiome d'exhaustivité, même s'il est possible d'avoir $\mathrm{Sub}(E)$ booléenne sans que ce soit le cas de $\mathfrak{h}(\mathbb{B})$.

Une seconde crainte pourrait là encore être d'ordre philosophique : si l'algèbre $\mathrm{Sub}(E)$ n'est pas booléenne, cela signifie notamment que pour un type ontologique donné, l'union de ce type et de son pesudo-complément ne suffit pas à couvrir l'ensemble des entités possibles. Pour un tel objet A, peut-on interpréter cette « partie » de E qui n'est pas atteinte par $A \cup \neg A$?[98] D'un point de vue intuitionniste, une possibilité serait de les voir comme les entités pour lesquelles on ne peut pas décider de si oui ou non elle peuvent recevoir le type associé à A, mais cette vision soulève alors un grand nombre de questions connexes, comme savoir si de telles entités ont un type tout court, et si cela ne risquerait pas de rendre le typage indécidable. Cependant, la direction de l'analyse linguistique pousserait à croire que de telles entités n'existent pas, ou bien que la question n'est finalement pas si importante, puisque les entités qui interviennent dans l'interprétation sémantique sont forcément typées : les entités accessibles sont en effet celles dont on parle, et si on peut en parler, on leur impose nécessairement un type au travers des mots et des prédications qu'on utilise pour en parler. Y a-t-il alors lieu de s'inquiéter des conséquences de l'axiome d'exhaustivité ? Étant données nos ambitions actuelles, nous doutons que les possibles conséquences philosophiques de cet axiome soient dommageables. L'étude de phénomènes linguistiques particuliers qui ne sont pas abordés dans le présent travail pourra peut-être nous donner tort par la suite, et c'est pourquoi nous préférons le souligner ici ; mais dans le cadre de la thèse discutée dans ces pages, il paraît raisonnable d'autoriser l'utilisation de cet axiome.

Il y a en résumé une certaine marge de manœuvre entre les hypothèses inhérentes à la théorie de types et les propriétés des modèles catégoriques, permettant d'imposer certaines conditions de chaque côté. En revanche, il convient de souligner ici que certains choix d'un côté peuvent contraindre les possibilités de l'autre, et pas toujours dans le sens auquel on s'attend : si beaucoup de contraintes évoquées dans le présent chapitre proviennent des théories de types et sont imposées aux modèles, l'exemple du lien entre logique interne et propriétés ontologiques nous rappelle que nous pouvons avoir besoin de recourir à des modèles avec des propriétés additionnelles spécifiques, qui peuvent avoir une influence sur la manière dont nous concevons les extensions de notre théorie de types. Dans l'optique de développer une théorie qui réponde au mieux aux problématiques linguistiques que nous avons décrites au chapitre 1, nous voyons ainsi qu'un développement double, où nous faisons évoluer notre théorie et nos modèles en parallèle, est nécessaire afin de s'assurer que les modèles accessibles à notre théorie soient suffisamment divers pour ne pas nous enfermer dans un paradigme logique trop restrictif. Dans le chapitre suivant, nous explorerons encore plus en profondeur cette interaction dans le but de définir une théorie de types avec coercions pensée pour la linguistique, et étudierons ses propriétés ainsi que celles de ses modèles.

98. La notion de « partie » utilisée ici a l'inconvénient de replacer un contexte ensembliste qui porte à confusion. En effet, si cette partie de E est entendue dans ce sens, alors on peut aisément construire le sous-ensemble concerné et travailler avec. Cependant, cette solution n'est pas toujours disponible dans un topos quelconque. Certains topos ont ainsi pour objets des ensembles enrichis de constructions supplémentaires, et même s'il est possible de voir « de l'extérieur » le contenu de ces ensembles, tous ces éléments ne sont pas toujours accessibles comme des éléments globaux de façon interne. Un exemple célèbre, très utilisé par Goldblatt [75], est le topos M-Set des ensembles munis d'une action du monoïde M lorsque celui-ci n'est pas un groupe, ayant notamment l'étrange propriété d'être bivalent mais non-classique parce que son classificateur de sous-objet est basé sur un ensemble à trois élements dont seuls deux sont accessibles comme éléments globaux.

Chapitre 6

Construction de théories de types sémantiques

Il est deux nuits, deux puits d'aveuglement, deux tables
D'obscurité, sans fin, sans forme, épouvantables,
L'algèbre, nuit de l'homme, et le ciel, nuit de Dieu.

Victor Hugo, *Toute la lyre*, III, LXVII

L'objectif de ce chapitre est de proposer une théorie des types qui répond à l'ensemble des contraintes formulées dans la première partie de cette thèse, incluant une ontologie de types, une relation de sous-typage engendrée par l'ordre sur cette dernière, des constructeurs capables de rendre compte des complexes hétérotypiques, et un système libre de coercions. Dans le chapitre précédent, nous avons notamment établi que la théorie de types montagovienne, entendue comme comprennant une logique d'ordre supérieur, avait pour modèles catégoriques les topos. Conformément à ce qui a été suggéré en section 5.4, nous allons cette fois-ci prendre la relation entre théories et modèles à contre-pied, en reformulant les principes généraux du chapitre 3 et nos exigences additionnelles éventuelles comme des contraintes catégoriques dans les topos, et utiliser les propriétés des topos alors obtenus pour en déduire une théorie de types pertinente pour nos impératifs de modélisation linguistique.

Notre point de départ est celui sur lequel nous avons conclu le chapitre précédent : dans toute la suite, on considèrera une ontologie de types $(\mathbb{B}, e, \leq)$ fixée dont on note $\mathcal{B}$ le pendant catégorique, ainsi qu'un topos $\mathcal{C}$ qui reflète cette ontologie selon les propriétés de la définition 41 donnée à la toute fin de la sous-section 5.4.3. Par ailleurs, suivant la construction donnée en sous-section 5.4.2, on considèrera un ensemble K de coercions de sous-typage isomorphe à l'ordre strict $<$ sur cette ontologie, chaque élément de cet ensemble dénotant également par convention un monomorphisme dans $\mathcal{C}$. Le classificateur de sous-objet de $\mathcal{C}$ est toujours noté Ω, et 0 et 1 sont ses objets initial et terminal, respectivement. Ceci forme l'architecture minimale du topos $\mathcal{C}$: les objets issus de l'ontologie et les trois objets universels ci-dessus forment la plus petite collection d'objets de base a priori nécessaire à nos modèles linguistiques, encodant toutes les composantes d'arité nulle dans la théorie que nous voulons construire. Tous les autres objets sont générés par les limites, colimites ou exponentielles de $\mathcal{C}$ et de ses catégories tranches, ainsi que par la propriété universelle du classificateur de sous-objet. D'autres objets indépendants peuvent exister, mais comme il n'ont aucun rôle dans notre théorie de types, nous pouvons les ignorer sans perte de généralité.

Nous cherchons désormais à étendre la théorie des types montagovienne MSTT par la prise en compte de l'ontologie $\mathbb{B}$, en étudiant notamment les opérations de sous-typage qui en découlent ainsi que notre capacité à construire des types pointés ayant un comportement adéquat. Notre première étape, discutée en section 6.1, va être d'imposer un changement de paradigme quant à la manière dont on considère les types prédicats afin de les séparer conceptuellement des coercions. Puis, la section 6.2 va étendre la théorie nouvellement formée par ce changement de paradigme à l'aide d'un système de sous-typage, qui présentera l'originalité de pouvoir modifier les types des prédicats dans les deux sens et sera pour cela qualifié de « covariant ». Nous complèterons la construction de notre théorie par l'ajout de types et de coercions spécifiques pour le traitement des hétérotypes en section 6.3. La section 6.4 proposera ensuite une théorie alternative offrant des mécanisme d'inférence, permettant d'envisager le calcul automatique des coercions au cours de la composition et offrant la possibilité de générer des coercions ad hoc qui seraient inaccessibles dans la théorie originelle. Enfin, nous conclurons en section 6.5 sur une discussion des correspondants catégoriques aux changements de types de la théorie montagovienne, ainsi que sur un exemple final exploitant les apports des différentes sections de ce chapitre pour illustrer les capacités de la théorie que nous proposons.

6.1 Le constructeur de types prédicats

Étant donné un prédicat $u : a \to t$ quelconque dans notre théorie de types, nous avons vu dans le chapitre 5 qu'il était possible de lui donner deux interprétations équivalentes dans $\mathcal{C}$ selon le jugement de typage utilisé pour construire l'interprétation : l'une comme un élément global $[\![u]\!] : 1 \to \Omega^A$ issu du jugement $\vdash u : a \to t$, et l'autre comme une application sur une variable libre $[\![ux]\!] : A \to \Omega$, issue du jugement $x : a \vdash ux : t$. Les deux sont bien équivalents par η-expansion (on supposera toujours que x n'est pas libre dans u) et lambda-abstraction (qui est accommodée dans le modèle par les propriétés des exponentielles). Nous avions cependant jugé la seconde interprétation comme plus intuitive dans le sens où l'application de u à un argument était alors interprétée comme une simple composition de morphismes. De même, nous avons vu qu'un traitement similaire permettait d'interpréter les coercions de sous-typage comme des monomorphismes, de sorte que $c : a < b$ soit assimilé à $c : A \rightarrowtail B$ dans $\mathcal{C}$. Si les coercions seules figuraient dans notre théorie comme des termes à part, cela signifierait qu'elles sont considérées comme des fonctions au même titre que les prédicats. De fait, la théorie des types simple ne distingue pas les types $a \to b$ et $a \to t$, qui sont considérés comme ayant les mêmes propriétés.

Pourtant, coercions et prédicats n'ont pas le même statut dans notre théorie, et se distinguent sur un point essentiel : les coercions sont les seuls termes fonctions de notre théorie dont le codomaine est un type ontologique, là où les prédicats sont toujours à valeur dans t — qui n'est pas ontologique et ne peut pas être coercé. Et surtout, t n'est pas n'importe quel type : son interprétation est le classificateur de sous-objet de $\mathcal{C}$, un objet disposant d'une propriété universelle unique et entretenant un lien fort avec les coercions elles-mêmes en tant que monomorphismes. La différence entre ces deux formes d'exponentielles est par ailleurs conceptuellement soulignée dans tout topos à travers la définition des objets puissances, pour lequels on privilégie d'ailleurs la notation $\mathcal{P}A$ plutôt que Ω^A. Or, nous avons vu au chapitre 4 que ses objets disposent de propriétés uniques au regard des autres exponentielles, propriétés qu'il serait envisageable d'exploiter pour la construction d'une théorie des types où tous les prédicats en bénéficieraient.

Nous proposons donc d'introduire un nouveau constructeur de types $\mathcal{P}$, destiné à remplacer la combinaison $\cdot \to t$ du constructeur de fonctions avec le type des propositions. Cependant,

pour que ce remplacement soit possible, il nous est nécessaire de décurryfier notre système : nous allons par conséquent redefinir un système qui sera comparable non pas à MSTT, mais à MSTT′ (cf. sous-section 5.4.1). À l'exception de ce constructeur, nous maintenons le reste du système de types similaire à ce qui précède. Par ailleurs, étant donné que les coercions seules ne sont pas considérées commes des termes, bien qu'elles s'assimilent à des termes de fonctions, nous nous permettons de supprimer purement et simplement le constructeur de fonction. Le système de types $\mathbb{T}$ considéré ici est librement engendré par $\mathbb{B}$ et par le système de constructeurs $\{1\,(0), t\,(0), \times(2), \mathcal{P}\,(1)\}$, c.-à-d. défini par la grammaire suivante :

$$\mathbb{T} ::= 1 \mid t \mid \mathbb{B} \mid \mathbb{T} \times \mathbb{T} \mid \mathcal{P}\mathbb{T}$$

Les interprétations de ces types dans $\mathcal{C}$ sont immédiates : les types ontologiques sont envoyés sur leurs homologues catégoriques, t est envoyé sur le classificateur de sous-objet, 1 sur l'objet terminal, et les constructeurs produit et puissance sont représentés par les foncteurs correspondants sur $\mathcal{C}$. On notera que le sous-ensemble des types prédicats est exactement l'ensemble des types dont le constructeur le plus externe est $\mathcal{P}$, ce qui tranche avec la difficulté de définition du même ensemble dans MSTT vue en sous-section 5.3.2 : c'est là l'une des vertus d'un système totalement décurryfié. [99]

Quelle influence ce nouveau système de types peut avoir sur la construction des termes ? À ce stade, nous laissons de côté les termes coercés, et reprenons l'ensemble Λ comme défini en section 5.2, dont nous reproduisons la grammaire ci-dessous : [100]

$$\Lambda ::= * \mid V \mid Q \mid \Lambda\Lambda \mid \lambda V.\Lambda \mid \langle \Lambda, \Lambda \rangle \mid \pi_i \Lambda \qquad (i \in \{1, 2\})$$

La plupart des règles d'introduction et d'élimination de ces termes restent inchangées ; les seules à être réellement affectées par la modification du système de types sont naturellement les règles d'abstraction et d'application. Les règles d'abstraction se déclinent en deux cas, selon le type du terme initial. S'il s'agit d'une proposition de type t, c'est la règle (lam_1) à gauche qui s'applique pour créer un terme de type prédicat correspondant, et s'il s'agit déjà d'un prédicat, la règle (lam_2) à gauche supprime la variable abstraite existante par une application avant d'y placer une variable fraîche représentant un produit, dont les projections fournissent l'argument d'application et la substitution qui s'applique pour remplacer la variable issue du contexte.

$$\frac{\Gamma, x : \alpha \vdash u : t}{\Gamma \vdash \lambda x.u : \mathcal{P}\alpha}\,(\mathrm{lam}_1) \qquad \frac{\Gamma, x : \alpha \vdash u : \mathcal{P}\beta \qquad y \notin \mathrm{fv}(u) \qquad y \notin \Gamma}{\Gamma \vdash \lambda y.(u(\pi_2 y))[\pi_1 y/x] : \mathcal{P}(\alpha \times \beta)}\,(\mathrm{lam}_2)$$

Une conséquence immédiate de ces règles est que les abstractions de Λ qui sont typables dans notre théorie sont nécessairement des fonctions à une variable, ce qui n'est pas étonnant compte tenu de la décurryfication induite par le système de types.

L'application est en revanche plus facile à formuler, du moins si on ne considère que la perspective d'une application totale : pour des termes à un argument, fournir un argument du bon type donne directement une proposition. La règle (app_t) suivante rend compte de cette idée.

$$\frac{\Gamma \vdash u : \mathcal{P}\alpha \qquad \Gamma \vdash v : \alpha}{\Gamma \vdash uv : t}\,(\mathrm{app}_t)$$

99. On notera la ressemblance de cette approche avec les présentations relationnelles de la théorie des types simples données dans [125, 165] par exemple, bien que nous verrons plus bas conserver une certaine présence explicite du type des propositions.

100. Comme précédemment, nous simplifions l'application sous la forme $\Lambda\Lambda$ plutôt que $(\Lambda)\Lambda$ selon l'usage habituel en lambda-calcul.

Le tableau se complique quelque peu si l'on envisage des applications partielles des prédicats, qui sont cependant nécessaires pour nos applications linguistiques. Afin d'alléger la présentation de ce cas de figure, nous rappelons le raccourci syntaxique définissant un n-uplet $\langle u_1, \dots, u_n\rangle$ comme $\langle u_1, \langle \dots, u_n\rangle\rangle$, et posant pour tout $i \in [\![1, n]\!]$ la projection π_i comme la composée de π_1 avec $(i-1)$ fois π_2. Une application partielle n'est alors possible que si le type de l'argument fourni « préfixe » le type attendu par le prédicat, c.-à-d. si l'argument fourni est de type α et que le type de l'argument peut s'écrire $\alpha \times \beta$ avec β non vide, à associativité près du produit. [101] Pour $n > 1$ et $i \in [\![1, n-1]\!]$, si l'on suppose que les types $\alpha_1, \dots, \alpha_n$ ne sont pas des produits, et si l'on adopte pour tous entiers j, k la convention de notation $\prod_{\ell=j}^{k} \alpha_\ell = \alpha_j$ lorsque $j = k$, nous pouvons alors définir la règle d'application partielle (app_p) comme ci-dessous :

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} \alpha_\ell) \qquad \Gamma \vdash v : \prod_{\ell=1}^{i} \alpha_\ell}{\Gamma \vdash uv : \mathcal{P}(\prod_{\ell=i+1}^{n} \alpha_\ell)}\,(\mathrm{app}_p)$$

Ces règles sont bien entendu cohérentes avec le modèle catégorique $\mathcal{C}$, par exploitation des propriétés d'exponentielles des objets puissances. Les cas (lam_1) et (app_t) sont des applications directes de ces propriétés, nous laissons aux lectrices et lecteurs le soin de le vérifier. La règle (lam_2) correspond à la commutativité du diagramme suivant, qui permet de définir $[\![\lambda y.(u(\pi_2 y))[\pi_1 y/x]]\!] = \lambda(\mathrm{ev} \circ ([\![u]\!] \times \mathrm{id}_B) \circ i)$, où i est l'isomorphisme d'associativité du produit.

$$\begin{array}{ccccc}
[\![\Gamma]\!] \times (A \times B) & \xrightarrow[\sim]{i} & ([\![\Gamma]\!] \times A) \times B & \xrightarrow{[\![u]\!] \times \mathrm{id}_B} & \mathcal{P}B \times B \\
\downarrow{\scriptstyle \lambda \times \mathrm{id}_{A\times B}} & & & & \downarrow{\scriptstyle \mathrm{ev}_B} \\
\mathcal{P}(A \times B) \times (A \times B) & & \xrightarrow{\qquad\qquad \mathrm{ev}_{A\times B} \qquad\qquad} & & \Omega
\end{array}$$

De façon similaire, la règle d'application partielle (app_p) suit simplement la commutativité du diagramme ci-dessous, et permet donc de définir dans ce cas $[\![uv]\!] = \lambda(\mathrm{ev} \circ j \circ (\langle [\![u]\!], [\![v]\!]\rangle \times \mathrm{id}))$, où j est un isomorphisme.

$$\begin{array}{ccc}
[\![\Gamma]\!] \times (\prod_{\ell=i+1}^{n} A_\ell) & \xrightarrow{\langle [\![u]\!], [\![v]\!]\rangle \times \mathrm{id}} & \mathcal{P}(\prod_{\ell=1}^{n} A_\ell) \times (\prod_{\ell=1}^{i} A_\ell) \times (\prod_{\ell=i+1}^{n} A_\ell) \\
 & & \sim \downarrow{\scriptstyle j} \\
\downarrow{\scriptstyle \lambda \times \mathrm{id}} & & \mathcal{P}(\prod_{\ell=1}^{n} A_\ell) \times (\prod_{\ell=1}^{n} A_\ell) \\
 & & \downarrow{\scriptstyle \mathrm{ev}} \\
\mathcal{P}(\prod_{\ell=i+1}^{n} A_\ell) \times (\prod_{\ell=i+1}^{n} A_\ell) & \xrightarrow{\qquad \mathrm{ev} \qquad} & \Omega
\end{array}$$

En ce qui concerne les conversions, le remplacement de $\to$ par $\mathcal{P}$ n'apporte pas de modifications drastiques : la théorie équationnelle entre termes est similaire à MSTT, les égalités se

101. L'associativité du produit dans le système de type n'est par défaut pas requise car elle demanderait de quotienter l'ensemble des termes par la relation d'équivalence obtenue pour les produits dont les interprétations sont isomorphes ; nous préférons maintenir ici un cadre aussi général que possible en évitant ce type de contraintes. La présente discussion rend cependant possible d'imposer par défaut l'associativité des produits sous la portée d'un constructeur de prédicat, de sorte que $\mathcal{P}\big((\alpha \times \beta) \times \gamma\big) = \mathcal{P}\big(\alpha \times (\beta \times \gamma)\big)$ soit vérifié pour tous types α, β et γ dans $\mathbb{T}$. Nous laissons toutefois de côté ici les implications qu'aurait une relation d'égalité entre les types eux-mêmes.

propageant comme à l'accoutumée dans les cas de l'abstraction et de l'application totales. Pour la règle d'abstraction (lam_2), il est de même aisé de se convaincre que la substitution menée préserve les égalités existantes. Dans le cas de l'application partielle, les égalités entre termes se propagent aussi, mais la règle de β-conversion associée requiert une attention particulière : puisque le terme résultant est toujours une abstraction, nous changeons la variable abstraite et substituons la précédente par un n-uplet formé des arguments fournis et de cette nouvelle variable, en suivant toujours notre raccourci syntaxique. La conversion correspondante est alors la suivante :

$$\frac{\Gamma \vdash \lambda x.u : \mathcal{P}(\prod_{\ell=1}^{n} \alpha_\ell) \qquad \Gamma \vdash v : \prod_{\ell=1}^{i} \alpha_\ell \qquad y \notin \mathrm{fv}(u, v) \qquad y \notin \Gamma}{\Gamma \vdash (\lambda x.u)v = \lambda y.u[\langle \pi_1 v, \ldots, \pi_i v, y\rangle / x] : \mathcal{P}(\prod_{\ell=i+1}^{n} \alpha_\ell)} \, (\beta_p)$$

Une comparaison attentive du diagramme commutatif ci-dessus, adapté comme nécessaire, avec l'interprétation du terme de droite, permet de vérifier que cette règle est cohérente dans le modèle $\mathcal{C}$. Remarquons au passage qu'elle a pour conséquence la dérivabilité d'une autre règle, qui s'obtient facilement par une η-expansion suivie d'une β_p-conversion et d'une application, en reconnaissant l'égalité $w = \langle \pi_1 w, \ldots, \pi_{j-i} w\rangle$:

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} \alpha_\ell) \qquad \Gamma \vdash v : \prod_{\ell=1}^{i} \alpha_\ell \qquad \Gamma \vdash w : \prod_{\ell=i+1}^{j} \alpha_\ell}{\Gamma \vdash (uv)w = u\langle \pi_1 v, \ldots, \pi_i v, \pi_1 w, \ldots, \pi_{j-i} w\rangle : \mathcal{P}(\prod_{\ell=j+1}^{n} \alpha_\ell)}$$

Peut-on alors affirmer avoir une théorie de types aussi expressive que $\mathrm{MSTT}_{\mathbb{B}}$, la théorie montagovienne enrichie par les types de base dans $\mathbb{B}$? [102] Malheureusement, ce n'est pas le cas à cause d'un effet secondaire qu'il paraît important de souligner ici : comme nous avons limité la construction des fonctions au cas des prédicats, il n'existe pas dans notre nouvelle théorie de fonction identité, à part pour les propositions ! Le terme $\lambda x.x$ n'est ainsi typable que par $\mathcal{P}t$. D'autres termes pouvant possiblement implémenter des fonctions à valeurs dans les entités sont de même restreintes en typage, comme par exemple l'expansion de la projection $\lambda x.\pi_1 x$ limitée aux types de la forme $\mathcal{P}(t \times \alpha)$. Il faut donc reconnaître que notre nouveau calcul est fondamentalement moins expressif que le système montagovien, dans le sens où certaines dérivations de typages possibles en MSTT sont impossibles dans cette théorie. Il n'est cependant pas moins expressif en terme de logique interne, dans la mesure où notre manipulation du système de types maintient notre capacité à construire des formules en logique d'ordre supérieur, avec les mêmes constantes de connecteurs : ainsi, les types de $\neg$ et $\wedge$ dans ce nouveau calcul seront respectivement $\mathcal{P}t$ et $\mathcal{P}(t \times t)$, et pour tout type α la constante $\exists_\alpha$ aura le type $\mathcal{P}\mathcal{P}\alpha$. Dans ce dernier cas bien sûr, la possibilité de quantifier sur les fonctions à valeurs dans les entités est perdue, mais nous estimons que ce n'est pas un problème pour notre objectif de modélisation linguistique puisque ce genre de quantification n'a à notre connaissance jamais été utilisé en sémantique formelle.

Malgré toutes ces variations et cette perte d'expressivité, il convient de souligner que ce nouveau calcul — appelons-le théorie des types prédicats, abrégé en PTT — n'est pas une modification drastique des théories montagoviennes, et est même parfaitement « codable » dans de telles théories. Rappelons que PTT ne diffère de $\mathrm{MSTT}_{\mathbb{B}}$ que par le système de types $\mathbb{T}$ et par le remplacement des règles d'application, et d'abstraction par celles présentées deux pages plus haut, et par l'ajout de la règle de conversion (β_p). Il est alors possible de « simuler » PTT

102. En dépit de la décurryfication naturellement présente dans notre calcul, celui-ci n'est pas comparable avec MSTT' pour la raison que l'application partielle n'est pas définie dans cette théorie, bien qu'accessible indirectement par construction du membre de droite de la règle (β_p).

en $\mathrm{MSTT}_{\mathbb{B}}$ dans le sens où les dérivations du premier correspondent systématiquement à des dérivations du second, dans lesquelles les prédicats deviennent des fonctions curryfiées. Plus précisément, posons θ la fonction qui agit comme l'identité sur 1, t et les types ontologique, agit comme un homomorphisme pour $\times$, et est définie pour les types prédicats par :

$$\theta(\mathcal{P}(\alpha_1 \times \cdots \times \alpha_n)) = \theta(\alpha_1) \to \cdots \to \theta(\alpha_n) \to t$$

Par ailleurs, soit dcr la fonction de décurryfication restreinte aux termes, définie comme θ_3 de l'exemple 13. On a alors le résultat suivant :

Proposition 32. *Pour tout jugement de typage $\Gamma \vdash u : \sigma$ dérivable dans* PTT*, si u n'est pas une variable ni une application partielle, alors il existe un terme u' tel que* $\mathrm{dcr}(u')$ *se convertit en u dans* $\mathrm{MSTT}_{\mathbb{B}}$ *et un contexte Γ' tels que $\Gamma' \vdash u' : \theta(\sigma)$ est dérivable dans* $\mathrm{MSTT}_{\mathbb{B}}$*. De même, pour tout jugement de conversion $\Gamma \vdash u = v : \sigma$ dérivable de mêmes conditions sur u et v, il existe u' et v' comme précédemment et Γ' tels que $\Gamma' \vdash u' = v' : \theta(\sigma)$ est dérivable dans* $\mathrm{MSTT}_{\mathbb{B}}$*.*

Démonstration. L'idée générale de cette preuve est de curryfier les lambda-termes tout en supprimant l'utilisation de variables pourvues d'un type produit. Commençons par définir une fonction ρ sur les contextes grâce à l'induction suivante, pour tous contextes Γ, types $\alpha_1, \ldots, \alpha_n$ et β dans $\mathbb{T}$ qui ne sont pas des produits et $\alpha \in \mathbb{T}$ quelconque :

$$\begin{aligned} \rho(x : \beta) &= x : \beta \\ \rho(x : \alpha_1 \times \cdots \times \alpha_n) &= x_1 : \alpha_1, \ldots, x_n : \alpha_n \\ \rho(\Gamma, x : \alpha) &= \rho(\Gamma), \rho(x : \alpha) \end{aligned}$$

On supposera que les variables ainsi introduites sont fraîches. On définit alors un morphisme de dérivations de jugements de typage par induction sur l'ensemble des règles de typages qui constituent PTT, et on s'assure que la condition de conversion est respectée sauf pour les variables.

— Pour (ax), deux cas se présentent : si le type σ dans le jugement $\Gamma, x : \sigma, \Gamma' \vdash x : \sigma$ n'est pas un produit, il suffit de remplacer le contexte par son image par ρ. Dans le cas contraire, où $\sigma = \alpha_1 \times \cdots \times \alpha_n$, la règle est remplacée par la dérivation suivante, où Γ'' désigne le contexte $\rho(\Gamma), x_1 : \alpha_1, \cdots, x_n : \alpha_n, \rho(\Gamma')$:

$$\frac{}{\Gamma, x : \sigma, \Gamma' \vdash x : \sigma} \quad \rightsquigarrow \quad \dfrac{\dfrac{}{\Gamma'' \vdash x_1 : \alpha_1}\,(\mathrm{ax}) \qquad \dfrac{\ldots \qquad \dfrac{}{\Gamma'' \vdash x_n : \alpha_n}\,(\mathrm{ax})}{\vdots}\,(\mathrm{pair})}{\Gamma'' \vdash \langle x_1, \langle \ldots, x_n \rangle \rangle : \sigma}\,(\mathrm{pair})$$

— Pour les règles (const), (unit), (pair), (proj_1) et (proj_2), il suffit de remplacer les contextes par leur image par ρ et de laisser le reste de la dérivation inchangée. Comme les égalités se propagent à travers tous les constructeurs associés, la propriété de conversion est bien maintenue.

— Le cas de (lam_1) correspond à la règle (lam) classique si le type α de la variable n'est pas un produit, en remplaçant alors $\mathcal{P}\alpha$ par $\alpha \to t$ et en appliquant toujours notre transformation ρ aux contextes ; et s'il s'agit d'un produit $\alpha_1 \times \cdots \times \alpha_n$ on construit une nouvelle dérivation par une série de règles (lam) qui curryfie la fonction originale sur les nouvelles variables

introduites par ρ. Afin de propager le remplacement des variables initié pour (ax), on note ici u' pour $u[\langle x_1, \cdots, x_n\rangle/x]$.

$$\frac{\Gamma, x : \alpha_1 \times \cdots \times \alpha_n \vdash u : t}{\Gamma \vdash \lambda x.u : \mathcal{P}(\alpha_1 \times \cdots \times \alpha_n)} \quad \rightsquigarrow \quad \frac{\dfrac{\rho(\Gamma), x_1 : \alpha_1, \ldots, x_n : \alpha_n \vdash u' : t}{\vdots}\,(\text{lam})}{\rho(\Gamma) \vdash \lambda x_1 \ldots \lambda x_n.u' : \alpha_1 \to \cdots \to \alpha_n \to t}\,(\text{lam})$$

Le cas de (lam_2) est une simple adaptation du même procédé. Dans les deux cas, on vérifie que $\text{dcr}(\lambda x_1 \ldots \lambda x_n.u[\langle x_1, \ldots, x_n\rangle/x]) = \lambda x.\text{dcr}(u)[\langle \pi_1 x, \ldots, \pi_n x\rangle/x] = \lambda x.u$.

— Au vu de ce qui précède, le traitement de (app_t) et de (app_p) est assez intuitif : si la variable a un type produit, il suffit de passer en arguments successifs de la traduction du prédicats les projections de l'argument initial, donnant une dérivation de la forme ci-dessous dans le cas général. Les règles (proj_i) utilisées sont admissibles pour tout $i \in [\![1, n]\!]$, et u' et v' dénotent les termes modifiés selons les règles précédentes.

$$\frac{\Gamma \vdash u : \mathcal{P}(\alpha_1 \times \cdots \times \alpha_n) \qquad \Gamma \vdash v : \alpha_1 \times \cdots \times \alpha_i}{\Gamma \vdash uv : \mathcal{P}(\alpha_{i+1} \times \cdots \times \alpha_n)} \quad \rightsquigarrow$$

$$\dfrac{\dfrac{\rho(\Gamma) \vdash u' : \alpha_1 \to \cdots \to \alpha_n \to t \qquad \dfrac{\rho(\Gamma) \vdash v' : \alpha_1 \times \cdots \times \alpha_n}{\rho(\Gamma) \vdash \pi_1 v' : \alpha_1}}{\vdots} \qquad \dfrac{\rho(\Gamma) \vdash v' : \alpha_1 \times \cdots \times \alpha_i}{\rho(\Gamma) \vdash \pi_i v' : \alpha_i}}{\rho(\Gamma) \vdash u'(\pi_1 v') \ldots (\pi_i v') : \alpha_{i+1} \to \cdots \alpha_n \to t}$$

De plus, lorsque l'application est totale, on a l'égalité suivante : $\text{dcr}\big(u'(\pi_1 v') \ldots (\pi_n v')\big) = \text{dcr}(u')(\langle \pi_1 \text{dcr}(v'), \ldots, \pi_n \text{dcr}(v')\rangle) = uv$.

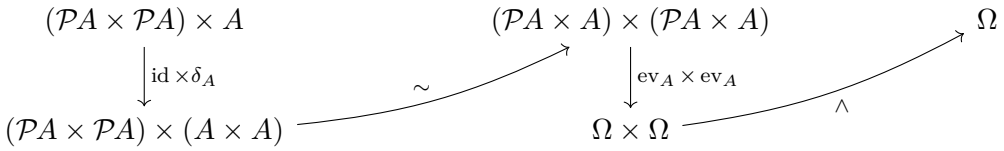
Le procédé est globalement similaire pour les dérivations de la théorie équationnelle. Les règles de PTT sont également présentes dans MSTT ne posent aucune difficulté, y compris pour la β-conversion. Un cas bien plus subtil est celui de la règle (β_p) pour les conversions d'applications partielles : on vérifie que si u' comme ci-dessus s'écrit $\lambda x_1 \ldots \lambda x_n.u[\langle x_1, \ldots, x_n\rangle/x]$ comme obtenu par les règles d'abstraction, alors le terme en conclusion de cette même dérivation se réduit à $\lambda x_{i+1} \ldots \lambda x_n.u[\langle \pi_1 v', \ldots, \pi_i v', x_{i+1}, \ldots, x_n\rangle/x]$ par β-conversions successives, rendant admissible une dérivation correspondant à la règle (β_p). □

On remarquera par ailleurs que si la décurryfication du terme équivalent à l'application partielle n'est pas définie, son réduit par la règle de β_p-conversion l'est bel et bien, de sorte qu'un lien indirect entre le terme original dans PTT et son correspondant dans MSTT existe toujours. Dans le cas des variables, le problème est l'apparition des n-uplets $\langle x_1, \ldots, x_n\rangle$ qui ne sont pas transformés par la décurryfication, même si leur relation avec la variable d'orginie est évidente au vu de la transformation de contexte. Ceci permet donc d'établir que notre nouveau calcul PTT n'est pas très éloigné des systèmes que nous connaissons déjà : seuls le changement conceptuel entre fonctions et prédicats ainsi que la possibilité d'application partielle peuvent être sources de confusion, mais se ramènent aisément à des correspondants curryfiés dans MSTT.

Avant de passer à la section suivante où nous traiterons de l'intégration des coercions de sous-typage à PTT, nous nous proposons d'introduire quelques raccourcis pratiques pour les connecteurs logiques. En effet, la philosophie qui a guidé l'introduction des types prédicats nous encourage à privilégier la manipulation directe des prédicats pour les opérations logiques communes, en utilisant uniquement des règles d'applications totales ou partielles pour construire

de nouveaux prédicats. La raison pour cela nous vient des fondements mêmes de l'approche montagovienne : le fait que tout élément linguistique s'interprète comme un prédicat, comme vu au chapitre 1. La plupart des prédicats qui interprètent ces éléments, notamment au premier ordre, sont des constantes, or l'articulation de ces constantes en formules plus complexes obéit le plus souvent à une méthode où elles sont appliquées à des variables pour fournir une proposition puis connectées entre elles par des connecteurs logiques, avant que l'on n'abstraie à nouveau les variables pour retrouver un prédicat. Dans PTT, nous préférerons si possible éviter l'introduction de variables et l'abstraction, pour mieux nous reposer sur l'application.

Il est possible à cette fin de définir des formes généralisées des différents connecteurs. Dans notre topos $\mathcal{C}$, de telles généralisations se définissent à partir des connecteurs usuels sur le classificateur de sous-objets comme des noms de morphismes ; par exemple, pour tout objet $A \in \mathsf{obj}(\mathcal{C})$, on peut définir la généralisation de la conjonction $\wedge_A := \mathrm{name}(f_A) : \mathcal{P}A \times \mathcal{P}A \to \mathcal{P}A$, où f_A est la composée :



Les constructions pour les autres connecteurs sont similaires. Ces morphismes peut alors servir de base pour définir des constantes dans PTT. Pour tout type α, on pourra ainsi considérer $\wedge^\alpha : \mathcal{P}(\mathcal{P}\alpha \times \mathcal{P}\alpha \times \alpha)$ comme une constante [103] interprétée par le morphisme correspondant, et de même pour les autres connecteurs binaires ; tandis que la négation se généralise en $\neg^\alpha : \mathcal{P}(\mathcal{P}\alpha \times \alpha)$. Le traitement des quantificateurs est légèrement plus compliqué, et fait appels aux représentants internes des adjoints au foncteur de produit fibré d'une projection bien choisie. En effet, si les quantificateurs de base sont paramétrés par le type de la variable qu'ils lient, la généralisation permet d'ajouter comme paramètre le type du prédicat résultant. Étant donnés deux objets A et B et la projection $\pi : A \times B \to B$, ils sont alors représentés dans $\mathcal{C}$ par $\exists_\pi$ et $\forall_\pi : \mathcal{P}(A \times B) \to \mathcal{P}B$, donnant alors les constantes suivantes pour tous types $\alpha, \beta \in \mathbb{T}$:

$$\exists^{\alpha,\beta} : \mathcal{P}(\mathcal{P}(\alpha \times \beta) \times \beta) \qquad \forall^{\alpha,\beta} : \mathcal{P}(\mathcal{P}(\alpha \times \beta) \times \beta)$$

Notons que si l'on souhaitait étendre la logique de notre calcul par l'ajout de quantificateurs, les généralisations de ces derniers prendraient les mêmes types que les deux ci-dessus. [104]

6.2 Un système de sous-typage covariant

Nous avons désormais une théorie de base PTT intégrant un type des prédicats plutôt qu'un type des fonctions, qui est aussi expressif que MSTT (au cas près des identités) tout en ayant suffisamment de ressources pour des applications linguistiques basiques. Cependant, il manque encore à cette théorie un ensemble de mécanismes permettant d'accommoder la composition des différents éléments lorsque certaines conditions sont remplies, mécanismes qui suivent notamment le mouvement initié par Pustejovsky pour GL et s'incarnent dans la construction de la relation de sous-typage. Au chapitre 3, nous avons notamment vu que les spécifications des

103. Techniquement, il s'agit d'une abréviation pour le lambda-terme $\lambda x. \wedge ((\pi_1 x)(\pi_3 x))((\pi_2 x)(\pi_3 x))$.

104. Remarquons que nous n'introduisons pas le cas complémentaire des quantificateurs de type $\mathcal{P}(\mathcal{P}(\alpha \times \beta) \times \alpha)$: en effet, il est toujours possible de se ramener au cas exposé ici en remplaçant le prédicat argument $u : \mathcal{P}(\alpha \times \beta)$ par $\lambda x.u\langle \pi_2 x, \pi_1 x\rangle : \mathcal{P}(\beta \times \alpha)$, sans perte de généralité.

arguments incluaient une condition sur la composante ontologique par (O1), et que cette condition était remplie lorsque la composante fournie en argument était plus petite pour l'ordre qui régit l'ontologie de types par (O2). Or, dans les théories d'inspiration montagovienne, la composante ontologique est généralement encapsulée dans une structure de type fonctionnel, pour lequel le sous-typage est généralement pris comme contravariant : comme noté par Luo [107] et comme nous l'avions évoqué en section 1.3, ceci pose un problème général pour les expressions qui sont traitées comme des modificateurs de prédicats du premier ordre, comme les adjectifs et les adverbes.

De telles expressions présentent en effet une structure de la forme $(\cdot \to t) \to \cdot \to t$, prenant donc en arguments à la fois une fonction du premier ordre et une entité, les deux ayant le plus souvent la même composante ontologique. Mais, si a et b sont deux types ontologiques avec $a < b$, il n'est pas possible pour une expression de type $(b \to t) \to b \to t$ d'accepter un argument de type $a \to t$ puisque les règles usuelles de sous-typage imposent $b \to t < a \to t$ et non le contraire. Ceci échappe à l'intuition définie dans le principe (O2) : comme $a \to t$ respecte la structure attendue et satisfait la condition ontologique, on devrait espérer pouvoir mener la composition à bien et obtenir une construction de type bien formé. Ceci met en lumière un problème que Luo assigne à l'interaction entre le sous-typage et les théories de types montagoviennes, et qu'il résout donc en s'éloignant du modèle montagovien pour baser son système UTT sur les théories de Martin-Löf [35], ce qui a notamment pour effet de réduire les ordres dans les types sémantiques tout en nécessitant un réencodage d'une partie de l'information sémantique au sein de ces derniers, aboutissant ainsi à un système de types non minimal. Dans la présente section, nous souhaiterions traiter le même problème mais en maintenant notre théorie dans la continuité des théories montagoviennes : on comprend alors que notre seule possibilité pour ce faire consiste à proposer des changements d'approche pour le sous-typage lui-même.

L'idée serait de rester sur l'intuition fournie par le principe de combinaison des types sémantiques : si la structure est respectée et que la composante ontologique est bien plus petite que celle attendue, la composition est valide. On voit très vite que ceci suggère la *covariance* du sous-typage, c.-à-d. le fait que si $a < b$, alors $a \to t < b \to t$. Il faut cependant garder à l'esprit que dans des situations générales, un tel sous-typage pose des problème de sûreté du système de type : des expressions bien typées pourraient alors produire une erreur parce qu'une fonction finirait par y recevoir un argument qu'il ne peut pas traiter (cf. p. ex. [20]). Heureusement, notre objectif linguistique pourrait bien rendre les choses a priori plus contrôlables, et en imposant une bonne discipline sur les compositions afin de prévenir les problèmes, un tel sous-typage devrait pouvoir être accommodé à PTT. En particulier, nous allons montrer dans les prochains paragraphes que la covariance pour PTT est appuyée par des possibilités techniques inhérentes au système de types et conséquences directes du but linguistique de la théorie associée.

Dans le cas général ensembliste, la contravariance et la covariance du sous-typage pour les domaines des fonctions pose la question de la restriction ou de l'extension de ces domaines. Soient A, B et C trois ensembles, avec $A \subset B$. On peut alors considérer le sous-typage comme un ensemble de fonctions, permettant de convertir un objet en un nouvel objet en fonction des circonstances ; et l'inclusion ensembliste constitue le cas de base de telles fonctions. La contravariance permet alors d'associer à une fonction $f \in C^B$ sa restriction $f|_A \in C^A$ à A, qui est normalement définie sur tout son domaine : pour tout $x \in A$, $f|_A(x) = f(x)$. L'opération associée à cette forme de sous-typage n'est cependant pas assimilable à une inclusion : elle n'est généralement pas même injective. Mais elle a l'avantage certain de sa définition systématique, dans la mesure où toute nouvelle fonction obtenue est entièrement déterminée. À l'inverse, le problème posé par la construction d'un système covariant serait de définir une fonction qui soit capable d'associer à tout élément de C^A un élement correspondant de C^B, qui conserverait de

préférence le comportement de la fonction initiale sur les arguments déjà connus. Il s'agit donc de trouver un *prolongement* des fonctions $f \in C^A$ au domaine B — sauf que la plupart du temps, de tels prolongements sont arbitraires, risquent de mener à des valeurs incohérentes, et manquent de canonicité : trop de prolongements différents sont possibles, alors que trop peu de critères sont généralement disponibles pour en choisir.

Les prédicats offrent cependant un certain contrepoint à ces affirmations. Du fait de leur codomaine qui est précisément déterminé, ces fonctions particulières disposent d'une méthode toute désignée pour les prolongements : l'extension vers le faux. Rappelons en effet, comme nous l'avons vu dans le cadre plus général des topos, que tout prédicat ensembliste peut être vu comme la fonction caractéristique d'un sous-ensemble de son domaine. Ainsi, si $f \in \{0,1\}^A$ est un tel prédicat, il définit exactement un ensemble $D = \{x \in A \mid f(x) = 1\}$ grâce à ses valeurs vraies. Il est alors possible, pour tout B tel que $A \subset B$, de définir l'extension g de f sur B qui préserve la propriété de caractéristique sur D : il suffit pour cela de poser $g(x) = f(x)$ pour $x \in A$ et $g(x) = 0$ pour $x \in B \setminus A$. Contrairement au cas général, cette transformation a tout d'une transformation canonique : elle se généralise très simplement à tout prédicat et à tout domaine d'extension. De plus, comme les antécédents de 1 déterminent entièrement une telle fonction, la transformation $\{0,1\}^A \to \{0,1\}^B$ correspondante est injective, tout comme l'est l'inclusion initiale $A \subset B$. Et la plus grande vertu de cette observation est qu'elle n'est pas cantonnée au cas ensembliste, mais est en fait généralisable à tout topos grâce à la proposition 28 (p. 121), qui permet de convertir une caractéristique $A \to \Omega$ en une caractéristique $B \to \Omega$ s'il existe un monomorphisme $A \rightarrowtail B$. Ceci met alors en lumière la raison pour laquelle nous avons choisi de définir explicitement à la section précédente le type des prédicats comme un type distinct de celui des fonctions : par les observations qui précèdent, les prédicats sont les seules fonctions à disposer d'une méthode canonique de prolongement, et en faire un type à part permettra donc d'introduire ces méthodes comme des constructions formelles qui leur seront spécifiquement dédiées grâce à leur type. Elle permet en effet de poser les bases d'un sous-typage covariant comme nous en avons besoin : la capacité d'étendre canoniquement les prédicats assure qu'une telle opération est toujours bien définie. Par ailleurs, les propriétés injectives du cas ensembliste se généralisent en propriétés monomorphiques, systématisant la correspondance entre opérations de sous-typage et monos dans le modèle associé.

Reprenons alors l'ensemble K des coercions de base, défini isomorphiquement à la relation d'ordre sur les types ontologiques de $\mathbb{B}$. Rappelons que les symboles de K désignent également les monomorphismes correspondant dans le topos $\mathcal{C}$. Nous allons maintenant étendre cet ensemble en un nouvel ensemble $\mathcal{K}$ de *coercions de sous-typage covariant*, en introduiant des constructeurs de coercions $\times$ et $\exists$ permettant d'adapter les coercions de base à l'ensemble de notre système de types $\mathbb{T}$. On observera que ces constructeurs permettront de maintenir la correspondance entre noms de coercions et leurs représentations dans $\mathcal{C}$. Formellement, $\mathcal{K}$ est défini par la grammaire suivante :

$$\mathcal{K} ::= K \mid \mathrm{id} \mid \mathcal{K} \times \mathcal{K} \mid \exists \mathcal{K}$$

et les jugements de sous-typage associés sont définis à travers les règles suivantes, où la règle (base) est répétée de la sous-section 5.4.2 et les quatre autres sont nouvelles :

$$\frac{c \in K}{\vdash c : \kappa_1(c) < \kappa_2(c)}\,(\text{base}) \qquad \frac{\vdash c : \sigma < \tau \qquad \vdash c' : \sigma' < \tau'}{\vdash c \times c' : \sigma \times \sigma' < \tau \times \tau'}\,(\text{prod}) \qquad \frac{\vdash c : \sigma < \tau}{\vdash \exists c : \mathcal{P}\sigma < \mathcal{P}\tau}\,(\text{pred})$$

$$\frac{\vdash c : \sigma < \tau \qquad \xi \in \mathbb{T}}{\vdash c \times \mathrm{id}_\xi : \sigma \times \xi < \tau \times \xi}\,(\text{idpr}_1) \qquad \frac{\vdash c : \sigma < \tau \qquad \xi \in \mathbb{T}}{\vdash \mathrm{id}_\xi \times c : \sigma \times \xi < \tau \times \xi}\,(\text{idpr}_2)$$

Les deux dernières règles permettent de coercer une partie d'un produit sans toucher à l'autre ; on notera l'utilisation ici de coercions identités que nous ne permettons pas dans le cas général. Nous pouvons alors immédiatement établir la proposition ci-dessous.

Proposition 33. *Pour tout $c \in \mathcal{K}$, l'interprétation de c dans le topos $\mathcal{C}$ est un monomorphisme.*

Démonstration. Le cas de base est vrai par construction, il s'agit de notre hypothèse de départ. Dans le cas du produit, il suffit de montrer que si $f : A \to C$ et $g : B \to D$ sont monos, alors $f \times g : A \times B \to C \times D$ l'est également, ce qui s'avère être un résultat classique de la théorie des catégories : si $h, h' : X \to A \times B$ sont tels que $(f \times g) \circ h = (f \times g) \circ h'$, la composition avec les projections nous permet d'établir $\pi_i \circ h = \pi_i \circ h'$ pour $i \in \{1, 2\}$. Il suffit alors d'exploiter l'égalité $\langle \pi_1 \circ h, \pi_2 \circ h \rangle = h$ pour conclure. Ce résultat s'étend aux cas incluant l'identité en remarquant que toute identité est iso et donc mono dans $\mathcal{C}$. Enfin, dans le cas des prédicats, il suffit d'appliquer directement la proposition 28 mentionnée plus haut. □

Notons de plus qu'il existe au plus une seule coercion dans $\mathcal{K}$ pour une paire de types donnée, assurant ainsi la cohérence du système. Par ailleurs, on se rappellera que pour toute coercion $c : \sigma < \tau \in \mathcal{K}$, on dispose d'une construction $\mathcal{P}c : \mathcal{P}\tau \to \mathcal{P}\sigma$ qui reproduit le comportement contravariant traditionnel des fonctions, et qui est en plus une rétraction de $\exists c$. Cette construction pourra s'avérer utile, mais ne sera pas considérée ici comme une coercion de sous-typage, puisqu'elle ne respecte pas les propriétés monomorphiques de ces dernières, qu'elle ne correspond pas aux exigences de notre théorie linguistique issue du chapitre 3, et que son introduction menacerait la cohérence. Nous allons donc définir un second ensemble de *coercions arbitraires*, noté Z, que nous nous permettrons de moduler à notre guise au cours du présent chapitre. Mais pour l'instant, nous posons simplement $Z = \{\mathcal{P}c \mid c \in \mathcal{K}\}$ et introduisons un *jugement de coercions arbitraires*, distinct du jugement de sous-typage par le symbole $\to$ qui remplace le symbole $<$, que nous construisons comme suit :

$$\frac{\vdash c : \sigma < \tau}{\vdash \mathcal{P}c : \mathcal{P}\tau \to \mathcal{P}\sigma}\,(\text{rev})$$

Il sera également pratique d'introduire une notation de *coercion inverse* pour invoquer ces nouvelles coercions à partir de coercions de prédicats déjà existantes : aussi, pour $c : \mathcal{P}\sigma < \mathcal{P}\tau \in \mathcal{K}$, on définit $\bar{c} = \mathcal{P}c' \in Z$ où c' est l'unique coercion telle que $c = \exists c'$.

Il nous faut alors articuler ces coercions avec le reste de notre théorie de types. Nous étendons naturellement notre ensemble de termes avec la construction des termes coercés. Comme dans le chapitre précédent, nous évitons de considérer les coercions elles-mêmes comme des termes à part. De plus, nous conservons explicite la séparation entre $\mathcal{K}$ et Z afin de souligner que leurs conditions d'introduction et de typage seront différentes.

$$\Lambda ::= \cdots \mid \mathcal{K}\Lambda \mid Z\Lambda$$

La difficulté qui se pose alors est de préserver un certain niveau de sûreté dans nos applications sémantiques. Le cas critique où toute notre attention est requise est le même que celui qui a motivé initialement notre introduction d'un sous-typage covariant : les modificateurs de prédicats du premier ordre. Dans la plupart des cas, les prédicats du type $\mathcal{P}(\mathcal{P}a \times a)$ sous-tendent une simple annotation d'un prédicat déjà existant, par exemple au moyen de l'ajout d'une conjonction logique avec un second prédicat appliqué à la même variable que le premier, par exemple *rouge* dont la traduction logique minimale serait **red** : $\mathcal{P}p$, avec p le type ontologique des entités physiques, mais dont la traduction montagovienne serait plutôt $\lambda P \lambda x.\mathbf{red}(x) \wedge P(x) : \mathcal{P}(\mathcal{P}p \times p)$.

Ce cas de figure a la particularité de lier les deux composantes ontologiques prévues par le type, dans la mesure où les deux p se rapportent à l'application d'une seule et même variable à deux prédicats distincts. Dès lors, il n'est pas souhaitable de pouvoir utiliser librement le sous-typage covariant ici : si l'on considère une constante $\mathbf{car} : \mathcal{P}v$ où v figurerait le type hypothétique des véhicules, et une constante $\mathbf{j} : h$ où h serait le type des humains, alors la composition $(\lambda P \lambda x.\mathbf{red}(x) \wedge P(x))\,(\exists c\,\mathbf{car})\,(c'\mathbf{j})$, avec $c : v < p$ et $c' : h < p$, est bien typée, alors qu'elle se réduit à un terme contenant $(\exists c\,\mathbf{car})(c'\mathbf{j})$ qui, quoique bien typé, ne correspond absolument pas à une sémantique correcte puisque la combinaison des deux termes devrait créer une erreur de catégorie ! Ce sous-terme illustre d'ailleurs le fait que, utilisé sans restrictions, le sous-typage covariant pourrait permettre d'appliquer n'importe quel prédicat à n'importe quel argument.

Par conséquent, il y a deux méthodes d'applications du sous-typage covariant à écarter : l'utilisation non-restreinte du sous-typage à la fois sur le prédicat et sur l'argument ; et l'utilisation du sous-typage sur l'argument sans prise en considération des liens éventuels entre les composantes ontologiques des différents arguments. Mais la prémunition de ce second problème suppose d'être capable de déterminer les liens entre les composantes des arguments au sein d'un type sémantique. Une idée simple serait d'annoter les composantes ontologiques à l'aide d'indices, à la fois pour les constantes et pour les variables, et d'unifier dans les contextes et le type final ces indices lorsqu'une application entre les termes correspondants a lieu. Afin de formaliser cette idée, nous introduisons une notion de *contextes structurels* de types, dont l'objectif est de séparer plus aisément la composante ontologique et la composante structurelle.

Définition 42. *Un* contexte structurel *est un type dans le système de types $\mathbb{T}$ enrichi d'un constructeur d'arité nulle supplémentaire $\Box$. Si d est un tel contexte et si $a \in \mathbb{B}$ est un type ontologique, on notera $d[a]$ pour $d[a/\Box]$.*

Naturellement, cette notion est généralisable à de multiples substitutions en utilisant une famille de constructeurs $\Box_1, \cdots, \Box_n$ indépendamment substituables. Notons que nous n'autorisons pour ces contextes que la substitution par des types ontologiques, ce qui explique notamment pourquoi l'appellation de ces contextes met l'accent sur l'aspect structurel : ces contextes sont censés représenter la composante structurelle des types sémantiques vidée de toute information ontologique, les symboles $\Box$ marquant l'emplacement de ces informations. [105] Considérons alors nos règles de typage : pour expliciter le lien ontologique entre différents arguments, on annote chaque composante ontologique des contextes de variables avec un nombre distinct, et de même pour chaque composante ontologique introduite par le type d'une constante. La plupart des règles sont alors préservées telles quelles, mais les règles d'applications sont modifiées comme suit, où $\Gamma^{i=j}$ désigne le contexte Γ dans lequel les indices i et j sont unifiés, par exemple en les substituant par $\min(i,j)$:

$$\frac{\Gamma \vdash u : \mathcal{P}(d[a^i]) \qquad \Gamma \vdash v : d[a^j]}{\Gamma^{i=j} \vdash uv : t}\,(\mathrm{app}'_t)$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[a^{i_\ell}]) \qquad \Gamma \vdash v : \prod_{\ell=1}^{m} d_\ell[a^{j_\ell}]}{\Gamma^{(i_\ell=j_\ell)_{\ell\in[\![1,m]\!]}} \vdash uv : \mathcal{P}(\prod_{\ell=m+1}^{n} d_\ell[a^{i_\ell}])}\,(\mathrm{app}_p\text{'})$$

Cette méthode permet alors de distinguer les types $\mathcal{P}(\mathcal{P}a^1 \times a^1)$ et $\mathcal{P}(\mathcal{P}a^1 \times a^2)$, où le premier serait le type d'un terme comme $\lambda P \lambda x.\mathbf{f}(x) \wedge P(x)$ avec $\mathbf{f} : \mathcal{P}a$, tandis que le second serait celui d'un terme tel que $\lambda P \lambda x.P(\mathbf{k}) \wedge \mathbf{f}(x)$, avec $\mathbf{f} : \mathcal{P}a$ et $\mathbf{k} : a$. Cette différence s'incarne dans

105. On notera la similarité de cette approche avec la construction usuelle du polymorphisme : elle n'est en effet pas dûe au hasard, et nous élaborerons cette idée en section 6.4.

la manière dont on peut les abstraire : étant donné un type σ quelconque et i l'indice d'un type ontologique a apparaissant dans σ, il sera toujours possible de construire le contexte d correspondant en posant $d = \sigma[\Box/a^i]$. Dans la suite de ce chapitre, nous supposerons toujours que les contextes utilisés respectent cette construction : lorsque nous noterons $d[a^i]$, il sera systématiquement supposé que $d[a^i][\Box/a^i] = d$, c.-à-d. que a^i n'apparaît pas dans d. Une fois cet usage entendu, nous nous permettons d'alléger la notation en retirant le marquage des indices, que nous n'aurons finalement introduit que pour appuyer le fait qu'il est possible et nécessaire de repérer les composantes ontologiques connectées entre elles.

Notre sous-typage covariant est donc utilisable sous deux modes : en coercion de l'argument lorsque l'application est totale, ou en coercion contrôlée du prédicat lui-même lors d'une application partielle où des composantes ontologiques liées sont modifiées et perdurent après l'application. Le premier est couvert par la règle (act) (pour *application coercée totale*) suivante :

$$\frac{\Gamma \vdash u : \mathcal{P}d[b] \qquad \Gamma \vdash v : d[a] \qquad \vdash c : d[a] < d[b]}{\Gamma \vdash u\,(cv) : t}\,(\text{act})$$

Cette règle ainsi que les suivantes sont bien entendu généralisables à des contextes avec plusieurs composantes en parallèle. Elle est évidemment sûre, dans le sens où tous les termes en argument sont forcés d'avoir des composantes ontologiques cohérentes avec les exigences imposées par le contexte d, et sont rassemblés avant application : il n'y a donc aucune marge pour générer une application avec un prédicat et un arguments coercés vers la même composante ontologique s'ils n'avaient pas déjà la même composante. Il y a cependant un cas limite à cette première règle, que l'on rencontre pourtant fréquemment dans les systèmes montagovien : du fait de l'interprétation des *NP* comme des prédicats du second ordre, il arrive que l'on doive appliquer un prédicat de type $\mathcal{P}\mathcal{P}a$ à un argument de type $\mathcal{P}b$ avec $a < b$, un cas qui est généralement pris en charge par le sous-typage contravariant mais qui n'est pas accessible avec (act). Nous proposons donc pour compenser une règle additionnelle d'*abaissement* extrêmement restreinte, et destinée à reproduire ce comportement :

$$\frac{\Gamma \vdash u : \mathcal{P}\mathcal{P}a \qquad \Gamma \vdash v : \mathcal{P}b \qquad c : a < b}{\Gamma \vdash u((\mathcal{P}c)v) : t}\,(\text{abais})$$

Cette nouvelle règle est bien distincte de la précédente puisque $\mathcal{P}c$ est dans Z et non dans $\mathcal{K}$ comme exigé par la dernière prémisse. Reproduire le caractère contravariant de manière globale nécessiterait des règles similaires pour $\mathcal{P}^{2k}$ et $\mathcal{P}^{2k-1}$ avec $k > 0$ au lieu de juste $k = 1$ comme ici, mais la règle ci-dessous suffit a priori pour nos applications linguistiques. Elle préserve par ailleurs la surêté de la théorie dans la mesure où le sous-typage contravariant est sûr par nature. Nous verrons par ailleurs que la théorie alternative de la section 6.4 efface la distinction entre la règle (act) et les généralisations de (abais).

Quoi qu'il en soit, ces règles totales ne sont de toute façon pas suffisantes : nous avons fréquemment besoin de réaliser des applications partielles, tout en permettant au sous-typage de s'appliquer. Ce second cas est traité par la règle générale (acp) (*application coercée partielle*) suivante, qui plutôt que d'élever le type de l'argument, prend le parti d'abaisser le type du prédicat grâce à l'usage d'une coercion inverse dans Z. Il s'agit du seul moyen d'invoquer une coercion de Z dans un terme typé, et suppose que la coercion de sous-typage associée existe bien. Pour $n \in \mathbb{N}$ et $i \in [\![1, n]\!]$, cette règle s'écrit alors :

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b]) \qquad \Gamma \vdash v : \prod_{\ell=1}^{i} d_\ell[a] \qquad \vdash c : a < b}{\Gamma \vdash (\vec{c}'u)v : \mathcal{P}(\prod_{\ell=i+1}^{n} d_\ell[a])}\,(\text{acp})$$

où c' désigne la coercion de sous-typage suivante, construite à partir de c :

$$c' : \mathcal{P}(\textstyle\prod_{\ell=1}^{n} d_\ell[a]) < \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b])$$

La vertu de cette règle est d'assurer que si la composante ontologique d'un argument a dû être diminuée de b vers a par sous-typage, alors toutes les composantes similaires liées à celle de cet argument sont également modifiées, de sorte que la prochaine application partielle doit s'accommoder avec le type a et non avec le type b, préservant là aussi la composition des problèmes de sûreté évoqués plus haut.

On notera cependant que dans le cas particulier où les types représentés par $d_{i+1}[b], \ldots, d_n[b]$ dans la règle précédente ne dépendent en fait pas du type b, il est tout à fait envisageable de se contenter de coercer l'argument comme pour l'application totale. Afin de formaliser cette variante, nous introduisons une fonction $\mathrm{co} : \mathbb{T} \to \mathsf{Set}$ associant chaque type à l'ensemble de ces composantes ontologiques, définie par induction comme suit :

$$\begin{aligned} \mathrm{co}(a) &= \{a\} \quad \text{si } a \in \mathbb{B} & \mathrm{co}(\sigma \times \tau) &= \mathrm{co}(\sigma) \cup \mathrm{co}(\tau) \\ \mathrm{co}(1) &= \varnothing & \mathrm{co}(\mathcal{P}\sigma) &= \mathrm{co}(\sigma) \end{aligned}$$

Nous pouvons alors utiliser cette fonction pour définir une variante de la règle précédente comme attendu :

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{i} d_\ell[b] \times \prod_{\ell=i+1}^{n} \sigma_\ell) \qquad \Gamma \vdash v : \prod_{\ell=1}^{i} d_\ell[a] \qquad \vdash c : a < b \qquad b \notin \bigcup_{\ell=i+1}^{n} \mathrm{co}(\sigma_\ell)}{\Gamma \vdash u(c''v) : \mathcal{P}(\prod_{\ell=i+1}^{n} \sigma_\ell)}\,(\text{acp'})$$

où c'' est la coercion $d_1[a] \times \cdots \times d_i[a] < d_1[b] \times \cdots \times d_i[b]$ construite à partir de c. Il est par ailleurs possible d'établir un lien entre ces deux variantes lorsque les conditions de la seconde sont rencontrées : par propriété du foncteur d'objet puissance, comme décrit par exemple dans [114, §IV.1], les deux coercions commutent dans la mesure où $\vec{c}' = \mathcal{P}(c'' \times \mathrm{id})$, ce qui nous permet alors d'établir la règle d'égalité suivante :

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{i} d_\ell[b] \times \prod_{\ell=i+1}^{n} \sigma_\ell) \qquad \Gamma \vdash v : \prod_{\ell=1}^{i} d_\ell[a] \qquad \vdash c : a < b \qquad b \notin \bigcup_{\ell=i+1}^{n} \mathrm{co}(\sigma_\ell)}{\Gamma \vdash (\vec{c}'u)v = u(c''v) : \mathcal{P}(\prod_{\ell=i+1}^{n} \sigma_\ell)}\,(\text{eqcp'})$$

Mais des adaptations de ce cas particulier à d'autres égalités plus générales sont possibles. Sans surprise, nous disposons ainsi d'une version de la règle précédente adaptée à l'application totale :

$$\frac{\Gamma \vdash u : \mathcal{P}d[b] \qquad \Gamma \vdash v : d[a] \qquad \vdash c : d[a] < d[b]}{\Gamma \vdash ((\mathcal{P}c)u)v = u(cv) : t}\,(\text{eqct})$$

Remarquons d'ailleurs que cette règle consacre les coercions inverses comme étant équivalentes aux coercions de sous-typage contravariant usuellement utilisées dans les théories de types plus classiques. Une autre généralisation permet d'obtenir une égalité pour toute application partielle, sans condition sur l'absence de la variable ontologique dans les types résultants : l'idée est de coercer l'argument, de mener l'application, puis de recoercer le résultat vers le type ontologique le plus bas ; cette méthode est équivalente à celle où le prédicat est coercé avant application :

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b]) \qquad \Gamma \vdash v : \prod_{\ell=1}^{i} d_\ell[a] \qquad \vdash c : a < b}{\Gamma \vdash (\vec{c}'u)v = \vec{c}''(u(c'''v)) : \mathcal{P}(\prod_{\ell=i+1}^{n} d_\ell[a])}\,(\text{eqcp})$$

Les coercions impliquées dans la règle ci-dessus sont toutes construites à partir de c, et sont liées entre elles de sorte que si k est la coercion telle que $\vec{c}'' = \mathcal{P}k$, alors $\vec{c}' = \mathcal{P}(c''' \times k)$. Dans les deux

cas comme pour les précédents, la conformité de ces règles au comportement général du topos $\mathcal{C}$ est assurée grâce aux propriétés des objets puissances.

Nous disposons dès lors d'un ensemble de règles de typage et de conversion qui permettent de manipuler un sous-typage covariant qui, dans notre cas particulier d'application sémantique, garantit un niveau de sûreté suffisant pour la formation des formules logiques. Il demeure néanmoins un dernier point sur lequel focaliser notre attention : suivant l'idée générale décrite au début de la présente section, le sous-typage covariant a pour effet sur les prédicats d'étendre leur domaine de définition en ajoutant des valeurs correspondant au faux logique, et ceci reste valable dans n'importe quel topos, y compris les topos non-classiques ou non-bivalents. Cela signifie que la transformation opérée par les relations de sous-typage sur les prédicats peut avoir des conséquences logiques au sein des formules dans lesquelles elles sont utilisées. Si l'on acceptait les coercions comme des termes typables, la coercion $\exists c : \mathcal{P}\sigma < \mathcal{P}\tau$ pour $c : \sigma < \tau$ pourrait en effet se traduire sous la forme $\exists c = \lambda u \lambda x . \exists y . (c(y) = x) \wedge u(y)$, qui contient donc un quantificateur existentiel. Mais puisque nous refusons de typer les coercions de sous-typage, nous devons prendre des moyens détournés pour nous assurer que les égalités entre formules logiques prennent en compte le comportement logique intrinsèque de ces coercions.

Nous nous appuyons pour cela sur les connecteurs généralisés introduits à la fin de la section précédente, qui permettent de modifier directement des prédicats sans nécessité d'introduire une variable, de réaliser les applications, puis d'abstraire sur cette même variable. Notre nouvel objectif est donc de déterminer, lorsqu'elles existent, des conversions générales dans des termes utilisant à la fois ces connecteurs et des coercions de sous-typage sur des prédicats, afin de mettre en évidence quelles équivalences logiques il est possible ou non de dériver dans le système logique sous-jacent. Comme précédemment, nous abordons ces interactions sur une base catégorique, en nous appuyant sur les représentations des connecteurs dans le topos $\mathcal{C}$ qui sert de modèle général à notre théorie. La proposition suivante nous permet de traiter les cas les plus faciles ; $\wedge_A$ et $\vee_A$ sont des morphismes $\mathcal{P}A \times \mathcal{P}A \to \mathcal{P}A$, et $\exists_{A,X}$ est un morphisme $\mathcal{P}(A \times X) \to \mathcal{P}X$ correspondant à $\exists\pi$ lorsque $\pi : A \times X \to X$. Dans la suite, $\mathcal{P}^+ : \mathcal{C} \to \mathcal{C}$ désignera le foncteur d'objet puissance covariant, défini par $\mathcal{P}^+ A = \mathcal{P}A$ pour tout $A \in \mathsf{obj}(\mathcal{C})$, et par $\mathcal{P}^+ f = \exists f$ pour $f \in \mathcal{C}(A, B)$.

Proposition 34. *La transformation $\vee_A$ est naturelle en A relativement à $\mathcal{P}^+$, et la transformation $\exists_{A,X}$ est naturelle en A et X relativement à $\mathcal{P}^+$.*

Démonstration. Cette démonstration sera l'occasion de présenter une méthode de « contournement » que nous utiliserons également pour les preuves suivantes, et qui consiste à se ramener à une démonstration sur des propriétés externes, portant sur les algèbres de Heyting de la forme $\mathrm{Sub}(-)$, pour en dériver des propriétés internes. Pour la naturalité de $\vee_A$ par exemple, ce que nous cherchons ultimement à prouver est, pour tout $f : A \to B$, la commutativité du carré à gauche ci-dessous :

$$\begin{array}{ccc} A & \mathcal{P}A \times \mathcal{P}A & \xrightarrow{\vee_A} \mathcal{P}A \\ \downarrow{\scriptstyle f} & \downarrow{\scriptstyle \exists f \times \exists f} & \downarrow{\scriptstyle \exists f} \\ B & \mathcal{P}B \times \mathcal{P}B & \xrightarrow{\vee_B} \mathcal{P}B \end{array} \qquad \begin{array}{ccc} \mathrm{Sub}(X \times A) \times \mathrm{Sub}(X \times A) & \xrightarrow{\cup_A} & \mathrm{Sub}(X \times A) \\ \downarrow{\scriptstyle \exists_{\mathrm{id} \times f} \times \exists_{\mathrm{id} \times f}} & & \downarrow{\scriptstyle \exists_{\mathrm{id} \times f}} \\ \mathrm{Sub}(X \times B) \times \mathrm{Sub}(X \times B) & \xrightarrow{\cup_B} & \mathrm{Sub}(X \times B) \end{array}$$

Pour obtenir l'équivalent externe de ce carré, il suffit de choisir un objet X quelconque et d'appliquer à tous les éléments du carré le foncteur $\mathcal{C}(X, -)$, puis de dérouler les isomorphismes naturels de la forme $\mathcal{C}(X, \mathcal{P}Y) \cong \mathcal{C}(X \times Y, \Omega) \cong \mathrm{Sub}(X \times Y)$ et $\mathcal{C}(X, Y \times Y) \cong \mathcal{C}(X, Y) \times \mathcal{C}(X, Y)$,

valables dans tout topos et pour tous objets X et Y. On ramène ainsi la fonction $\mathcal{C}(X, \vee_A)$ à son équivalent $\cup_A : \mathrm{Sub}(X \times A) \times \mathrm{Sub}(X \times A) \to \mathrm{Sub}(X \times A)$, la fonction $\mathcal{C}(X, \exists f)$ à $\exists_{\mathrm{id} \times f} : \mathrm{Sub}(X \times A) \to \mathrm{Sub}(X \times B)$, et ainsi de suite, de sorte à ramener notre problème initial à celui de prouver la commutativité du carré à droite ci-dessus dans Set. Une fois cette propriété établie, on pourra alors se ramener à l'égalité $\mathcal{C}(X, \exists f \circ \vee_A) = \mathcal{C}(X, \vee_B \circ (\exists f \times \exists f))$, dont on pourra ensuite déduire la propriété interne en choisissant $\mathcal{P}A \times \mathcal{P}A$ pour X, et en appliquant ces foncteurs au morphisme $\mathrm{id}_{\mathcal{P}A \times \mathcal{P}A}$.

Dans notre cas, il pourra être plus simple conceptuellement de travailler sur les sous-objets de A et B plutôt que $X \times A$ et $X \times B$, quitte à substituer par la suite les seconds aux premiers. Nous nous permettrons alors de nous passer des indices aux opérateurs d'algèbre de Heyting sur $\mathrm{Sub}(A)$ et $\mathrm{Sub}(B)$ lorsque l'ensemble dans lequel on travaille est entendu. Soit $f : A \to B$ un morphisme. Rappelons que $\exists_f : \mathrm{Sub}(A) \to \mathrm{Sub}(B)$ est adjoint à gauche au foncteur de produit fibré $f^{-1} : \mathrm{Sub}(B) \to \mathrm{Sub}(A)$, ce qui a pour conséquence notable que pour tous $U \in \mathrm{Sub}(A)$ et $V \in \mathrm{Sub}(B)$, on a l'inégalité $\exists_f(U) \leq V$ dans $\mathrm{Sub}(B)$ si et seulement si on a $U \leq f^{-1}(V)$ dans $\mathrm{Sub}(A)$. Rappelons de plus la propriété des algèbres de Heyting suivante : (i) pour tous S, T, U, $S \cup T \leq U$ si et seulement si $S \leq U$ et $T \leq U$. Pour $f : A \to B$, $S, T \in \mathrm{Sub}(A)$ et $U \in \mathrm{Sub}(B)$, on établit alors que :

$$\begin{array}{lll}
\exists_f(S \cup T) \leq U & \text{ssi} \;\; S \cup T \leq f^{-1}(U) & \text{par } \exists_f \dashv f^{-1} \\
& \text{ssi} \;\; S \leq f^{-1}(U) \text{ et } T \leq f^{-1}(U) & \text{par (i)} \\
& \text{ssi} \;\; \exists_f(S) \leq U \text{ et } \exists_f(T) \leq U & \text{par } \exists_f \dashv f^{-1} \\
& \text{ssi} \;\; \exists_f(S) \cup \exists_f(T) \leq U & \text{par (i)}
\end{array}$$

Cette équivalence étant valable pour tout U, on peut en particulier instancier U par les membres gauches des inégalités initiales et finales, et établir ainsi l'égalité $\exists_f(S \cup T) = \exists_f(S) \cup \exists_f(T)$, d'où l'on peut conclure la naturalité souhaitée.

Pour la binaturalité de $\exists_{A,X}$, nous n'aurons pas même besoin de nous ramener à des propriétés externes puisque ce connecteur est lui-même construit comme images de projections par le foncteur $\mathcal{P}^+$. L'expression des naturalités que nous voulons montrer ici se rapporte à montrer la commutativité de diagrammes où tous les morphismes sont de la forme $\exists f$ avec f morphisme ; par propriété des foncteurs, cette commutativité est établie dès que le diagramme constitué par les morphismes initiaux commute lui-même. Pour $f : A \to B$, si $\pi_A : A \times X \to X$ et $\pi_B : B \times X \to X$ désignent les secondes projections des produits, alors on a $\pi_B \circ (f \times \mathrm{id}_X) = \pi_A$ par définition. Par ailleurs, si $\pi'_A : A \times Y \to Y$ est aussi une seconde projection, alors pour tout $g : X \to Y$ on a $g \circ \pi_A = \pi'_A \circ (\mathrm{id}_A \times g)$. Par passage au foncteur $\mathcal{P}^+$, on obtient alors la commutativité des deux diagrammes ci-dessous :

$$\begin{array}{ccc}
\mathcal{P}(A \times X) & \xrightarrow{\exists_{A,X}} & \mathcal{P}X \\
{\scriptstyle \exists(f \times \mathrm{id}_X)} \downarrow & \nearrow_{\exists_{B,X}} & \\
\mathcal{P}(B \times X) & &
\end{array}
\qquad\qquad
\begin{array}{ccc}
\mathcal{P}(A \times X) & \xrightarrow{\exists_{A,X}} & \mathcal{P}X \\
{\scriptstyle \exists(\mathrm{id}_A \times g)} \downarrow & & \downarrow {\scriptstyle \exists g} \\
\mathcal{P}(A \times Y) & \xrightarrow{\exists_{A,Y}} & \mathcal{P}Y
\end{array}$$

Ceci assure la propriété souhaitée pour le quantificateur existentiel. □

La naturalité de ces transformations signifie que les connecteurs logiques représentés par les morphismes correspondants commutent avec les coercions de sous-typage des prédicats. En

conséquence, cela justifie l'introduction d'une règle de conversion pour la disjonction, et de deux règles pour la quantification existentielle correspondant à chacun de ces deux paramètres. Pour la disjonction, il s'agit de la règle suivante :

$$\frac{\Gamma \vdash u : \mathcal{P}\sigma \qquad \Gamma \vdash v : \mathcal{P}\sigma \qquad \vdash c : \mathcal{P}\sigma < \mathcal{P}\tau}{\Gamma \vdash \vee^{\tau}(c \times c)\langle u, v\rangle = c(\vee^{\sigma}\langle u, v\rangle) : \mathcal{P}\tau}\,(\vee\text{-eq}),$$

et pour la quantification existentielle, des deux règles ci-dessous :

$$\frac{\Gamma \vdash u : \mathcal{P}(\sigma \times \xi) \qquad \vdash c : \mathcal{P}(\sigma \times \xi) < \mathcal{P}(\tau \times \xi)}{\Gamma \vdash \exists^{\tau,\xi}(cu) = \exists^{\sigma,\xi} u : \mathcal{P}\xi}\,(\exists_1\text{-eq})$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\sigma \times \xi) \qquad \vdash c : \mathcal{P}(\sigma \times \xi) < \mathcal{P}(\sigma \times \zeta) \qquad \vdash c' : \mathcal{P}\xi < \mathcal{P}\zeta}{\Gamma \vdash \exists^{\sigma,\zeta}(cu) = c'(\exists^{\sigma,\xi} u) : \mathcal{P}\zeta}\,(\exists_2\text{-eq})$$

Le cas de la conjonction est légèrement plus subtil, car cette propriété de naturalité n'est plus valable. Il est cependant possible d'établir une règle similaire à ($\vee$-eq) en nous appuyant sur le fait que nos coercions de sous-typage sont toujours représentées par des monomorphismes : en effet, la proposition suivante montre alors que l'interaction entre l'opérateur de conjonction et les monomorphismes est suffisante pour justifier une telle règle.

Proposition 35. *Pour tout monomorphisme $m : A \rightarrowtail B$ dans $\mathcal{C}$ et pour tous sous-objets $S, T \in \mathrm{Sub}(A)$, on a $\exists_m(S \cap T) = \exists_m(S) \cap \exists_m(T)$.*

Démonstration. Le principe de cette démonstration sera le même que pour la disjonction à la proposition précédente. Comme m est mono, nous disposons d'une propriété plus forte pour l'adjonction $\exists_m \dashv m^{-1}$: on a en effet (i) $m^{-1}\exists_m = \mathrm{id}_{\mathrm{Sub}(A)}$. Rappelons également que (ii) m^{-1} est un homomorphisme d'algèbres de Heyting, et que (iii) dans toute algèbre de Heyting les opérateurs $\cap$ et $\Rightarrow$ sont adjoints dans le sens où pour tous éléments S, T et U, on a $S \cap T \leq U$ si et seulement $S \leq T \Rightarrow U$. Soient donc $S, T \in \mathrm{Sub}(A)$ et $U \in \mathrm{Sub}(B)$. On établit alors que :

$$\begin{array}{lll}
\exists_m(S) \cap \exists_m(T) \leq U & \text{ssi} \;\; \exists_m(S) \leq \exists_m(T) \Rightarrow U & \text{par (iii)} \\
& \text{ssi} \;\; S \leq m^{-1}(\exists_m(T) \Rightarrow U) & \text{par } \exists_m \dashv m^{-1} \\
& \text{ssi} \;\; S \leq m^{-1}(\exists_m(T)) \Rightarrow m^{-1}(U) & \text{par (ii)} \\
& \text{ssi} \;\; S \leq T \Rightarrow m^{-1}(U) & \text{par (i)} \\
& \text{ssi} \;\; S \cap T \leq m^{-1}(U) & \text{par (iii)} \\
& \text{ssi} \;\; \exists_m(S \cap T) \leq U & \text{par } \exists_m \dashv m^{-1}
\end{array}$$

Ceci étant vrai pour tout U, on peut en déduire comme précédemment l'égalité voulue. □

Comme toutes nos coercions de sous-typage sont des monomorphismes, ce résultat légitime donc l'introduction de la règle ci-dessous, qui est le reflet exact de la règle pour la disjonction :

$$\frac{\Gamma \vdash u : \mathcal{P}\sigma \qquad \Gamma \vdash v : \mathcal{P}\sigma \qquad \vdash c : \mathcal{P}\sigma < \mathcal{P}\tau}{\Gamma \vdash \wedge^{\tau}(c \times c)\langle u, v\rangle = c(\wedge^{\sigma}\langle u, v\rangle) : \mathcal{P}\tau}\,(\wedge\text{-eq})$$

Malheureusement, la situation ne sera pas aussi simple pour les autres connecteurs et la quantification universelle : des équivalents des règles ci-dessus ne pourront pas être établis, excepté dans un cas particulier. Et pour asseoir notre propos, nous allons démontrer formellement

que sous certaines conditions, ces égalités ne peuvent pas être vérifiées. Pour ces démonstrations, il nous sera à nouveau plus facile de raisonner sur la logique externe comme dans la preuve précédente, sauf qu'au lieu de démontrer des égalités, nous allons montrer des inégalités strictes. L'existence d'une telle inégalité dans la logique externe implique en effet l'impossibilité d'établir la commutativité du diagramme interne correspondant, grâce à un raisonnement par l'absurde : si la commutativité interne était vraie, alors l'égalité externe le serait aussi, et l'inégalité ne pourrait pas être stricte, ce qui est pourtant le cas. Pour tout sous-objet $U \in \mathrm{Sub}(A)$, on notera $\overline{U}$ pour $U \Rightarrow 0$; et on dira qu'un objet est *non-zéro* s'il n'est pas isomorphe à l'objet initial 0. Nous commençons par la propriété suivante :

Proposition 36. *Soit un morphisme $f : A \to B$ dans $\mathcal{C}$. Si $\overline{\exists_f(A)}$ est non-zéro, alors pour tous $U, V \in \mathrm{Sub}(A)$, on a $\exists_f(U \Rightarrow V) < \exists_f(U) \Rightarrow \exists_f(V)$.*

Démonstration. Commençons par établir l'inégalité non-stricte. Pour cela, soit $S \in \mathrm{Sub}(B)$ tel que $S \leq \exists_f(U \Rightarrow V)$. Alors, $\exists_f(U) \cap S \leq \exists_f(U) \cap \exists_f(U \Rightarrow V)$ puisque l'intersection est monotone, et comme la proposition précédente a montré la commutativité des images directes avec l'intersection, le membre de droite est égal à $\exists_f\big(U \cap (U \Rightarrow V)\big)$; or $U \cap (U \Rightarrow V) \leq V$ par propriété des algèbres de Heyting, et comme $\exists_f$ est monotone,[106] on en déduit l'inégalité $\exists_f(U) \cap S \leq \exists_f(V)$, qui permet finalement d'établir $S \leq \exists_f(U) \Rightarrow \exists_f(V)$. Ceci étant vrai pour tout S minorant $\exists_f(U \Rightarrow V)$, on peut notamment utiliser ce dernier élément qui est son propre minorant afin d'obtenir $\exists_f(U \Rightarrow V) \leq \exists_f(U) \Rightarrow \exists_f(V)$.

Pour établir l'inégalité stricte, nous allons faire entrer en jeu le sous-objet $\overline{\exists_f(A)}$, que nous supposons ici non-zéro : notre objectif est de montrer qu'il minore aussi $\exists_f(U) \Rightarrow \exists_f(V)$, mais qu'il a une intersection vide avec $\exists_f(U \Rightarrow V)$, de sorte à construire un sous-objet intermédiaire entre les deux sous-objets cibles pour lequel on aura une inégalité stricte évidente. La seconde de ces propriétés est la plus simple à établir : par définition, $\exists_f(A) \cap \overline{\exists_f(A)} = 0$, et on a $U \Rightarrow V \leq A$ puisque A est le plus grand élément de $\mathrm{Sub}(A)$. Par monotonie de $\exists_f$ puis de $\cap$, on en déduit que $\exists_f(U \Rightarrow V) \cap \overline{\exists_f(A)} \leq \exists_f(A) \cap \overline{\exists_f(A)} = 0$. Quant à la première propriété, elle exploite le même principe : par une manipulation similaire, on a $\exists_f(U) \cap \overline{\exists_f(A)} = 0$, et comme 0 est le plus petit élément de $\mathrm{Sub}(B)$ on a en particulier $\exists_f(U) \cap \overline{\exists_f(A)} \leq \exists_f(V)$, ce qui permet d'établir $\overline{\exists_f(A)} \leq \exists_f(U) \Rightarrow \exists_f(V)$. La propriété de l'union donne alors que $\exists_f(U \Rightarrow V) \cup \overline{\exists_f(A)} \leq \exists_f(U) \Rightarrow \exists_f(V)$, et comme l'union du membre de gauche est disjointe et que $\overline{\exists_f(A)}$ est non-zéro, on a l'inégalité stricte $\exists_f(U \Rightarrow V) < \exists_f(U \Rightarrow V) \cup \overline{\exists_f(A)}$, ce qui conclut notre démonstration. □

Transposée à notre théorie, cette propriété affirme la chose suivante : pour une coercion $c : \sigma < \tau$ donnée qui représente une inclusion stricte entre types (ce qui est exprimé par la condition non-zéro), il existe des arguments de type τ pour lesquel le prédicat $(\exists_c)(\Rightarrow^\sigma \langle u, v\rangle) : \mathcal{P}\tau$ donne une valeur fausse, tandis que le prédicat $\Rightarrow^\tau (\exists c \times \exists c)\langle u, v\rangle : \mathcal{P}\tau$ en donne une valeur vraie. Ces deux prédicats ne peuvent donc pas être égaux par conversion, puisque les vérités logiques ne seraient pas toutes conservées. Notons que l'hypothèse que la coercion soit stricte est loin d'être déraisonnable : par construction, toutes nos coercions sont considérées comme strictes, ce qui rend ce résultat général sur quasi-toutes les coercions possibles. En corollaire du résultat précédent, la même démarche nous permet d'obtenir un résultat équivalent pour la négation.

Proposition 37. *Soit un morphisme $f : A \to B$ dans $\mathcal{C}$. Si $\overline{\exists_f(A)}$ est non-zéro, alors pour tout $U \in \mathrm{Sub}(A)$, on a $\exists_f(\overline{U}) < \overline{\exists_f(U)}$.*

106. Ceci est une conséquence du fait que $\exists_f$ préserve les intersections, puisque par propriété des treillis (dont les algèbres de Heyting sont un cas particulier), $U \leq V$ si et seulement si $U \cap V = U$. Si $U \leq V$, on a alors $\exists_f(U) \cap \exists_f(V) = \exists_f(U \cap V) = \exists_f(U)$, d'où $\exists_f(U) \leq \exists_f(V)$.

Démonstration. Reproduire la démonstration précédente dans le cas particulier où $V = 0$. □

Le dernier cas qu'il nous reste à traiter est celui du quantificateur universel, qui tout comme l'existentiel peut être considéré sous deux paramètres, la partie quantifiée et celle restante. Il ne sera pas surprenant que la première de ces deux parties interagisse mal avec le sous-typage : cela revient en effet à vérifier que les quantificateurs universels et existentiels ne commutent généralement pas. En termes d'ensembles de sous-objets, cette propriété est capturée par la proposition suivante, qui est cette fois limitée aux monomorphismes — ce qui suffit largement à nos besoin puisque d'après la proposition 33 ci-dessus, toutes nos coercions sont bien des monomorphismes. Remarquons que cette proposition fait appel à la notion d'objet *non-vide*, caractérisée par l'existence d'au moins un élément global ayant cet objet pour codomaine. Cette hypothèse, bien qu'assez restrictive au regard de nos modèles, n'est cependant pas dénuée de sens pour nos applications linguistiques : en effet, la sémantique de la langue naturelle a fréquemment recours à des entités isolées, ne serait-ce que pour les constantes associées aux noms propres ; nous considérons par conséquent que la restriction imposée par cette condition de non-vacuité n'a qu'un impact négligeable sur l'intérêt de notre théorie.

Proposition 38. *Soient $m : A \rightarrowtail B$ un monomorphisme et X un objet quelconque de $\mathcal{C}$, et posons $m' = m \times \mathrm{id}_X : A \times X \to B \times X$. Si X est non-vide et si $A \ncong B$, alors $\forall_{\pi_{B,X}} \exists_{m'} \neq \forall_{\pi_{A,X}}$.*

Démonstration. Comme les fonctions considérées ici sont bien dans Set, il suffit pour prouver le résultat d'exhiber un élément de $\mathrm{Sub}(A \times X)$ sur lequel l'égalité ne tient pas. Notre choix se porte ici sur $A \times X$ lui-même, qui est certainement le sous-objet le plus simple à manipuler. Commençons alors par évaluer $\forall_{\pi_{A,X}}(A \times X)$. Par propriété générale des algèbres de Heyting, $A \times X = \overline{0}$ dans $\mathrm{Sub}(A \times X)$; par application de la proposition 29, on a alors :

$$\forall_{\pi_{A,X}}(A \times X) = \forall_{\pi_{A,X}}(\overline{0}) = \overline{\exists_{\pi_{A,X}}(0)} = \overline{0} = X$$

Il s'avère en revanche que $\forall_{\pi_{B,X}} \exists_{m'}(A \times X)$ est beaucoup plus difficile à évaluer directement. Néanmoins, nous allons pouvoir prouver qu'il diffère de $\forall_{\pi_{A,X}}$ en établissant une propriété incompatible avec les propriétés de ce dernier. Commençons pour cela par constater que pour tout $V \in \mathrm{Sub}(X)$, le carré suivant est celui d'un produit fibré :

$$\begin{array}{ccc} B \times X & \xrightarrow{\pi_{B,X}} & X \\ {\scriptstyle \mathrm{id}_B \times n}\uparrow & & \uparrow{\scriptstyle n} \\ B \times V & \xrightarrow{\pi_{B,V}} & V \end{array}$$

Par définition du foncteur de produit fibré, on en déduit alors que $\pi^{-1}_{B,X}(V) = B \times V$. Nous pouvons donc appliquer cette propriété à $\forall_{\pi_{B,X}} \exists_{m'}(A \times X)$. En exploitant les propriétés de l'adjonction $\pi^{-1}_{B,X} \dashv \forall_{\pi_{B,X}}$, on en déduit l'inégalité suivante :

$$B \times \forall_{\pi_{B,X}} \exists_{m'}(A \times X) = \pi^{-1}_{B,X} \forall_{\pi_{B,X}} \exists_{m'}(A \times X) \leq \exists_{m'}(A \times X)$$

Supposons que $\forall_{\pi_{B,X}} \exists_{m'}(A \times X)$ est non-vide. Il existe alors un morphisme x ayant cet objet pour codomaine, et pour domaine 1. Nous pouvons exploiter ce morphisme x pour montrer que le morphisme $\langle \mathrm{id}_B, x \circ !_B \rangle$ est une section de la première projection du produit apparaissant comme terme de gauche dans l'inégalité ci-dessus, ce qui fait de ce dernier un split épi. Ceci implique par définition des images que l'image du produit en question par $\exists_{\pi'_{B,X}}$ est égal à B. En

appliquant cette même fonction au terme de droite, on peut exploiter l'égalité $m\pi'_{A,X} = \pi'_{B,X}m'$ (avec $\pi'_{A,X} : A \times X \to A$ split épi également puisque X est non-vide) pour obtenir :

$$\exists_{\pi'_{B,X}}\exists_{m'}(A \times X) = \exists_m\exists_{\pi'_{A,X}}(A \times X) = \exists_m(A),$$

ce dernier terme désignant simplement A lui-même en tant que sous-objet dans $\mathrm{Sub}(B)$. Ayant vu dans les démonstrations précédentes que $\exists_{\pi'_{B,X}}$ est monotone, son application à l'inégalité précédente conduit à $B \leq A$ dans $\mathrm{Sub}(B)$, ce qui implique alors $A \cong B$ dans $\mathcal{C}$. Mais ceci est absurde au regard des hypothèses de la proposition, ce qui nous amène à conclure que $\forall_{\pi_{B,X}}\exists_{m'}(A \times X)$ est en fait vide. Or, $\forall_{\pi_{A,X}}(A \times X) = X$ est non-vide par hypothèse ; un même objet ne pouvant être à la fois vide et non-vide, nous avons par conséquent établi l'inégalité $\forall_{\pi_{A,X}}(A \times X) \neq \forall_{\pi_{B,X}}\exists_{m'}(A \times X)$, qui conclut la démonstration. □

En d'autres termes, cette proposition nous indique l'impossibilité dans le cas général d'établir une équivalence logique entre $\forall^{\tau,\xi}(cu) : \mathcal{P}\xi$ et $\forall^{\sigma,\xi}u : \mathcal{P}\xi$, pour $u : \mathcal{P}\sigma$ et $c : \mathcal{P}\sigma < \mathcal{P}\tau$. En revanche, nous allons voir grâce à la proposition suivante qu'il y a bien une interaction possible entre le quantificateur universel et le sous-typage pour la partie d'un prédicat qui n'est pas quantifiée, ce qui nous permettra alors d'introduire une dernière règle de conversion équivalente à ($\exists_2$-eq) plus haut.

Proposition 39. *Soient $m : X \rightarrowtail Y$ un monomorphisme et A un objet quelconque non-vide dans C. Alors, on a $\forall_{\pi_{A,Y}}\exists_{(\mathrm{id}_A \times m)} = \exists_m\forall_{\pi_{A,X}}$.*

Démonstration. Comparée aux précédentes, cette démonstration se révèle sensiblement plus difficile à mener par les méthodes élémentaires que nous avons utilisées jusque là. Nous allons par conséquent mobiliser une nouvelle approche, appellée *sémantique de Kripke-Joyal* [114, §VI.6]. Du fait de l'utilisation très locale de cette approche au sein de la présente thèse, elle n'a pas fait l'objet d'une introduction dans le chapitre 4 ; nous nous contenterons ici d'une brève exposition du principe général et renvoyons les lectrices et lecteurs désireux de plus de détails vers la source citée précédemment. La sémantique de Kripke-Joyal repose sur l'idée d'équivalence entre sous-objets et formules logiques, comme soulignée en section 5.3 : si $\varphi : A \to \Omega$ est une formule logique, alors le sous-objet de A associé peut être assimilé à l'expression ensembliste $\{x \mid \varphi(x)\}$ dans le langage interne des topos, dit de Mitchell-Bénabou [114, §VI.5]. Étant donné un morphisme $f : U \to A$ quelconque, on dira que U force $\varphi(f)$ si son image se factorise à travers $\{x \mid \varphi(x)\}$. Deux cas particuliers sont alors à retenir : si $g : U \to A$, alors U force $f = g$ si et seulement les deux morphismes sont effectivement égaux dans $\mathcal{C}$, et si $n : V \rightarrowtail A$ est un sous-objet de A, U force $f \in V$ si et seulement si f se factorise à travers n. L'idée de la présente preuve sera donc de considérer les formules associées à chaque membre de l'égalité qu'on cherche à prouver, et à montrer que tout morphisme de codomaine Y se factorise à travers l'un si et seulement si il se factorise à travers l'autre : ceci permettra en particulier d'appliquer le résultat sur chacun des sous-objets eux-mêmes pour en déduire leur égalité dans $\mathrm{Sub}(Y)$.

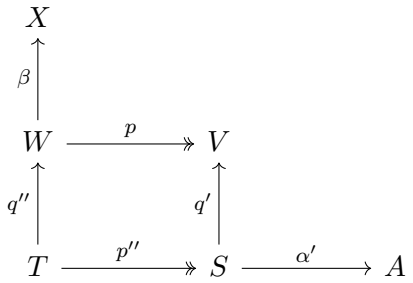
Soit $U \in \mathrm{Sub}(A \times X)$. Nous allons commencer par exprimer les formules associées à l'image de U par les deux membres de notre égalité cible. Avec un peu de simplifications grâce aux règles de la logique intuitionniste, nous pouvons établir les expressions suivantes :

$$\exists_m\forall_{\pi_{A,X}}(U) = \{y \mid \exists x\forall a.\langle a, x\rangle \in U \wedge mx = y\}$$
$$\forall_{\pi_{A,Y}}\exists_{\mathrm{id}\times m}(U) = \{y \mid \forall a\exists x.\langle a, x\rangle \in U \wedge mx = y\}$$

Nous pouvons constater que les deux formules ne diffèrent qu'au regard de l'ordre des quantificateurs. S'il est généralement impossible d'inverser deux quantificateurs différents sans modifier la valeur d'une telle formule, nous allons pourtant voir que nous nous trouvons grâce

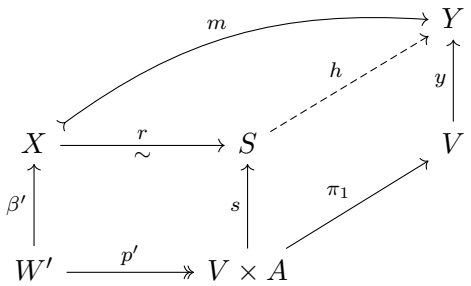
à nos hypothèses dans un cas particulier où cette inversion est en fait réalisable. Pour ce faire, nous allons explorer la sémantique de ces formules plus en détail. Soit un morphisme $y : V \to Y$ quelconque ; nous pouvons alors établir grâce aux théorèmes de [114, §VI.6] que V force $\exists x \forall a.\langle a, x\rangle \in U \wedge mx = y$ si et seulement si (1) il existe un objet W avec $p : W \twoheadrightarrow V$ et $\beta : W \to X$ tels que pour tous objet W' et morphismes $q : W' \to W$ et $\alpha : W' \to A$, W' force $\langle \alpha, \beta q\rangle \in U \wedge m\beta q = ypq$. De même, V force $\forall a \exists x.\langle a, x\rangle \in U \wedge mx = y$ si et seulement si (2) pour tout W avec $q' : W \to V$ et $\alpha' : W \to A$, il existe W' avec $p' : W' \twoheadrightarrow W$ et $\beta' : W' \to X$ tels que W' force $\langle \alpha' p', \beta'\rangle \in U \wedge m\beta' = yq'p'$. Le cœur de la démonstration sera donc d'établir l'équivalence entre (1) et (2).

Commençons par le sens direct (1) $\Rightarrow$ (2), et soient W, $p : W \twoheadrightarrow V$ et $\beta : W \to X$ l'objet et les morphismes dont l'existence est fournie par (1). Soit alors un objet S quelconque avec $q' : S \to V$ et $\alpha' : S \to A$. On construit un objet T par le produit fibré suivant :



Le morphisme p'' est alors épi car d'après la proposition 23, les produits fibrés dans un topos préservent les épis. Posons $\beta'' = \beta q''$. Grâce à notre hypothèse (1), nous savons que T force $\langle \alpha' p'', \beta q''\rangle \in U \wedge m\beta q'' = ypq''$. Donc, par définition de β'' et par l'égalité $pq'' = q'p''$ contenue dans le produit fibrée, nous en déduisons que T force $\langle \alpha' p'', \beta''\rangle \in U \wedge m\beta'' = yq'p''$, qui correspond bien à la propriété énoncée en (2). Cette construction étant valide pour tous S, q' et α', on conclut que (2) est satisfaite.

Ce premier cas était naturellement la partie évidente de l'équivalence ; nous allons voir désormais que l'implication (2) $\Rightarrow$ (1) requiert davantage de ressources. Supposons donc que la propriété en (2) est vraie ; notre objectif est alors de fournir un objet possédant un épimorphisme vers V et un morphisme vers X satisfaisant les propriétés attendues en (1), et nous allons ici montrer que le produit $V \times A$ convient à ce rôle. Commençons pour cela par remarquer que $\pi_1 : V \times A \to V$ est split épi puisque A est non-vide par hypothèse, ce qui permet de construire une section de π_1 comme dans la démonstration précédente. Montrons ensuite qu'il existe un morphisme $V \times A \to X$. Par (2), il existe un objet W' et deux morphismes $p' : W' \twoheadrightarrow V \times A$ et $\beta' : W' \to X$ tels que W' force $\langle \pi_2 p', \beta'\rangle \in U \wedge m\beta' = y\pi_1 p'$. Nous considérons alors l'objet S obtenu par la somme amalgamée suivante :



Comme les sommes amalgamées préservent les épis (dual de la proposition 15), r est épi ; or $m = h \circ r$ et m est mono, donc r est également mono par proposition 2. Par conséquent,

comme nous travaillons dans un topos, r est un isomorphisme, et $S \cong X$. Nous pouvons alors poser le morphisme $\beta : V \times A \to X$ comme $\beta = r^{-1}s$. Par construction, ce morphisme vérifie $\langle \pi_2 p', \beta p' \rangle \in U$, ce qui permet de déduire $\langle \pi_2, \beta \rangle \in U$ par localité (cf. [114, §VI.6]) car p' est épi, ainsi que $m\beta p' = y\pi_1 p'$, ce qui nous permet de déduire l'égalité $m\beta = y\pi_1$ toujours car p' est épi. Cette seconde propriété nous permet même d'aller plus loin, et nous verrons que cela sera utile par la suite : nous pouvons en effet obtenir un objet T par la somme amalgamée du β nouvellement défini avec π_1. Par un raisonnement similaire à ceci que nous venons d'établir, on a alors également $T \cong X$, et il existe donc un morphisme $\beta'' : V \to X$ tel que $\beta''\pi_1 = \beta$.

Maitenant que $V \times A$ est établi comme ayant les morphismes nécessaires, il ne nous reste plus qu'à montrer qu'il subvient aux exigences de la propriété (1). Soit donc un objet W quelconque avec $q : W \to V \times A$ et $\alpha : W \to A$. Par composition, il est immédiat que $m\beta q = y\pi_1 q$, qui est donc forcé par W. De plus, par composition toujours, $\langle \pi_2, \beta \rangle \circ \langle \pi_1 q, \alpha \rangle$ se factorise à travers U. Or, en mobilisant le résultat supplémentaire $\beta = \beta''\pi_1$ obtenu précédemment, on a :

$$\begin{aligned}
\langle \pi_2, \beta \rangle \circ \langle \pi_1 q, \alpha \rangle &= \langle \pi_2 \langle \pi_1 q, \alpha \rangle, \beta \langle \pi_1 q, \alpha \rangle \rangle \\
&= \langle \alpha, \beta'' \pi_1 \langle \pi_1 q, \alpha \rangle \rangle \\
&= \langle \alpha, \beta'' \pi_1 q \rangle \\
&= \langle \alpha, \beta q \rangle
\end{aligned}$$

Par conséquent, on a bien $\langle \alpha, \beta q \rangle \in U$. Ceci étant vrai pour tous W, q et α, nous pouvons alors en conclure que la propriété (1) est bien vérifiée, ce qui achève notre démonstration. □

Relevons à nouveau que considérer A comme non-vide n'est pas une contrainte très forte dans notre perspective linguistique, puisqu'elle s'accorde avec notre besoin d'accéder à des entités uniques pour chaque type considéré. Cette proposition justifie alors l'introduction de la règle suivante, qui complète notre étude sur l'interaction entre les coercions de sous-typage et les connecteurs logiques :

$$\frac{\Gamma \vdash u : \mathcal{P}(\sigma \times \xi) \qquad \vdash c : \mathcal{P}(\sigma \times \xi) < \mathcal{P}(\sigma \times \zeta) \qquad \vdash c : \mathcal{P}\xi < \mathcal{P}\zeta}{\Gamma \vdash \forall^{\sigma,\zeta}(cu) = c'(\forall^{\sigma,\xi} u) : \mathcal{P}\zeta} (\forall\text{-eq})$$

Par l'ajout des coercions de sous-typages, des termes coercés et des règles de typage et de conversion correspondantes, nous avons étendu notre théorie initiale PTT en une théorie encore plus complète que nous appellerons désormais CPTT pour souligner que l'extension est stricte. Il s'agit bien entendu d'une extension conservatrice de PTT, puisque tous les nouveaux termes introduits font nécessairement usage de coercions qui n'existent pas dans PTT, et puisque aucun mécanisme ne permet « d'effacer » des coercions pour établir des égalités qui n'existeraient pas déjà dans cette théorie initiale. Ceci nous permet de répondre aux exigences posées par notre théorie linguistique développée au chapitre 3, en précisant les mécanismes de composition qui permettent de vérifier les correspondances entre le critère ontologique du prédicat et la composante ontologique de l'argument. Mais notre aventure dans le développement de notre théorie de types n'est pas achevée pour autant : si la construction d'une relation de sous-typage était une condition nécessaire à un système de type adapté à la sémantique lexicale, il nous reste pour cette dernière d'autres phénomènes à capturer, parmi lesquels les fameux composés hétérotypiques, ou objets pointés, que nous avions abordés aux sous-sections 1.2.2 et 3.3.3 principalement. Dans la section suivante, nous allons ainsi tenter d'étendre CPTT avec des mécanismes supplémentaires capables de gérer ce phénomène.

6.3 Le traitement des hétérotypes

Comme nous avons pu le constater au cours des discussions du chapitre 3, les hétérotypes occupent une place à part dans la théorie des types sémantiques : il s'agit de types qui articulent parallèlement plusieurs composantes ontologiques distinctes, exprimant différentes facettes d'une même expression. Ces dernières obéissent alors à un mécanisme de sélection particulier, où le prédicat qui les prend en arguments force une coercion vers l'un des aspects disponibles. Mais il est important de noter que ces aspects ne peuvent pas être stockés séparément, comme on le ferait avec des homonymes : la polysémie qui s'exprime à travers l'hétérotypie autorise le phénomène de copredication, au cours duquel plusieurs prédicats de natures ontologiques différentes sont appliqués à une seule et même expression argument, pour laquelle il est impossible d'exhiber plusieurs types correspondant à des entrées distinctes — il doit nécessairement y avoir un unique type capable de fournir les variations attendues.

Les travaux de Pustejovsky [135] proposent l'utilisation de l'opérateur $\bullet$, parfois annoté d'une relation spécifique, pour lier plusieurs types de base, et utilise une relation de projection similaire à un sous-typage pour récuperer les différents aspects ; cette même approche se retrouve également chez Asher [4] et est également évoquée par Luo [107] comme générant des difficultés au regard de la sémantique montagovienne, pour des raisons similaires à celles qui ont motivé notre sous-typage covariant dans la section précédente. L'opérateur $\bullet$ n'est évidemment pas un opérateur « standard » : pour nous qui sommes intéressés par la construction d'une théorie des types dirigée par sa sémantique catégorique, il est nécessaire de déterminer comment un tel opérateur peut être intégré à notre topos. Nous pouvons pour cela nous appuyer sur les travaux d'Asher [4, §5], qui propose exactement un modèle catégorique de cet opérateur reposant sur l'usage de produits fibrés, que nous allons donc commencer par explorer.

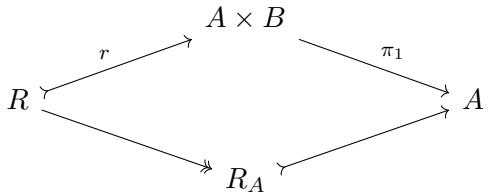
En préambule, notons que nous simplifierons légèrement le modèle d'Asher en ne considérant qu'un seul topos, et non deux comme il le propose ; nous estimons qu'aucune perte de généralité n'a lieu par ce choix puisque la représentation des objets pointés existe dans les deux topos qu'il utilise. Son modèle repose alors sur l'idée très juste qu'un objet pointé repose avant tout sur une relation entre les différents aspects — une idée également présente chez Pustejovsky et qui paraît essentielle pour la bonne compréhension de ces types. Prenons par exemple *livre*, qui exhibe un aspect d'entité physique p ainsi qu'un aspect d'entité informationnelle i : ces deux aspects sont liés par une relation de contenant-contenu, dans laquelle l'aspect physique est le support sur lequel l'aspect informationnel est inscrit. Cette relation induit une structure interne inhérente au composé $p \bullet i$ qui est associé à *livre*, et qu'on pourra également retrouver associé à des mots similaires comme *carnet*, *magazine*, *journal*, etc. La question de savoir s'il existe une autre structure possible entre ces deux types ontologiques reste ouverte (voir cependant notre discussion sur *livre* et *CD* en sous-section 1.2.2), mais nous ne pouvons pas exclure la possibilité que des hétérotypes dont la structure relationnelle interne diffère malgré l'identité de leurs types aspectuels existent effectivement ; il est donc essentiel que l'information sémantique inhérente à cette relation soit conservée.

Mais cela soulève une autre question : la nature de cette relation relève-t-elle des types sémantiques, ou bien de la connaissance lexicale ? La réponse à cette question dépendra des observations faites autour d'unités lexicales présentant une structure interne différente pour des aspects communs : s'il est avéré que de telles unités existent, il sera alors nécessaire de déterminer s'il existe des prédicats dont l'application serait signifiante pour un premier groupe mais absurde pour le second. En l'absence de tels prédicats, la distinction des deux groupes au niveau des types sémantiques ne sera pas requise, et toute l'information structurelle pourra être reléguée au lexique, ou bien intégrée à une théorie de types non-minimale. Dans le doute,

nous utiliserons des symboles abstraits pour représenter ces types ; mais ces symboles pourront aiséments être substitués par les constructions avec l'opérateur $\bullet$ si la structure se révèle non nécessaire au niveau des types sémantiques.

Considérons donc deux types ontologiques a et b incomparables, et désignons par R la relation qui structure un composé hétérotypique fondé sur les aspects a et b. Cette relation constitue un prédicat logique binaire de type $(a \times b) \to t$ dans une formulation traditionnelle. Prise dans une acception ensembliste, cette relation génère alors une fonction $in : A \to \mathcal{P}B$ définie par $in(x) = \{y \mid R(x, y)\}$ et une fonction $ex : B \to \mathcal{P}A$ définie similairement. Dans un topos, ces relations sont obtenues immédiatement par la propriété des objets puissances en tant qu'exponentielles, pour $[\![R]\!] : A \times B \to \Omega$. Asher propose alors de définir l'interprétation de $a \bullet b$ comme un objet disposant de projections vers des sous-objets spécifiques de A et B, qui sont censés décrire les entités issues de ces ensembles pour lesquels il existe une entité de l'autre avec laquelle elles sont en relation. Plus précisément, l'objet $A \bullet B$ introduit par Asher se projette vers un objet particulier qu'il note $A \times_{\mathcal{P}B} \mathcal{P}B[In]$, et qui correspondrait conceptuellement à $\{x \in A \mid \exists y.R(x, y)\}$, ainsi que vers un second objet noté $\mathcal{P}A \times_{\mathcal{P}A} B[Ex]$, qui correspondrait conceptuellement à $\{y \in B \mid \exists x.R(x, y)\}$. Le choix de ces projections très spécifiques permet alors à Asher de résoudre des problèmes de comptage que les hétérotypes peuvent poser : ses exemples considèrent notamment des livres physiques contenant une anthologie de plusieurs œuvres qui constituent en elles-mêmes des livres informationnels [107] ainsi que de multiples exemplaires d'un même ouvrage. [108]

Si nous avons une certaine idée de comment définir formellement $A \bullet B$ à partir de cette formulation (par exemple comme le produit des codomaines des deux projections attendues), nous estimons qu'il existe une manière plus simple de présenter les choses. Rappelons que la relation R peut se représenter dans notre topos $\mathcal{C}$ sous la forme d'un prédicat $[\![R]\!] : A \times B \to \Omega$. Par propriété du classificateur de sous-objet, nous savons également que ce prédicat peut se voir associer une *relation interne* par le sous-objet $R \rightarrowtail A \times B$, dont on notera r un morphisme représentant. Le domaine de r, un objet également noté R, est alors un candidat de choix pour interpréter l'objet $A \bullet B$: en effet, il incarne par définition la relation associée au composé hétérotypique et dispose de projections vers A et B par composition de r avec les projections usuelles du produit. De plus, les objets spécifiquement ciblés par Asher pour la caractérisation de ses projections sont également accesibles depuis R par factorisation épi-monique ; ainsi, l'objet $A \times_{\mathcal{P}B} \mathcal{P}B[In]$ correspond exactement à l'objet R_A obtenu par la factorisation de $\pi_1 \circ r$, comme décrit par le schéma suivant :



Par définition de la fonction $\exists_{\pi_1} : \mathrm{Sub}(A \times B) \to \mathrm{Sub}(A)$ comme générant l'image de la composée de son argument avec π_1, on vérifie en effet que $\exists_{\pi_1}(R) = R_A$ (cf. [75, §15.4]), ce qui démontre que R_A est bien le sous-objet correspondant au prédicat $\lambda x.\exists y.R(x, y) : \mathcal{P}A$ comme attendu. Il en va bien sûr de même pour l'autre aspect. Toutes les conditions sont donc réunies pour

107. Relevons toutefois que sont généralement considérées comme livres des œuvres qui ont été publiées ou imprimées séparément a une certaine période temporelle, semblant indiquer que l'aspect physique « précède » conceptuellement l'aspect informationnel. Cette précédence pourrait former un possible argument en faveur de l'existence d'un aspect « principal » des hétérotypes ; nous reviendrons sur cette idée un peu plus bas.

108. Pour des questionnements similaires en MGL, voir [116].

établir que l'objet R ainsi introduit dans $\mathcal{C}$ correspond à l'idée générale d'un type composé. Et ce procédé de construction est bien sûr généralisable à tout nombre d'aspect.

Nous voyons donc que les hétérotypes peuvent être modélisés catégoriquement par des sous-objets des produits de leurs différents aspects, sous-objets qui enregistrent nécessairement un fragment d'information relationnelle entre les aspects en question. Pour intégrer ces composés à notre théorie de type, nous allons utiliser comme annoncé plus haut un ensemble de symbole abstraits, disons $\mathbb{H}$ (pour « hétérotype »), que nous associerons canoniquement aux ensembles de types ontologiques qui constituent leurs aspects. La définition suivante fournit les fondations formelles d'un tel ensemble.

Définition 43. *Soit* $(\mathbb{B}, \leq, e)$ *une ontologie de types. Un ensemble d'*hétérotypes *sur* $\mathbb{B}$ *est une paire* $(\mathbb{H}, \mathfrak{a})$ *où* $\mathbb{H}$ *est un ensemble de symboles distinct de* $\mathbb{B}$ *et* $\mathfrak{a}$ *est une fonction* $\mathbb{H} \to \mathcal{P}(\mathbb{B})$ *telle que pour tout* $r \in \mathbb{H}$ *et tous* $a, b \in \mathfrak{a}(r)$*,* a *et* b *sont incomparables pour* $\leq$*.*

Si $r \in \mathbb{H}$ est un tel hétérotype avec $\mathfrak{a}(r) = \{a_1, \ldots, a_n\}$, son interprétation R dans $\mathcal{C}$ est alors définie comme un sous-objet $R \rightarrowtail \prod_{\ell=1}^{n} A_\ell$. Notons que par isomorphisme l'ordre des a_i n'est théoriquement pas important ; néanmoins il est possible de fixer les idées en imposant un ordre lexicographique sur $\mathbb{B}$ (total et indépendant de $\leq$) permettant de ranger les types ontologiques dans un ordre standard. Le sous-objet R dispose alors de projections correspondant à ses différents aspects, que nous allons exploiter pour former des coercions. Ces dernières sont alors globalement construites comme les coercions de sous-typage de la section précédente, avec un cas de base et une propagation aux constructeurs de types ; cependant, comme ces projections ne sont généralement pas des monomorphismes, nous préférons les conserver conceptuellement séparées du sous-typage, et les introduisons donc comme des coercions supplémentaires dans un ensemble H à part, introduit par des jugements $\vdash_{\text{H}}$, et défini par les règles ci-dessous.

$$\frac{a \in \mathfrak{a}(r)}{\vdash_{\text{H}} p_{r,a} : r \to a}\,(\text{h-base}) \qquad \frac{\vdash_{\text{H}} p : \sigma \to \tau}{\vdash_{\text{H}} \exists p : \mathcal{P}\sigma \to \mathcal{P}\tau}\,(\text{h-pred}) \qquad \frac{\vdash_{\text{H}} p : \sigma \to \tau \quad \vdash_{\text{H}} p' : \sigma' \to \tau'}{\vdash_{\text{H}} p \times p' : \sigma \times \sigma' \to \tau \times \tau'}\,(\text{h-prod})$$

$$\frac{\vdash_{\text{H}} p : \sigma \to \tau}{\vdash_{\text{H}} p \times \mathrm{id} : \sigma \times \xi \to \tau \times \xi}\,(\text{h-prid}) \qquad \frac{\vdash_{\text{H}} p : \sigma \to \tau}{\vdash_{\text{H}} \mathrm{id} \times p : \xi \times \sigma \to \xi \times \tau}\,(\text{h-idpr})$$

Nous étendons ensuite l'ensemble auxiliaire Z avec les coercions inversées définies par la règle (h-rev) suivante, et étendons l'ensemble des termes par la forme $\Lambda ::= \cdots \mid H\Lambda$.

$$\frac{\vdash_{\text{H}} p : \sigma \to \tau}{\vdash \mathcal{P}p : \mathcal{P}\tau \to \mathcal{P}\sigma}\,(\text{h-rev})$$

Remarquons que, contrairement aux coercions de sous-typage, il n'est pas prévu que les coercions hétérotypiques puissent se composer : chacune de ces coercions comporte un hétérotype dans son domaine qui est éliminé dans son codomaine. [109] La méthode d'utilisation de ces coercions se déroule alors en deux parties : il faut tout d'abord projeter l'hétérotype vers son aspect adéquat, puis appliquer le cas échéant une coercion de sous-typage pour ajuster ce type. Dans le cas d'application totale, on obtient alors la règle suivante :

$$\frac{\Gamma \vdash u : \mathcal{P}d[b] \qquad \Gamma \vdash v : d[r] \qquad \vdash_{\text{H}} p : d[r] \to d[a] \qquad \vdash c : d[a] < d[b]}{\Gamma \vdash u(c(pv)) : t}\,(\text{h-app}_t)$$

109. Nous estimons à ce stade qu'il n'y a pas de raison particulière à vouloir projeter un hétérotype vers un sous-hétérotype, c.-à-d. un hétérotype dont l'ensemble des aspects est inclus dans celui du premier. Néanmoins, il est très facile pour quiconque le souhaiterait d'étendre H à ces coercions supplémentaires : il suffit de définir un ordre partiel $\leq$ sur $\mathbb{H}$ par $r \leq r'$ si et seulement si $\mathfrak{a}(r') \subseteq \mathfrak{a}(r)$, et de poser la règle qui génère $\vdash_{\text{H}} p : r \to r'$ à partir de la prémisse $r < r'$. Il est alors possible de définir la composition des coercions et les égalités liées comme attendu.

Dans le cas de l'application partielle, il est encore une fois souhaitable d'abaisser le type du prédicat pour rendre compte de cas comme celui du modificateur de prédicat : ainsi, un *livre lourd* conserve le même type qu'un *livre*, ce qui rend possible la coprédication. La règle qui en résulte est alors la suivante :

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b]) \qquad \Gamma \vdash v : \prod_{\ell=1}^{i} d_\ell[r] \qquad \vdash_{\mathrm{H}} p : r \to a \qquad \vdash c : a < b}{\Gamma \vdash (\vec{p}'(\vec{c}'u))v : \mathcal{P}(\prod_{\ell=i+1}^{n} d_\ell[r])}\,(\text{h-app}_p),$$

où comme d'habitude, $\vec{c}' : \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b]) \to \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[a])$, tandis que $\vec{p}'$ désigne la coercion $\mathcal{P}(\prod_{\ell=1}^{n} d_\ell[a]) \to \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[r])$. Enfin, de la même manière que dans la section précédente, il est possible de ne coercer que l'argument pour une application partielle dont le type résultant ne dépend pas de l'hétérotype r, et de formuler les égalités correspondantes, mais comme le principe général ne change pas, nous nous permettons de ne pas reproduire ces règles ici.

On observera que, au détail près que les coercions hétérotypiques ne sont pas interprétées comme des monomorphismes, leur comportement que ce soit pour la construction de coercions complexes comme pour l'utilisation au sein des termes ressemble fortement à celui des coercions de sous-typage. De plus, les deux types de coercions semblent se composer parfaitement, suggérant une forte capacité d'interaction entre les deux classes. Il est donc clair ici qu'il serait possible de simplifier notre théorie d'une certaine lourdeur en ajoutant simplement les projections hétérotypiques de base à l'ensemble $\mathcal{K}$ des coercions de sous-typage ; c'est d'ailleurs ce qui est fait en général dans les formalismes sémantiques qui les modélisent. Nous retenons cependant trois raisons principales pour maintenir la distinction entre les deux classes de coercions. La première est évidemment celle de la différence d'interprétation catégorique : les projections hétérotypiques n'étant pas monomorphiques, leur utilisation a des conséquences sémantiques importantes, notamment du point de vue du comptage des objets ; cet aspect est justement central dans le traitement proposé par Asher. La seconde, toujours dans cette même perspective, est le fait que la propriété monomorphique du sous-typage autoriserait dans l'absolu à « effacer » les coercions de sous-typage des termes finaux, à la manière du traitement définitionnel de ceux-ci dans UTT (cf. [110]), car l'information qu'elles portent est importante pour la composition elle-même mais peu utile en aval du traitement sémantique — ce qui n'est pas le cas des projections hétérotypiques qui renseignent sur l'aspect sélectionné, et doivent nécessairement être conservées. La troisième enfin a trait à la modélisation des capacités créatives du langage, que nous aborderons dans la section suivante : nous verrons alors que cette modélisation repose sur la création d'hétérotypiques locaux et de coercions associées, ce qui donne à H une certaine évolutivité ; or, il nous paraît primordial que le système de sous-typage ne soit pas aussi évolutif mais agisse au contraire comme une structure fixe et constante, servant de point de référence à la composition : séparer les deux classes de coercions est le meilleur moyen de maintenir ce comportement.

À des fins d'exhaustivité, il convient d'inclure dans notre discussion des coercions hétérotypiques le cas de MGL, dans lequel les coercions ne dépendent pas du type mais sont rattachées aux entrées au sein du lexique [149, 116]. Comme présenté par Retoré [149], deux noms comme *classe* et *promotion*, qui désignent des groupes d'étudiants, se distinguent par le fait que le premier peut également désigner la salle dans laquelle le groupe se retrouve. Si g est la composante ontologique de ces entrées lexicales, et si s est la composante ontologie correspondant à l'aspect « salle », alors l'entrée pour *classe* dispose d'une coercion $c : g \to s$ qui lui est spécifiquement associée. Cette même coercion peut se retrouver pour d'autres entrées ayant la même composante ontologique, mais pas toutes, puisque *promotion* en est un contre-exemple. Comment adapter ce modèle à notre théorie ? Le fait que la coercion $c : g \to s$ ne soit pas disponible partout suggère qu'elle peut être considérée comme une fonction partielle. En théorie des catégorie, il

est possible de reproduire ce comportement à l'aide de *morphismes partiels*, dont la définition donnée ci-dessous est inspirée de [177].

Définition 44. *Soient $\mathcal{C}$ une catégorie quelconque et $X, Y \in \mathsf{obj}(\mathcal{C})$. Un* morphisme partiel *de X vers Y, noté $X \rightharpoonup Y$, est une paire (m, f) telle que $m : U \rightarrowtail X$ décrit un sous-objet de X, et f est un morphisme $U \to Y$. On dit alors que l'objet U est le* support *du morphisme partiel.*

Une interprétation de la coercion c est alors donnée par le morphisme partiel $[\![c]\!] : G \rightharpoonup S$ dans $\mathcal{C}$. Nous pouvons alors montrer que le support de ce morphisme partiel est un objet ayant le potentiel de représenter l'hétérotype d'aspects g et s, grâce au résultat suivant :

Proposition 40. *Soit $(m, f) : X \rightharpoonup Y$ un morphisme partiel. Alors, le support de ce morphisme est un sous-objet du produit $X \times Y$.*

Démonstration. La preuve de ce fait est quasi-immédiate. Soit U le support (m, f), par définition du produit comme une limite, il existe un unique morphisme $h : U \to X \times Y$ qui vérifie notamment $\pi_1 \circ h = m$. Or, m est un monomorphisme, donc par application de la proposition 2 h en est un également ; d'où le résultat. □

Cette proposition signifie donc que toute fonction partielle induit un hétérotype associé, qui est son support. Replacée dans le contexte dans notre théorie des types et de son modèle de topos, cette proposition nous donne alors une équivalence en termes de modèles entre l'introduction d'une coercion partielle $g \to s$, dont l'usage est soumise à vérification que le terme à coercé y est bien éligible via un appel au lexique, et l'introduction d'un hétérotype r avec $\mathfrak{a}(r) = \{g, s\}$ pour occuper la position ontologique des entrées qui auraient reçu le droit à la coercion partielle. La conclusion que nous tirons cette analyse est donc que le choix d'imposer des coercions dirigées par les types à l'aide d'hétérotypes, comme chez Pustejovsky, Asher et dans notre théorie, ou bien d'imposer des coercions partielles dirigées par les entrées lexicales comme Retoré et ses collègues, n'apporte aucune modification profonde à la théorie des types en elle-même. Il s'agit davantage d'un choix de design, pouvant avoir trait à des problématiques d'implémentation, que d'un réel choix de paradigme théorique : avec les règles de typage adaptées, les mêmes résultats pourront être obtenus dans l'un ou l'autre cas.

Terminons enfin par une note sur la relation entre projections hétérotypiques et sous-typage. L'exemple qui précède sous-entend que si U est le support du morphisme partiel $(m, f) : G \rightharpoonup S$, alors $m : U \rightarrowtail G$ est un monomorphisme, ce qui rendrait donc la relation entre les types sous-jacents éligible à la relation de sous-typage, et non simplement à une projection. La raison de ce phénomène tient aux propriétés de la relation interne induite par le morphisme partiel : par la nature même de la coercion comme une fonction partielle de G à S, la relation en question est *fonctionnelle* sur G. Prise en termes ensemblistes, cette observation signifie que pour toutes paires d'aspects $(x, y), (x', y') \in U$, l'égalité $x = x'$ implique l'égalité $y = y'$; autrement dit, chaque élément de G est en relation avec au plus un élément de S. Sémantiquement, ceci traduit l'idée que toute classe est associée à une unique salle dans laquelle elle se réunit. Et cette fonctionnalité des relations au sein des hétérotypes n'est d'ailleurs pas un cas isolé : pour *livre*, par exemple, on peut admettre que chaque délimitation physique d'un livre est associée à exactement un élément informationnel qui décrit son contenu, [110] alors qu'à l'inverse, un même livre informationnel peut être incarné dans plusieurs exemplaires physiquement distincts.

110. Les problèmes de comptage évoqués plus haut suggèrent que ce n'est pas tout à fait le cas, puisque si un livre rassemble plusieurs œuvres distinctes, on peut théoriquement mettre en relation l'entité physique avec les différentes œuvres séparément. Une possibilité pour résoudre ce problème sans s'affranchir des possibilités de comptage serait d'imposer sur l'objet des entités informationnelles une structure interne, de sorte à pouvoir

En termes ensemblistes toujours, on comprend alors qu'une relation $R \subseteq A \times B$ sera considérée comme fonctionnelle sur A si sa première projection est injective. Généralisée à toute catégorie et à toute arité, on pourra alors dire qu'une relation interne $R \rightarrowtail \prod_{i=1}^{n} A_i$ est fonctionnelle sur A_i si sa i^{e} projection $\pi_i : R \to A_i$ est un monomorphisme. Aussi, lorsqu'un hétérotype est caractérisée par une relation fonctionnelle, il est possible de le considérer comme un sous-type de l'aspect sur lequel cette relation est fonctionnelle. Ceci rejoint l'idée qu'il pourrait exister dans les hétérotypes un aspect *principal*, qui serait privilégié par rapport aux autres quitte à être considéré par défaut lorsque le contexte ne force pas de coercion particulière ; si de tels aspects existent, il y a de fortes chances qu'il s'agisse d'aspects pour lesquels la relation structurant l'hétérotype est fonctionnelle. Ceci pourra se révéler intéressant pour des cas d'ambiguïté impliquant des hétérotypes : en effet, bien que les aspects d'un tel type sont incomparables entre eux, ils disposent nécessairement de majorants communs, dont le plus grand type ontologique e. Alors, une phrase comme « ce livre existe » est ambiguë sur l'aspect de *livre* coercé, puisque les types des entités physiques et des entités informationnelles sont tous deux sous-types de e. Bien évidemment, le contexte pragmatique pourra servir à la résolution de cette ambiguïté comme pour d'autres, mais en l'absence de tout contexte, il peut être utile de savoir qu'un aspect principal peut être considéré ; et cette information, bien que relevant davantage du lexique à l'instar des qualia de Pustejovsky, a malgré tout une conséquence sur le modèle catégorique de notre théorie, tendant à flouter encore la frontière que nous avons tirée entre coercions de sous-typage et coercions hétérotypiques.

Il ne nous reste que peu à ajouter sur le sujet des hétérotypes sans devoir se pencher sur les nouvelles questions de la prochaine section. Toutefois, avant de nous y rendre, nous pensons qu'il pourrait être pertinent de donner un petit exemple concret de composition en CPTT enrichi d'hétérotypes.

Exemple 18. *La dérivation suivante illustre un exemple de composition minimaliste pour l'expression nominale « heavy boring book », qui représente une forme simple de coprédication jouant sur les deux aspects de book. Afin de construire cette dérivation, nous supposons disposer dans* $\mathbb{B}$ *de types ontologiques* $\mathtt{p}$ *et* $\mathtt{i}$ *représentant respectivement les entités physiques et informationnelles, et dans* $\mathbb{H}$ *d'un type* $\mathtt{ic}$ *(pour « information carrier ») tel que* $\mathfrak{a}(\mathtt{ic}) = \{\mathtt{p}, \mathtt{i}\}$*, destiné à représenter le type des entités structurées par une relation où l'aspect physique contient l'aspect informationnel. De plus, nous supposons que notre ensemble de constantes est composé de* $\mathbf{heavy}' : \mathcal{P}(\mathcal{P}\mathtt{p} \times \mathtt{p})$*, de* $\mathbf{boring}' : \mathcal{P}(\mathcal{P}\mathtt{i} \times \mathtt{i})$*, et de* $\mathbf{book} : \mathcal{P}(\mathtt{ic})$*.*[111] *Voici alors la dérivation obtenue :*

$$\dfrac{\dfrac{}{\vdash \mathbf{heavy}' : \mathcal{P}(\mathcal{P}\mathtt{p} \times \mathtt{p})} \quad \dfrac{\dfrac{}{\vdash \mathbf{boring}' : \mathcal{P}(\mathcal{P}\mathtt{i} \times \mathtt{i})} \quad \dfrac{}{\vdash \mathbf{book} : \mathcal{P}(\mathtt{ic})} \quad \dfrac{\mathtt{i} \in \mathfrak{a}(\mathtt{ic})}{\vdash_H p : \mathtt{ic} \to \mathtt{i}}}{\vdash (\mathcal{P}(\exists p \times p)\mathbf{boring}')\mathbf{book} : \mathcal{P}(\mathtt{ic})} \quad \dfrac{\mathtt{p} \in \mathfrak{a}(\mathtt{ic})}{\vdash_H p' : \mathtt{ic} \to \mathtt{p}}}{\vdash (\mathcal{P}(\exists p' \times p')\mathbf{heavy}')(\mathcal{P}(\exists p \times p)\mathbf{boring}')\mathbf{book} : \mathcal{P}(\mathtt{ic})}$$

Comme nous pouvons le constater, la présence des coercions est assez lourde dans le terme final, mais est néanmoins nécessaire ; cependant, en exploitant la règle de conversion pour les

considérer la somme méréologique des œuvres contenues dans l'exemplaire physique comme l'image de celui-ci par la relation fonctionnelle. Cette idée suppose bien entendu davantage de définitions et de formalismes que nous ne pouvons en fournir ici ; nous laissons cette question en suspens pour de futures recherches.

111. Le prime sur les constantes adjectivales a pour intention de souligner que ces dernières ne sont pas les représentations minimales habituelles, dont les types seraient plutôt $\mathcal{P}\mathtt{p}$ et $\mathcal{P}\mathtt{i}$ respectivement. Nous laissons aux lectrices et lecteurs le soin de reproduire la dérivation qui suit en remplaçant $\mathbf{heavy}'$ par le terme complexe $\lambda x.(\mathbf{heavy}(\pi_1 x) \wedge \big((\pi_2 x)(\pi_1 x)\big)$, et similairement pour $\mathbf{boring}'$.

projections inverses (qui est similaire à la règle (eqct) donné en section 6.2), il sera par la suite possible de se ramener à une expression plus simple lorsqu'un terme de type t sera obtenu à partir de celui-là.

6.4 Coercions libres et principes d'inférence

La section précédente nous a permis de compléter la construction de notre théorie CPTT par l'ajout d'un ensemble d'hétérotypes fixés, déterminés par les besoins du lexique. Rappelons en effet que comme posé en hypothèse en section 1.4, nous attendons du lexique qu'il fournisse les constantes typées correspondant à ses entrées, ainsi que les aspects et la relation qui structure ces derniers pour les hétérotypes. La détermination de l'ensemble des hétérotypes, tout comme celle des types ontologiques, est donc soumise à l'observation du langage et à la distinction des compositions qui sont considérées comme « normales », c.-à-d. qui donnent lieu à des significations cohérentes et explicables sans difficulté. Il est donc admis que le sous-typage et la polysémie hétérotypique entrent dans ce cas de figure. Mais dans ce second cas, la limite n'est pas nécessairement facile à déterminer. Ainsi, le mot *party* dans « *John left the party* » est considéré comme un évènement coercé vers le lieu où il se déroule, mais cette coercion relève-t-elle d'un cas d'hétérotypie ? La question reste ouverte, et il est probable que différents lexiques et différentes implémentations pourraient fournir des réponses opposées à cette question. Dans tous les cas, c'est la manière d'encoder les informations dans le lexique qui prime, tout comme pour les coercions partielles face aux hétérotypes : si le lexique prétend que *party* est hétérotypique, le système le gèrera en conséquence, et dans le cas contraire, il faudra reconnaître que quelque chose manque à notre système actuel.

En effet, en dehors des coercions prévues par le sous-typage et l'hétérotypie, CPTT ne permet pas à ce stade de traiter des coercions plus arbitraires ou plus libres, alors que cela est absolument nécessaire. Le cas de *party* en est une bonne illustration : s'il avérait par exemple que tous les évènements ont un lieu associé, alors la fonction associant chaque évènement à son lieu serait totale, et l'introduction d'un hétérotype pour cela n'aurait pas de sens ; un mécanisme marquant directement que la coercion a lieu sans avoir besoin d'introduire un hétérotype supplémentaire pour cela sera beaucoup plus léger et minimaliste — critères qui ont de bonnes raisons de nous intéresser. Mais d'autres exemples motivent également ce type d'opération, à l'instar de la conjonction d'adjectifs en vue d'une coprédication : dans l'exemple de la section précédente, nous avions appliqué *boring* et *heavy* à la suite l'un de l'autre, rendant la composition simple à effectuer. Mais comment peut-on dériver la sémantique de *boring and heavy*, comme il devrait se produire au cours de la dérivation pour la phrase « *the book is boring and heavy* » ? Il se trouve qu'à ce stade de notre développement, cette dérivation est impossible, à moins d'avoir pour entrée du mot *and* un terme de type $\mathcal{P}(\mathcal{P}(\mathtt{ic}) \times \mathcal{P}(\mathtt{ic}) \times \mathtt{ic})$, ce qui est non-souhaitable, car cela signifierai avoir d'autant d'entrées pour *and* qu'il y a de types ontologiques et d'hétérotypes possibles — allant donc à l'encontre de nos ambitions minimalistes.

Il nous manque donc des mécanismes combinatoires capables de former des coercions libres, et de préparer des coprédications même si l'argument hétérotypique n'a pas encore été fourni. L'objectif de la présente section est de proposer une légère reformulation de CPTT qui rende cela possible, et qui permette au passage de simplifier l'utilisation des coercions, dont nous avons constaté l'effet alourdissant sur les termes. Pour ce faire, nous proposons une modification dans l'expression des jugements de typage, afin de laisser plus d'espace à des constructions dynamiques de coercions, voire d'hétérotypes, qui utilise un système de variables de types de base et de contraintes. Considérons donc un ensemble $\mathcal{V}$ de *variables de types*. Pour construire

notre nouvelle représentation des types, nous rappellons la version annotée des types de $\mathbb{T}$ comme introduite informellement en section 6.2 : nous avions notamment supposé une annotation des types des constantes pour marquer si les composantes ontologiques étaient liées entre elles, annotation que nous avons d'ailleurs implicitement utilisée dans l'exemple à la fin de la section précédente. L'idée générale ici sera alors de former des nouveaux types en substituant une même variable de type pour toutes les occurrences d'un type ontologique d'annotation fixée, et d'étendre la méthode aux hétérotypes également. Mais l'information des types substitués n'est pas perdue pour autant : nous la conservons sous forme de contraintes dans un ensemble apparié à ses nouveaux types. Les définitions suivantes posent le cadre formel dans lequel cette nouvelle vision des types s'incrit.

Définition 45. *Une* contrainte ontologique *est une paire d'élements dans* $\mathcal{V} \cup \mathbb{B} \cup \mathbb{H}$.

Lorsque x et y sont deux éléments de l'ensemble susmentionné, la contrainte ontologique qu'ils forment sera notée $x \sqsubseteq y$ plutôt que (x, y). Cette première définition permet de donner la suivante, qui redéfinit notre conception des types sémantiques :

Définition 46. *Un* type contraint *est une paire* (σ, C) *où* σ *est un élément de l'ensemble des types ouverts* $\mathbb{T}'$ *défini par la grammaire*

$$\mathbb{T}' ::= 1 \mid t \mid \mathcal{V} \mid \mathbb{T}' \times \mathbb{T}' \mid \mathcal{P}\mathbb{T}'$$

et C *est un ensemble de contraintes ontologiques tel que pour tout* $\alpha \in \mathcal{V}$ *ayant au moins une occurrence dans* σ*, il existe une contrainte ontologique dans* C *de la forme* $\alpha \sqsubseteq a$ *avec* $a \in \mathbb{B} \cup \mathbb{H}$.

L'intuition derrière cette nouvelle conception est assez simple : dans la continuité des principes formulés au chapitre 3, les types contraints séparent un peu plus la composante structurelle, qui est donnée par le type ouvert, de la composante ontologique, qui est représentée par l'ensemble des contraintes. La condition d'avoir des contraintes où les variables de types sont à gauche et des types de bases à droite permet d'exprimer les critères ontologiques de façon plus lâche que pour les types montagoviens habituels ou pour les types dans $\mathbb{T}$. Cette construction présente évidemment certaines similitudes avec les types paramétrés et le polymorphisme borné [26], à ceci près qu'ici les variables de types n'ont pas pour vocation d'être instanciées au moment de l'application d'un terme à un autre, mais à la fin de la dérivation ; c'est pourquoi nous conservons les contraintes dans un ensemble séparé. On notera aussi des similitudes de notre approche par contraintes avec celle de Pottier [134], dont elle s'inspire partiellement ; notre conception des types et des contraintes, du fait de l'abandon du type flèche, diffère cependant significativement de ce travail.

La manipulation de ces types implique également une relecture de la notion de bon typage : pour être bien typé par un type contraint, il faudra non seulement s'assurer que le terme a une structure de type légitime, mais également que l'ensemble des contraintes ne contient pas d'incohérence. Pour la suite de la présente section, nous définissons une extension de l'ordre $\leq$ dans $\mathbb{B}$ de la manière suivante : notons $\mathcal{P}(\mathbb{B})^{\square}$ l'ensemble des parties de $\mathbb{B}$ dont les éléments sont deux à deux incomparables ; alors, pour tous $U, V \in \mathcal{P}(\mathbb{B})^{\square}$, on notera $U \Rrightarrow V$ s'il existe une fonction $f : V \to U$ injective telle que pour tout $a \in V$ on ait $f(a) \leq a$. Remarquons qu'il n'est pas demandé que la fonction injective soit unique : des ambiguïtés peuvent exister si un élément de V admet plus d'un antécédent possible. Nous pensons cependant que ceci n'est pas dommageable et au contraire essentiel à notre objectif linguistique, où les ambiguïtés sont courantes.

Définition 47. *Une contrainte ontologique* $x \sqsubseteq y$ *est* correcte *si l'une des conditions suivante est remplie :*

— *au moins d'un des deux éléments x ou y est une variable de $\mathcal{V}$;*

— $x, y \in \mathbb{B}$ *et* $x \leq y$ *;*

— $x \in \mathbb{H}$, $y \in \mathbb{B}$ *et il existe* $z \in \mathfrak{a}(x)$ *tel que* $z \leq y$ *;*

— $x, y \in \mathbb{H}$ *et* $\mathfrak{a}(x) \Rrightarrow \mathfrak{a}(y)$.

Un ensemble de contraintes ontologiques définit intuitivement une relation sur $\mathcal{V} \cup \mathbb{B} \cup \mathbb{H}$. Si C est un tel ensemble, on notera alors C^* la fermeture réflexive et transitive de C.

Définition 48. *Un ensemble C de contraintes ontologiques est* cohérent *si tous les élements de C^* sont corrects.*

Contrairement à la version de CPTT établie dans les sections précédentes, nous allons nous permettre ici d'alléger les critères qui fondent la sûreté du typage : en effet, nous cherchons désormais un mécanisme capable de révéler des usages créatifs de la langue, ce qui implique d'être capable de construire des termes qu'une théorie des types plus classique serait normalement incapable de construire ou de typer correctement, à la condition qu'un élément dans le jugement de typage associé conserve une information explicite indiquant qu'un usage créatif a eu lieu. La stratégie développée autour des types contraints est de construire des termes où les variables caractérisent des coercions : il s'agit alors de déterminer s'il est possible de remplacer ces différentes variables par des types ontologiques ou des hétérotypes connus, et d'instancier les coercions conséquemment. Un usage créatif sera alors détecté si la seule manière d'instancier une variable tout en préservant la cohérence des contraintes serait d'étendre une entité de type ontologique simple vers un hétérotype connu, voire de créer un hétérotype qui n'existe pas : la coercion qui en résulte est alors appelée *coercion libre*, car elle implique qu'une entité d'un type déterminé a dû être réinterprété comme un composé hétérotypique éventuellement ad hoc pour pouvoir être passé en argument d'un prédicat. Tout ceci indique évidemment que nous considérons les ensembles de contraintes comme des équations à résoudre ; la définition suivante formalise un peu plus cette idée.

Définition 49. *Soit C un ensemble de contraintes ontologiques cohérent. Une* solution *de C est une fonction partielle $\gamma : \mathcal{V} \rightharpoonup \mathcal{P}(\mathbb{B})$, définie sur toutes les variables qui apparaissent dans C, qui satisfait les conditions suivantes :*

— *pour tout α, les éléments de $\gamma(\alpha)$ sont incomparables (c.-à-d. que $\gamma(\alpha) \in \mathcal{P}(\mathbb{B})^{\square}$) ;*

— *pour tout $\alpha \sqsubseteq a$ dans C avec $a \in \mathbb{B}$, il existe $b \in \gamma(\alpha)$ tel que $b \leq a$;*

— *pour tout $\alpha \sqsubseteq r$ dans C avec $r \in \mathbb{H}$, $\gamma(\alpha) \Rrightarrow \mathfrak{a}(r)$;*

— *pour tout $a \sqsubseteq \beta$ dans C avec $a \in \mathbb{B}$, il existe $b \in \mathbb{B}$ tel que $\gamma(\beta) = \{b\}$ et $a \leq b$;*

— *pour tout $r \sqsubseteq \beta$ dans C avec $a \in \mathbb{B}$, $\mathfrak{a}(r) \Rrightarrow \gamma(\beta)$;*

— *pour tout $\alpha \sqsubseteq \beta$ dans C, $\gamma(\alpha) \Rrightarrow \gamma(\beta)$.*

Le choix des sous-ensembles de $\mathbb{B}$ aux éléments incomparables comme codomaine des solutions a pour but de permettre la représentation des hétérotypes : un tel hétérotype correspondra ainsi à l'ensemble de ses aspects. Les types ontologiques seront quant à eux représentés par des singletons. Naturellement, cette définition de solution à un tel système de contrainte n'est vraiment intéressante que si nous disposons de moyens pour calculer une telle solution. Fort heureusement, il se trouve que la notion même de cohérence sur les ensembles de contraintes ontologiques rassemble un nombre suffisant de bonnes propriétés pour nous permettre de produire des résultats qui vont dans ce sens, à l'instar du théorème suivant :

Algorithme : up_unify

Données : Deux éléments A et B dans $\mathcal{P}(\mathbb{B})^{\square}$.
Résultat : La borne supérieure de A et B pour $\Rrightarrow$.
si $|A| \leq |B|$ alors :
  $S \leftarrow \varnothing$
  $T \leftarrow \varnothing$
  pour tout $a \in A$: $T \leftarrow T \cup \{\mathsf{min}(\{a \cup b \mid b \in B\})\}$
  pour tout $x \in T$: si $\exists y \in T.\, y < x$ alors skip sinon $S \leftarrow S \cup \{x\}$
sinon $S \leftarrow \mathsf{up\_unify}(B, A)$
retourner S

FIGURE 6.1 – Définition de up_unify

Algorithme : down_unify

Données : Deux éléments A et B dans $\mathcal{P}(\mathbb{B})^{\square}$.
Résultat : La borne inférieure de A et B pour $\Rrightarrow$.
$S \leftarrow \varnothing$
$M \leftarrow A \cup B$
pour tout $x \in M$: si $\exists y \in M.\, y < x$ alors skip sinon $S \leftarrow S \cup \{x\}$
retourner S

FIGURE 6.2 – Définition de down_unify

Théorème 3. *Tout ensemble de contraintes ontologiques cohérent admet au moins une solution calculable.*

Démonstration. Pour mener à bien cette démonstration, nous proposons un algorithme qui permet de construire une solution à un ensemble de contrainte, et produit une erreur lorsqu'il n'en est pas capable. Il suffira alors de vérifier que les cas où l'algorithme ne peut pas calculer une solution correspondent aux cas où la l'ensemble de contraintes initial n'est pas cohérent. Soit donc C un ensemble de contraintes ontologiques quelconques. L'idée de l'algorithme que nous allons décrire ci-dessous est de maintenir pour chaque variable de $\mathcal{V}$ apparaissant dans C un ensemble de ses majorants et un ensemble de ses minorants, que nous représenterons respectivement par deux fonctions partielles R et $I : \mathcal{V} \rightharpoonup \mathcal{P}(\mathbb{B})$. Les singletons représenteront les types ontologiques dans $\mathbb{B}$, et les ensembles de cardinal 2 ou plus correspondront aux hétérotypes. Initialement, pour toute variable α apparaissant dans C, nous posons $I(\alpha) = R(\alpha) = \varnothing$.

Pour mettre à jour ces fonctions, nous utiliserons deux algorithmes auxiliaires d'unification notés up_unify et down_unify, qui calculent respectivement les bornes supérieure et inférieure de deux éléments de $\mathcal{P}(\mathbb{B})$ pour l'ordre $\Rrightarrow$. Les définitions de ces deux algorithmes en pseudo-code sont données ci-dessus en figures 6.1 et 6.2. Ces définitions utilisent des propriétés inhérentes à la structure d'arbre de $\mathbb{B}$, et l'opération $\cup$ doit être comprise comme une opération sur $\mathbb{B}$ étendu d'un plus petit élément 0. Le reste de l'algorithme se base sur le traitement des contraintes de C selon un ordre précis : on traitera d'abord les contraintes de minoration des variables, puis celles de majoration, avant de traiter celles portant uniquement sur des variables.

On commence donc par les contraintes de C de la forme $x \sqsubseteq \alpha$ avec $x \in \mathbb{B} \cup \mathbb{H}$ et $\alpha \in \mathcal{V}$. Si $I(\alpha) = \varnothing$, aucune contrainte sur α n'a été rencontrée, et on peut assigner à $I(\alpha)$ la valeur $\{x\}$ si $x \in \mathbb{B}$, ou la valeur $\mathfrak{a}(x)$ si $x \in \mathbb{H}$. Dans le cas contraire où $I(\alpha) \neq \varnothing$, une certaine forme d'unification est nécessaire : on appliquera $I(\alpha) \leftarrow \mathsf{up\_unify}(\{x\}, I(\alpha))$ si $x \in \mathbb{B}$, et $I(\alpha) \leftarrow \mathsf{up\_unify}(\mathfrak{a}(x), I(\alpha))$ sinon.

La seconde étape est de traiter les contraintes de la forme $\alpha \sqsubseteq x$, en tenant compte des contraintes de minoration qui doivent également être satisfaites. Avant de décrire en détail les différents cas de figure, il est nécessaire de souligner une propriété essentielle de $\mathsf{down\_unify}$ qui simplifiera grandement nos opérations : pour tout $S \in \mathcal{P}(\mathbb{B})$, $\mathsf{down\_unify}(S, \varnothing) = S$.[112] L'application de cette propriété aux valeurs initiales de la fonction R permettent de regrouper toutes les mises à jour des valeurs de celle-ci. Le cas de base est alors celui d'une contrainte $\alpha \sqsubseteq x$ où $I(\alpha) = \varnothing$: il suffit de remplacer la valeur précédente de $R(\alpha)$, quelle qu'elle soit, par $\mathsf{down\_unify}(R(\alpha), f(x))$, où $f(x)$ est $\{x\}$ si $x \in \mathbb{B}$ et $\mathfrak{a}(x)$ si $x \in \mathbb{H}$. Si en revanche $I(\alpha) \neq \varnothing$, un contrôle supplémentaire est nécessaire. Dans le cas où $x \in \mathbb{B}$, il faut tester s'il existe $a \in I(\alpha)$ tel que $a \leq x$: si c'est bien le cas, on applique $R(\alpha) \leftarrow \mathsf{down\_unify}(R(\alpha), \{x\})$, et sinon, on renvoie une erreur puisqu'aucune solution ne peut être trouvée. On notera alors que $I(\alpha)$ non-vide implique l'existence d'au moins un type y dans $\mathbb{B} \cup \mathbb{H}$ tel que $y \sqsubseteq \alpha$ est dans C, auquel cas $y \sqsubseteq x$ est dans C^* : si la condition précédente n'est pas remplie, C n'est alors pas cohérent. Dans l'autre cas où $x \in \mathbb{H}$, le principe est le même : on vérifie que $|\mathfrak{a}(x)| \leq |I(\alpha)|$ et que pour tout $b \in \mathfrak{a}(x)$, il existe $a \in I(\alpha)$ tel que $a \leq b$, auquel cas on applique $R(\alpha) \leftarrow \mathsf{down\_unify}(R(\alpha), \mathfrak{a}(x))$; dans le cas contraire, on renvoie là encore une erreur, et on vérifie comme précédemment que ce cas de figure implique l'incohérence de C.

La troisième étape est le traitement des contraintes de la forme $\alpha \sqsubseteq \beta$ avec $\alpha, \beta \in \mathcal{V}$. Pour ce faire, il est nécessaire de s'occuper en priorité des éventuelles contraintes cycliques de la forme $\alpha_1 \sqsubseteq \alpha_2, \ldots, \alpha_n \sqsubseteq \alpha_1$, qui doivent donner lieu à des égalités. Il faut alors calculer les bornes supérieures et inférieures de toutes contraintes déjà connues, et s'assurer qu'elles sont compatibles. On initialise pour cela deux variables S et T respectivement à $I(\alpha_{i_0})$ où $i_0 = \mathsf{min}(\{i \in [\![1, n]\!] \mid I(\alpha_i) \neq \varnothing\})$ et à $\varnothing$, et pour tout $i \in [\![1, n]\!]$, on effectue $S \leftarrow \mathsf{up\_unify}(S, I(\alpha_i))$ si $I(\alpha_i) \neq \varnothing$ ainsi que $T \leftarrow \mathsf{down\_unify}(T, R(\alpha_i))$. Il faut ensuite vérifier que $|T| \leq |S|$ et que pour tout $b \in T$, il existe $a \in S$ tel que $a \leq b$: si c'est le cas, on applique $I(\alpha_i) \leftarrow S$ et $R(\alpha_i) \leftarrow T$ pour tout $i \in [\![1, n]\!]$, et sinon on renvoie une erreur, qui démontre une fois de plus l'existence d'une série de contraintes dont l'extension transitive aboutit à une contrainte incorrecte, rendant C à nouveau incohérent. Enfin, il ne reste plus qu'à s'occuper des contraintes à deux variables qui ne font pas partie d'un cycle. Si $\alpha \sqsubseteq \beta$ est dans C, on définit deux variables S et T de la façon suivante : si $I(\alpha)$ et $I(\beta)$ sont non-vides, alors $S \leftarrow \mathsf{up\_unify}(I(\alpha), I(\beta))$; sinon si seul $I(\alpha)$ est non-vide alors $S \leftarrow I(\alpha)$, sinon $S \leftarrow I(\beta)$; quant à T, on l'obtient par $T \leftarrow \mathsf{down\_unify}(R(\alpha), R(\beta))$. Alors, si S est non-vide et pour tout $b \in R(\beta)$, il existe $a \in S$ tel que $a \leq b$, ou si S est vide, on applique $I(\beta) \leftarrow S$; sinon on renvoie une erreur signifiant que les minorants de α et les majorants de β ne sont pas compatibles, ce qui est un motif d'incohérence de C une fois de plus. De même, si $I(\alpha)$ est non-vide on vérifie que pour tout $b \in T$ il existe $a \in I(\alpha)$ tel que $a \leq b$, auquel cas on pose $R(\alpha) \leftarrow T$, et sinon on renvoie une nouvelle erreur pour les mêmes raison que précédemment.

Une fois cet algorithme appliqué sans erreur, nous obtenons une solution γ de C par le procédé suivant : pour tout α dans C, si $R(\alpha) \neq \varnothing$ alors $\gamma(\alpha) = R(\alpha)$, et dans le cas contraire, $\gamma(\alpha) = I(\alpha)$. Notons que les deux valeurs de I et R sur une même variable ne peuvent pas être vides en même temps. Par construction, cette fonction γ remplit bien les propriétés d'une solution telle que donnée dans la définition précédente. De plus, nous avons décrit un algorithme capable de construire explicitement cette solution, et qui ne peut échouer que si C n'est pas cohérent ; par contraposition, nous avons donc bien établi notre théorème. □

112. Cette propriété est cohérente avec la conception de $\mathsf{down\_unify}$ comme opérateur de borne inférieure sur $\Rightarrow$, ordre pour lequel $\varnothing$ est le plus grand élément. De la même façon, on a logiquement $\mathsf{up\_unify}(\varnothing, S) = \varnothing$ pour tout $S \in \mathcal{P}(\mathbb{B})$.

Pour les fins de notre théorie de types, il sera néanmoins pratique de fixer une forme particulière d'ensemble de contraintes, qui aura notamment l'avantage de rendre le calcul d'une solution encore plus facile si nous parvenons à maintenir cette propriété tout au long de nos dérivations :

Définition 50. *Un ensemble de contraintes ontologiques C est* normalisé *si toutes les contraintes de C sont de la forme $\alpha \sqsubseteq x$ avec $\alpha \in \mathcal{V}$ et x quelconque, et si pour toute variable α apparaissant dans C il existe au moins une contrainte de la forme $\alpha \sqsubseteq y$ avec $y \in \mathbb{B} \cup \mathbb{H}$.*

Notons que cette propriété de normalisation n'est pas préservée par un passage à la fermeture transitive. L'idée derrière cette définition est d'imposer une propriété qui assure que les types contraints que nous manipulons expriment bien des critères ontologiques : chaque variable représente une composante ontologique à instancier, et les bornes supérieures caractérisent les critères correspondants. Pour pouvoir utiliser cette propriété en pratique dans la construction de termes, il va donc être nécessaire de l'imposer sur les fondations même du typage, c.-à-d. pour les variables et les constantes. Un *contexte contraint* sera donc défini comme une liste de paires de la forme $x : (d[\alpha], \{\alpha \sqsubseteq a\})$, avec $x \in V$, $\alpha \in \mathcal{V}$ et $a \in \mathbb{B} \cup \mathbb{H}$.[113] De même, la signature de constantes $(Q, \mathfrak{t})$ de notre système est redéfinie de telle sorte que la fonction de typage $\mathfrak{t}$ associe à chaque constante un ensemble de contraintes normalisé portant exactement sur les variables du type ouvert ; autrement dit, on impose que pour $\mathfrak{t}(q) = (\sigma, C)$ on a $\alpha \sqsubseteq a$ dans C si et seulement si α a au moins une occurrence dans σ et $a \in \mathbb{B} \cup \mathbb{H}$.

Mais les spécificités de la théorie de types que nous envisageons de construire ne s'arrêtent pas là. Comme la construction des termes repose sur la résolution de contraintes ontologiques, il est impossible de déterminer les coercions nécessaires au moment de la composition : nous sommes obligés d'attendre la fin de la dérivation, une fois qu'une solution aux contraintes a été calculée, pour pouvoir finaliser la construction de ces coercions. En effet, si en termes de structure de types les coercions sont faciles à inférer (il suffira pour cela de remonter les règles données en section 6.2 pour le sous-typage et en section 6.3 pour les hétérotypes, sachant qu'une combinaison des deux pourra parfois être nécessaire), rien ne pourra être finalisé tant que les coercions ontologiques de base sont indeterminées, ce que seule la résolution des contraintes permettra. Par conséquent, une dérivation dans une théorie de types avec de telles contraintes doit introduire des symboles de coercions encore non calculées, et les accumuler en mémoire jusqu'à pouvoir les traiter à la toute fin du processus. Pour ce faire, nous introduisons un ensemble G de *variables de coercions*, qu'on ajoutera à Z de telle sorte que l'apparition de ces variables dans les termes soit permise. Un *jugement de typage contraint* est alors un jugement de la forme $\Gamma; \Delta \vdash u : (\sigma, C)$ où Γ est un contexte contraint ; Δ est un contexte de coercions, c.-à-d. une liste d'éléments de la forme $c : d[\alpha] \to d[\beta]$ avec d un contexte structurel de $\mathbb{T}'$, $\alpha, \beta \in \mathcal{V}$, et $c \in G$; u est un terme dans Λ et (σ, C) est un type contraint.

Nous donnons alors l'intégralité des règles de notre théorie de types avec contraintes en figure 6.3. Les coercions sont une fois de plus introduites par les règles d'applications totale et partielle ; on observe que leur construction assure effectivement que chaque coercion obéisse à une loi d'équivalence structurelle entre son domaine et son codomaine : seules les composantes ontologiques sont potentiellement différentes. Ces mêmes règles sont également les seules à enrichir l'ensemble des contraintes avec de nouvelles contraintes, à travers l'opération $\mathsf{insert}(C, \alpha, \beta)$ qui a deux effets : ajouter à C la contrainte $\alpha \sqsubseteq \beta$, et pour toute contrainte de C de la forme $\delta \sqsubseteq \beta$, ajouter à C la contrainte $\delta \sqsubseteq \alpha$. Cette définition répond à la logique suivante : pour chaque application, la variable de type de l'argument est conçue comme plus petite que celle du

113. Une extension à plusieurs variables parallèles est bien évidemment possible.

$$\frac{}{\Gamma, x : (\sigma, C), \Gamma'; \varnothing \vdash x : (\sigma, C)}\,(\text{ax}) \quad \frac{q \in Q}{\Gamma; \varnothing \vdash q : \mathfrak{t}(q)}\,(\text{const}) \quad \frac{}{\Gamma; \varnothing \vdash * : (1, \varnothing)}\,(\text{unit})$$

$$\frac{\Gamma; \Delta \vdash u : (\sigma, C) \qquad \Gamma; \Delta' \vdash v : (\tau, C')}{\Gamma; \Delta \cup \Delta' \vdash \langle u, v\rangle : (\sigma \times \tau, C \cup C')}\,(\text{pair}) \quad \frac{\Gamma; \Delta \vdash u : (\sigma_1 \times \sigma_2, C)}{\Gamma; \Delta \vdash \pi_i u : (\sigma_i, C)}\,(\text{proj})$$

$$\frac{\Gamma, x : (\sigma, C); \Delta \vdash u : (t, C')}{\Gamma; \Delta \vdash \lambda x.u : (\mathcal{P}\sigma, C \cup C')}\,(\text{lam}_1) \quad \frac{\Gamma, x : (\sigma, C); \Delta \vdash u : (\mathcal{P}\tau, C')}{\Gamma; \Delta \vdash \lambda y.(u(\pi_2 y))[\pi_1 y/x] : (\mathcal{P}(\sigma \times \tau), C \cup C')}\,(\text{lam}_2)$$

$$\frac{\Gamma; \Delta \vdash u : (\mathcal{P}(\prod_{k=1}^n d_k[\beta]), C) \qquad \Gamma; \Delta' \vdash v : (\prod_{k=1}^n d_k[\alpha], C')}{\Gamma; \Delta \cup \Delta' \cup \{c : \prod_{k=1}^n d_k[\alpha] \to \prod_{k=1}^n d_k[\beta]\} \vdash u(cv) : (t, \mathsf{insert}(C \cup C', \alpha, \beta))}\,(\text{app}_t)$$

$$\frac{\Gamma; \Delta \vdash u : (\mathcal{P}(\prod_{k=1}^n d_k[\beta]), C) \qquad \Gamma; \Delta' \vdash v : (\prod_{k=1}^i d_k[\alpha], C')}{\Gamma; \Delta \cup \Delta' \cup \{c : \mathcal{P}(\prod_{k=1}^n d_k[\beta]) \to \mathcal{P}(\prod_{k=1}^n d_k[\alpha])\} \vdash (cu)v : (\mathcal{P}(\prod_{k=i+1}^n d_k[\alpha]), \mathsf{insert}(C \cup C', \alpha, \beta))}\,(\text{app}_p)$$

FIGURE 6.3 – Règles de la théorie des types avec contraintes

prédicat qui lui est appliqué,[114] de sorte qu'une contrainte $\alpha \sqsubseteq \beta$ dans C indique qu'un sous-terme est une application d'un prédicat de variable β à un argument de variable α. Lorsqu'une nouvelle application est formée, il faut alors prendre en compte la possibilité que l'argument vienne instancier des positions de prédicats, initialement occupées par la variable abstraite, auquel cas les variables de type des arguments à ces positions doivent être ajustées par rapport à la nouvelle variable introduite : si α vient remplacer un argument intial β, tous les minorants de β doivent alors devenir des minorants de α, car ces minorants proviennent d'arguments dans les positions de prédicats que α va occuper. Cette insertion de α dans les minorations de β ne nécessite cependant pas d'ajustement supplémentaire au niveau des coercions, car par compositions des coercions la nouvelle coercion de β vers α s'occupe automatiquement de connecter les prédicats et les arguments. Par ailleurs, notons que ces règles sont aisément adaptables à des cas d'application avec de multiples variables en parallèle, il suffit alors d'introduire autant de contraintes associées. Enfin, considérant la définition de la fonction insert ainsi que les hypothèses sur les contextes et sur les types des constantes, nous sommes en mesure d'établir la propriété suivante, dont la preuve est immédiate par induction sur les règles ci-dessus :

Proposition 41. *Si le jugement $\Gamma; \Delta \vdash u : (\sigma, C)$ est dérivable, alors C est normalisé.*

Une autre règle qui requiert quelques commentaires est celle des projections : en effet, on constate aisément que cette dernière conserve les informations des deux composantes de la paire initiale, que ce soit en termes de contraintes ontologiques comme de variables coercions, même si les éléments ayant trait à la composante de la paire éliminée ne sont plus censés être utiles. La raison à cela a trait aux unions ensemblistes qui apparaîssent dans la règle (pair) : une fois ces opérations effectuées, il n'est théoriquement plus possible de retrouver de quelle composante

114. On notera cependant que le phénomène inhérent au système montagovien faisant qu'un prédicat issu d'un syntagme nominal a parfois un type plus petit que celui de son argument semble échapper à cette logique. Néanmoins, les contraintes ajoutées par (app_t) ne s'opposent pas à ce cas de figure, le seul risque étant d'obtenir un abaissement tacite du type de l'argument, abaissement qui pourra être identifié après l'obtention d'une solution comme dans l'exemple 19 plus bas. Par conséquent, nous pouvons considérer que (app_t) capture à la fois les règles (act) et (abais) de la section 6.2.

est issue quelle contrainte. Il serait au moins envisageable de « nettoyer » ces ensembles un minimum, par exemple en retirant de Δ toute référence à une variable de coercion qui n'apparaîtrait pas dans $\pi_i u$, et en retirant de même de C toute référence à une variable de types qui n'apparaîtrait pas dans σ_i ; cependant, cette approche ne pourrait pas repérer des contraintes issues de la composante éliminée mais portant sur des variables communes aux deux types de la paire, parce qu'issus du même contexte Γ notamment. Pour compenser, nous disposons naturellement des règles structurelles qui permettent de gérer ces variations contextuelles. Quoi qu'il en soit, nous pensons que cet aspect n'est pas dommageable pour nos objectifs, et est même dans l'esprit de notre vision des types sémantiques : en effet, chaque application observée apporte sa contrainte sur le typage de la même manière que l'observation d'une relation de sens indique une compatibilité ontologique dans la théorie de Sommers (cf. section 2.5), et même si l'application en question n'apparaît plus dans le terme final, le simple fait qu'elle ait eu lieu au cours de la dérivation suffit à la considérer comme observée, et ses conséquences doivent donc être prises en compte.

Une autre difficulté de cette théorie est l'élaboration des règles de conversion équationnelles, car le typage varie avec les contraintes ontologiques : ainsi, si $\mathbf{f}$ est une constante de type $(\mathcal{P}\alpha, \{\alpha \sqsubseteq a\})$ et que $x : (\beta, \{\beta \sqsubseteq a\})$ est une variable, alors les règles d'application totale et d'abstraction permettent de dériver le jugement

$$\varnothing; c : \alpha \to \beta \vdash \lambda x.\mathbf{f}(cx) : (\mathcal{P}\beta, \{\alpha \sqsubseteq a, \beta \sqsubseteq a, \beta \sqsubseteq \alpha\})$$

dont les contraintes ne correspondent pas vraiment à celle de $\mathbf{f}$, rendant difficile la définition de l'η-expansion dans ce système. Mais il faut en fait bien voir que ce problème apparent n'en est pas vraiment un si l'on accepte de voir la théorie proposée ici non pas comme une théorie complète, mais comme un processus transitoire dont l'objectif est de nous ramener à CPTT : s'il n'est pas possible de donner le même type contraint à des termes censés être équivalents par conversion, il est possible de le faire sur les termes obtenus lorsque la solution aux contraintes est calculée et ses effets appliqués. Ainsi, une solution au jugement ci-dessus est donné par γ tel que $\gamma(\alpha) = \gamma(\beta) = \{a\}$. La propagation de cette solution à l'ensemble du jugement mène alors à la coercion $c = \mathrm{id}_a$ qui peut être effacée, et au jugement final $\vdash \lambda x.\mathbf{f}x : \mathcal{P}a$, qui a le même type que la constante $\mathbf{f}$ elle-même sous la même propagation ; l'η-conversion peut alors être retrouvée.

Comme affirmé par le théorème 3 ci-dessus, l'ensemble de contraintes issu d'un jugement dérivable admet toujours une solution calculable, et il est même possible de préciser certaines propriétés supplémentaires pour ces solutions du fait que les ensembles sont normalisés, dont la proposition 42 ci-dessous qui aura l'intérêt de fixer un cas limite qui répond le mieux à nos attentes. En préambule, nous posons la définition suivante :

Définition 51. *Soit C un ensemble de contraintes ontologiques. Une solution γ de C est* maximale *si pour toute solution γ' de C et pour toute variable α dans C, on a $\gamma'(\alpha) \Rrightarrow \gamma(\alpha)$.*

Les solutions maximales nous intéressent ici pour leur capacité à s'approcher au maximum des types ontologiques qu'on fixerait dans une dérivation de CPTT, du fait que les ensembles normalisés expriment uniquement des majorations des variables. Cette propriété est d'autant plus intéressante que nous disposons comme annoncé du résultat suivant :

Proposition 42. *Tout ensemble de contraintes normalisé admet une solution maximale.*

Démonstration. Pour établir cette propriété, il suffit de reprendre la démonstration du théorème 3, à ceci près que nous n'avons plus besoin de nous soucier des minorants qui sont exclus

des ensembles de contraintes normalisés. Nous n'avons donc plus qu'à maintenir la fonction partielle R des majorants de variables, qui par construction contient toujours la borne inférieure des majorants traités pour la variable considérée : par conséquent, la solution qui résulte de ce cas particulier de l'algorithme génère bien une solution maximale. □

Nous allons par conséquent privilégier la manipulation des solutions maximales. Il nous reste donc à définir plus précisément le processus de propagation d'une solution. Supposons qu'une dérivation a abouti à un ensemble de contraintes ontologiques C de solution γ ; notre but ici est alors de construire une substitution des variables de types de C par des types ontologiques ou hétérotypiques fixés. Si α est une variable dans C, sa valeur par la substitution est évidemment influencée par son image par γ. Si cette dernière est un singleton $\gamma(\alpha) = \{a\}$, on substituera simplement α par a. Sinon, deux cas de figure se posent : dans le premier, il existe $r \in \mathbb{H}$ tel que $\mathfrak{a}(r) = \gamma(\alpha)$, auquel cas la substitution remplacera α par r. Notons alors que comme nous n'écartons pas explicitement la possibilité que plusieurs symboles de $\mathbb{H}$ aient la même image par $\mathfrak{a}$ (dans le cas où plusieurs relations distinctes entre aspects de mêmes types existeraient), des ambiguïtés peuvent exister : il appartiendra aux étapes ultérieures de l'analyse sémantique de les résoudre. Mais si aucun antécédent dans $\mathbb{H}$ n'est trouvé, nous nous retrouvons face à un cas d'usage créatif de la langue : il est alors nécessaire d'introduire un hétérotype ad hoc pour le marquer, noté par exemple comme l'ensemble de ses aspects, éventuellement indexé si plusieurs occurrences pour des variables distinctes apparaissent. Cet hétérotype indique l'existence d'une relation inconnue entre différents aspects incomparables, qui devra là encore être déterminée par des analyses sémantiques plus complètes. Mais le fait que ces hétérotypes ad hoc soient observés au terme d'une dérivation doit être pris comme l'indication qu'un tel usage créatif a eu lieu : en cela, le rôle de la théorie des types est de repérer ces compositions non standard et de les marquer pour qu'elles puissent être prises en compte et expliquées par d'autres analyses, au lieu de rejeter en bloc la dérivation pour des questions d'erreur de typage. Notre système d'inférence par contraintes offre donc une plus grande flexibilité dans la composition, tout en distinguant par des éléments visibles les dérivations qui sont naturellement correctes de celles qui présentent des comportements inattendus.

Et comme sous-entendu plus haut, la substitution ainsi obtenue se propage au calcul des coercions, de sorte qu'une seconde substitution portant sur les variables de coercions peut être déduite de la première. À la différence de CPTT, il est possible comme dans l'exemple ci-dessus qu'une coercion ait lieu entre deux types absolument identiques, suggérant une coercion identité. lorsque c'est le cas, la variable correspondante sera simplement effacée du terme final, comme résultat d'une application non coercive. Naturellement, des hétérotypes ad hoc impliqueront la création de coercions projectives ad hoc. Le résultat le plus important de cette section est alors de comparer les dérivations dans le système contraint défini ici et CPTT tel qu'introduit au cours des sections précédentes. Étant donné une solution γ, on notera $\bar{\gamma} : \mathbb{T}' \longrightarrow \mathbb{T}$ la substitution associée, définie uniquement sur les types dont les variables sont couvertes par γ. Cette substitution s'étend naturellement en une fonction $\rho_{\bar{\gamma}}$ qui convertit un contexte contraint en un contexte classique, définie inductivement par les formules ci-dessous :

$$\rho_{\bar{\gamma}}(\varnothing) = \varnothing \qquad \rho_{\bar{\gamma}}(x : (\sigma, C), \Gamma) = x : \bar{\gamma}(\sigma), \rho_{\bar{\gamma}}(\Gamma)$$

De même, $\bar{\gamma}$ s'étend également en une substitution des variables de coercion vers leur coercion finale, laquelle permet de construire une fonction inductive sur Λ remplaçant les variables par leurs coercions associées, excepté les variables associées à l'identité qui sont simplement supprimées. Enfin, comme notre système est bien plus permissif que CPTT, nous allons à la place considérer une théorie « libre » CPTT$'(\bar{\gamma})$ qui est similaire à CPTT sauf pour les hétérotypes qui

sont librement engendrés, c.-à-d. que $\mathbb{H} \cong \mathcal{P}(\mathbb{B})^{\square}$, et pour la fonction de typage des constantes $\mathfrak{t}'$ qui est définie par $\mathfrak{t}' = \rho_{\bar{\gamma}} \circ \mathfrak{t}$, avec $\mathfrak{t}$ la fonction correspondante dans notre théorie contrainte. Nous pouvons alors établir le résultat qui suit :

Théorème 4. *Si le jugement* $\Gamma; \Delta \vdash u : (\sigma, C)$ *est dérivable et que* γ *est la solution maximale de* C*, alors le jugement* $\rho_{\bar{\gamma}}(\Gamma) \vdash \psi_{\bar{\gamma}}(u) : \bar{\gamma}(\sigma)$ *est dérivable dans* CPTT$'(\bar{\gamma})$*.*

Démonstration. Ce théorème se démontre par induction sur la dernière règle dans la dérivation du jugement $\Gamma; \Delta \vdash u : (\sigma, C)$, les règles à considérer étant celles présentées en fig. 6.3 plus haut. Un examen rapide montre que les cas de (ax) et (unit) sont immédiats, et que le cas de (const) est pris en charge par la définition même de CPTT$'(\bar{\gamma})$ qui compose $\mathfrak{t}$ avec $\rho_{\bar{\gamma}}$. Un autre cas direct est celui de (proj) : on obtient la dérivation attendue à partir de celle de la prémisse, qui existe par hypothèse, par une simple application de la projection correspondante en CPTT$'(\bar{\gamma})$.

Pour les autres cas, la conversion de dérivation sera assez simple, à condition de reconnaître un lemme préliminaire : si γ est une solution maximale de C et γ' une solution maxiamle de C' avec $C \subseteq C'$, alors toute dérivation de CPTT$'(\bar{\gamma})$ peut se transposer en une dérivation de CPTT$'(\bar{\gamma}')$. Pour bien le voir, il faut commencer par remarquer que comme $C \subseteq C'$ et qu'il s'agit ici d'ensembles de contraintes normalisés, on a nécessairement $\gamma(\alpha) \Rrightarrow \gamma'(\alpha)$ pour toute variable α appraissant dans C : ceci est dû au fait que l'ajout de contraintes normalisées réduit les bornes supérieures possibles, donc les solutions maximales. Étant donné une dérivation dans CPTT$'(\bar{\gamma})$, une dérivation similaire peut être produite dans CPTT$'(\bar{\gamma}')$ par la simple modification des coercions appliquées pour prendre en compte les modifications d'assignement des variables de type : si une coercion $\gamma(\alpha) \to \gamma(\beta)$ est utilisée dans la première, alors on utilisera une coercion $\gamma'(\alpha) \to \gamma'(\beta)$, qui est également bien défini ; le reste de la dérivation reste inchangé.

La démonstration s'achève alors par l'application de ce lemme aux règles restantes : comme l'ensemble des contraintes en conclusion contient toujours les contraintes des prémisses, il suffit de transformer les dérivations associées aux prémisses en dérivations dans CPTT$'(\bar{\gamma})$ selon le procédé présenté ci-dessus. Il ne reste alors plus qu'à appliquer la règle de CPTT qui correspond à la règle initialement examinée, afin de retrouver le jugement attendu. □

Il est à noter que si CPTT$'(\bar{\gamma})$ est plus permissif que CPTT, ses divergences avec la théorie principale peuvent aider à identifier précisément les coercions libres qui ont eu lieu au cours de la dérivation dans notre théorie contrainte. Ces divergences seront principalement constituées de variations sur le typage des constantes où une telle coercion a eu lieu. De manière générale, il est possible de se ramener à CPTT en réinterprétant les constantes modifiées comme les constantes initiales où une coercion supplémentaire est appliquée pour leur faire prendre leur nouveau type, en sachant que ce nouveau type est nécessairement un sous-type du type initial. Par exemple, si une constante $\mathbf{f}$ qui vérifie $\mathfrak{t}(\mathbf{f}) = \mathcal{P}a$ se retrouve réinterprétée par $\mathfrak{t}'(\mathbf{f}) = \mathcal{P}b$, alors $b \leq a$ et on substituera toute occurrence de $\mathbf{f}$ dans la dérivation en CPTT$'(\bar{\gamma})$ par $(\mathcal{P}c)\mathbf{f}$, avec $c : b \leq a$. [115] Afin d'observer plus en détail le fonctionnement et les effets du présent système, nous proposons de terminer cette section par deux nouveaux exemples offrant une étude de plus en plus approfondie d'un même énoncé.

Exemple 19. *Nous proposons ci-dessous une dérivation simplifiée pour la phrase « the sandwich is sitting at table 20 », qui apparaît notamment dans* [123]*. Nous supposons qu'il existe dans* $\mathbb{B}$ *un type* `a` *des entités animées et un type* `f` *de la nourriture incomparables. À des fins démonstratives,*

115. Notons qu'une telle réinterprétation ne peut pas se produire avec une constante qui n'est pas un prédicat à moins qu'elle apparaisse comme argument d'un prédicat dont le type attendu est plus petit, ce qui contrevient à toutes nos hypothèses ontologiques (cf. section 3.3).

les composantes de l'interprétation sémantique sont simplifiées en une constante de remplacement $\mathbf{the}' : (\mathcal{P}(\mathcal{P}\alpha \times \mathcal{P}\alpha), \{\alpha \sqsubseteq \mathtt{e}\})$*, une constante* **sandwich** $: (\mathcal{P}\beta, \{\beta \sqsubseteq \mathtt{f}\})$*, et une constante* **sitting_table_20** $: (\mathcal{P}\delta, \{\delta \sqsubseteq \mathtt{a}\})$*. Notre système permet alors la dérivation suivante :*

$$\dfrac{\dfrac{\dfrac{}{\vdash \mathbf{the}' : (\mathcal{P}(\mathcal{P}\alpha \times \mathcal{P}\alpha), \{\alpha \sqsubseteq \mathtt{e}\})} \quad \dfrac{}{\vdash \mathbf{sandwich} : (\mathcal{P}\beta, \{\beta \sqsubseteq \mathtt{f}\})}}{c \vdash (c\,\mathbf{the}')\mathbf{sandwich} : (\mathcal{P}\mathcal{P}\beta, \{\alpha \sqsubseteq \mathtt{e}, \beta \sqsubseteq \mathtt{f}, \beta \sqsubseteq \alpha\})} \quad \dfrac{}{\vdash \mathbf{sitting\_table\_20} : (\mathcal{P}\delta, \{\delta \sqsubseteq \mathtt{a}\})}}{c, c' \vdash ((c\,\mathbf{the}')\mathbf{sandwich})(c'\,\mathbf{sitting\_table\_20}) : (t, \{\alpha \sqsubseteq \mathtt{e}, \beta \sqsubseteq \mathtt{f}, \delta \sqsubseteq \mathtt{a}, \beta \sqsubseteq \alpha, \delta \sqsubseteq \beta\})}$$

où les variables de coercion sont $c : \mathcal{P}(\mathcal{P}\alpha \times \mathcal{P}\alpha) \to \mathcal{P}(\mathcal{P}\beta \times \mathcal{P}\beta)$ *et* $c' : \mathcal{P}\delta \to \mathcal{P}\beta$*. L'ensemble des contraintes ontologiques final admet pour solution maximale* γ *avec* $\gamma(\alpha) = \{\mathtt{e}\}$*,* $\gamma(\beta) = \{\mathtt{f}\}$ *et* $\gamma(\delta) = \{\mathtt{f}, \mathtt{a}\}$*, qui sous-tend un hétérotype ad hoc* $\mathtt{f} \bullet \mathtt{a}$ *indiquant que le sandwich a été réinterprété comme une entité animée qui entretient une relation spécifique avec l'aspect de nourriture initialement considéré. Une étude pragmatique pourra alors déterminer que la relation en question détermine une personne ayant commandé un sandwich dans un restaurant. Les coercions sont résolues comme* $c = \mathcal{P}(\exists k \times \exists k)$ *avec* $k : \mathtt{f} \leq \mathtt{e}$ *et* $c' = \exists p$ *avec* $p : \mathtt{f} \bullet \mathtt{a} \to \mathtt{f}$*. La dérivation équivalente dans* $\mathrm{CPTT}'(\bar{\gamma})$ *est aisée à déterminer et sera laissée en exercice. Notons toutefois qu'elle implique un retypage* **sitting_table_20** $: \mathcal{P}(\mathtt{f} \bullet \mathtt{a})$ *qui diverge du typage initial, signe immédiat qu'une coercion libre a eu lieu. Ce retypage ne peut malheureusement pas être évité, mais comme souligné plus haut, il peut néanmoins être interprété comme l'application d'une coercion cachée à la constante initiale, dont la prise en compte permet de se ramener à un terme de CPTT : ici, il s'agirait de considérer* $p' : \mathtt{f} \bullet \mathtt{a} \to \mathtt{a}$ *et de réinterpréter la constante en* $(\mathcal{P}p')$**sitting_table_20***. Pour mesurer pleinement la portée de cette réinterprétation, on peut considérer le fait qu'en lambda-calcul plus commun la coercion* $\mathcal{P}p'$ *correspondrait à* $\lambda u \lambda x.u(p'x)$*, tandis que* $\exists p$ *correspondrait à* $\lambda u \lambda x.\exists y.(py = x) \wedge uy$*. Alors, le sous-terme réinterprété* $(\exists p)\big((\mathcal{P}p')\mathbf{sitting\_table\_20}\big)$ *peut être compris comme le lambda-terme* $\lambda x.\exists y.(py = x) \wedge \mathbf{sitting\_table\_20}(p'y)$*.*

Exemple 20. *Nous allons ici reprendre la dérivation précédente, mais en donnant cette fois une plus grande complexité à* $\mathbf{the}'$*, qui va être défini comme correspondant au lambda-terme montagovien* $\lambda u \lambda v.\exists!x.ux \wedge vx$*, basé sur la constante de quantificateur existentiel avec unicité* $\exists!$ *(qui est ici un raccourci pour un terme plus complexe, voir section suivante) de type similaire à* $\exists$*, et que nous reconstruisons ici à l'aide de notre opérateur généralisé* $\wedge^{\mathtt{e}}$*. La dérivation de* $\mathbf{the}'$ *comme un terme défini est donnée ci-dessous. Par souci de lisibilité, nous posons le contexte contraint* $\Gamma = x : (\mathcal{P}\alpha \times \mathcal{P}\alpha, \{\alpha \sqsubseteq \mathtt{e}\}), y : (\varepsilon, \{\varepsilon \sqsubseteq \mathtt{e}\})$*, et nous nous permettrons de ne pas reproduire les typages des variables au début des jugements ni les contraintes répétées des ensembles parents.*

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\Gamma; \varnothing \vdash \wedge : (\mathcal{P}(\mathcal{P}\eta \times \mathcal{P}\eta \times \eta), \{\eta \sqsubseteq \mathtt{e}\})} \quad \dfrac{}{\Gamma; \varnothing \vdash x : (\mathcal{P}\alpha \times \mathcal{P}\alpha, \{\alpha \sqsubseteq \mathtt{e}\})}}{\Gamma; c \vdash (c\wedge)x : (\mathcal{P}\alpha, \{\ldots, \alpha \sqsubseteq \eta\})} \quad \dfrac{}{\Gamma; \varnothing \vdash y : (\varepsilon, \{\varepsilon \sqsubseteq \mathtt{e}\})}}{\Gamma; c, c' \vdash (c\wedge)x(c'y) : (t, \{\ldots, \varepsilon \sqsubseteq \alpha\})} \quad \dfrac{}{x; \varnothing \vdash \exists! : (\mathcal{P}\mathcal{P}\iota, \{\iota \sqsubseteq \mathtt{e}\})}}{x; c, c' \vdash \lambda y.(c\wedge)x(c'y) : (\mathcal{P}\varepsilon, \{\ldots\})}}{x; c, c', c'' \vdash \exists!(c''(\lambda y.(c\wedge)x(c'y))) : (t, \{\ldots \varepsilon \sqsubseteq \iota\})}}{c, c', c'' \vdash \lambda x.\exists!(c''(\lambda y.(c\wedge)x(c'y))) : (\mathcal{P}(\mathcal{P}\alpha \times \mathcal{P}\alpha), \{\ldots\})}$$

Notons alors Δ *la liste des coercions résultant de cette dérivation et* C *l'ensemble de contraintes ontologiques final, posons* $\mathbf{the}'$ *comme le terme ainsi dérivé, et désignons l'entière dérivation par la notation* π*. Nous soulignons ici le fait que* α *admet un minorant dans* C *par la contrainte* $\varepsilon \sqsubseteq \alpha$ *introduite à la troisième ligne de l'arbre en partant de sa plus haute branche. Nous pouvons*

alors répéter la dérivation de l'exemple précédent, en tenant compte cette fois des contraintes déjà présentes dans la dérivation de **the′**.

$$\dfrac{\dfrac{\begin{array}{c}\pi\\ \vdots\\ \Delta \vdash \textbf{the}' : (\mathcal{P}(\mathcal{P}\alpha \times \mathcal{P}\alpha), C)\end{array} \quad \dfrac{}{\vdash \textbf{sandwich} : (\mathcal{P}\beta, \{\beta \sqsubseteq \mathtt{f}\})}}{\Delta, c_1 \vdash (c_1\, \textbf{the}')\textbf{sandwich} : (\mathcal{P}\mathcal{P}\beta, C \cup \{\beta \sqsubseteq \mathtt{f}, \beta \sqsubseteq \alpha, \varepsilon \sqsubseteq \beta\})} \quad \dfrac{}{\vdash \textbf{sitting\_table\_20} : (\mathcal{P}\delta, \{\delta \sqsubseteq \mathtt{a}\})}}{\Delta, c_1, c_2 \vdash ((c_1\, \textbf{the}')\textbf{sandwich})(c_2\, \textbf{sitting\_table\_20}) : (t, C \cup \{\beta \sqsubseteq \mathtt{f}, \delta \sqsubseteq \mathtt{a}, \beta \sqsubseteq \alpha, \varepsilon \sqsubseteq \beta, \delta \sqsubseteq \beta, \varepsilon \sqsubseteq \delta\})}$$

Outre les nouvelles contraintes apparaissant dans C, la différence majeure de cette dérivation avec la précédente est l'insertion des contraintes $\varepsilon \sqsubseteq \beta$ et $\varepsilon \sqsubseteq \delta$ qui permet de relier les deux prédicats **sandwich** *et* **sitting_table_20** *par le type de leur argument commun, la variable existentiellement liée y dans la définition de* **the′**, *dont le type ε est résolu par* $\mathtt{f} \bullet \mathtt{a}$ *— tout comme δ dont les contraintes n'ont cependant pas changé. Les variables issues de C sont toutes résolues comme* `e`, *et β comme* `f` *de façon similaire à l'exemple précédent.*

6.5 Transformations structurelles

Dans les sections précédentes, nous avons étudié en détail les coercions issues du sous-typage et celles issues des hétérotypes, ainsi qu'une méthode permettant de les inférer automatiquement. Ces coercions ont toutes pour point commun de maintenir la structure des types qu'elles modifient : entre le domaine et le codomaine de ces transformations, seule la composante ontologique change. Dans la dernière section du présent chapitre, nous proposons de nous pencher davantage sur des transformations qui effectuent l'inverse, c.-à-d. qui agissent sur la composante structurelle des types sémantiques tout en maintenant leur composante ontologique. L'intérêt de telles transformations, que nous appellerons aussi *coercions transstructurelles*, serait de simplifier le stockage mémoire des implémentations du système compositionnel en se restreignant aux interprétations sémantiques minimales, quitte à modifier ces interprétations en fonction du contexte en syntaxe profonde, qui indiquerait le cas échéant les modifications à apporter au terme pour que la composition soit structurellement possible. L'idée serait de porter ces transformations comme une étape intermédiaire de l'analyse pré-sémantique qui se déroulerait *avant* l'inférence des coercions ontologiques. Nous allons donc ici énumérer quelques unes des coercions transstructurelles possibles, et tenter de déterminer certaines de leurs propriétés d'un point de vue catégorique.

Comme vu au chapitre 1, la diversité des structures de types sémantiques nécessaires à la modélisation linguistique est plutôt réduite, rassemblant des prédicats du premier et du second ordre, des modificateurs et des articulateurs logiques de ces prédicats, ainsi que des structures pouvant être interprétées comme des convertisseurs de prédicats d'un ordre à un autre, comme les déterminants. Ces derniers constituent la classe d'opérateurs la plus proche des transformations structurelles que nous souhaitons étudier : elles peuvent en effet être vues comme capables de « promouvoir » un prédicat du premier ordre (une structure nominale N) en un prédicat du second ordre (un syntagme nominal NP), et ce sans apporter de constantes sémantiques supplémentaires. Les transformations réalisées sont d'ordre logique, introduisant généralement des connecteurs et des quantificateurs pour permettre la combinaison des informations sémantiques intiales avec celle du futur prédicat argument. Montague [120] donne par exemple les termes suivants pour *the*, *a* et *every*, respectivement :

$$\lambda u \lambda v. \exists x. (\forall y. uy \Leftrightarrow (x = y)) \wedge vx, \qquad \lambda u \lambda v. \exists x. ux \wedge vx, \qquad \lambda u \lambda v. \forall x. ux \Rightarrow vx.$$

Ces déterminants permettent la bonne composition des prédicats issus du syntagme nominal avec ceux du syntagme verbal, et les éléments logiques qui soutiennent cette combinaison dépendent du déterminant utilisé. Pourtant, il y a de nombreux cas où la langue anglaise, comme de nombreuses autres langues d'ailleurs, autorise l'article zéro, c.-à-d. l'absence de déterminant syntaxiquement réalisé. [116] Il est alors nécessaire, en suivant les indications apportées par l'analyse syntaxique, de fournir une conversion à la volée pour passer du nom au syntagme nominal, en fonction du type de phénomène exprimé (pluriel, masse, etc.) : c'est ici que des coercions transstructurelles peuvent entrer en jeu. D'autres cas de transformations similaires et nécessitant les mêmes approches incluent notamment les appositions et certains verbes acceptant des syntagmes adverbiaux ou nominaux comme arguments. De fait, comme résumé par Partee [129], un certain nombre de transformations ont ainsi été considérées entre les formes générales de types e, $e \to t$ et $(e \to t) \to t$, dans un esprit proche des exemples donnés ci-dessus pour les déterminants.

Considérons pour commencer le cas de la promotion d'entités en prédicats du premier ordre, que nous souhaiterions généraliser dans CPTT par des transformations $\mathsf{ident}_a : a \to \mathcal{P}a$, pour tout $a \in \mathbb{B}$. La transformation équivalente dans une théorie montagovienne classique est donnée par le terme $\lambda x \lambda y.(x = y)$, et repose donc sur le prédicat d'égalité. Son interprétation catégorique nous est alors connue : il s'agit du morphisme singleton $\{\cdot\}_A : A \to \mathcal{P}A$, qui est un monomorphisme pour tout $A \in \mathsf{obj}(\mathcal{C})$ d'après la proposition 24, ce qui garantit la systématicité de la coercion associée. D'après Partee, cette transformation admet un inverse partiel qui n'est autre que l'opérateur iota de Russell. Cependant, en tant que morphisme partiel, cet opérateur est simplement donné par $(\{\cdot\}_A, \mathrm{id}_A) : \mathcal{P}A \to A$, dont le support est donc A lui-même ; il y a a priori peu de propriétés intéressantes supplémentaires à récupérer ici. L'intérêt de cette transformation est en particulier de permettre le stockage des informations relatives aux entités nommées sous la forme sémantique minimale d'entités constantes, tout en les utilisant comme des prédicats du premier ordre lorsque nécessaires. Cependant, on observera que la définition du morphisme singleton s'étend à tout objet de $\mathcal{C}$, de sorte que la transformation structurelle associée n'est pas nécessairement limitée aux types ontologiques : on peut alors introduire la coercion $\mathsf{ident}_\sigma : \sigma \to \mathcal{P}\sigma$ pour tout type $\sigma \in \mathbb{T}$.

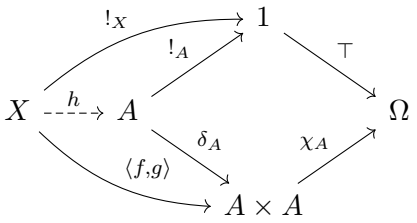
Une seconde transformation à considérer, caractéristique du système montagovien, est celle qui permet à une entité de devenir un prédicat du second ordre : l'opération $\mathsf{lift} = \lambda x \lambda u.ux$, que l'on retrouve notamment pour le traitement des noms propres comme sujets ou objets des verbes. Sa généralisation catégorique $\mathsf{lift}_A : A \to \mathcal{P}\mathcal{P}A$ est définissable dans $\mathcal{C}$ comme le nom du morphisme d'évaluation $\mathrm{ev}_A : \mathcal{P}A \times A \to \Omega$ précomposé avec l'isomorphisme de commutativité du produit. Si dans le cas ensembliste la fonction est clairement injective (elle associe à $x \in A$ l'ensemble $\{X \subseteq A \mid x \in X\}$), le résultat équivalent dans un cadre plus général est plus délicat à obtenir. Nous en proposons néanmoins une démonstration ci-après, qui requiert néanmoins un lemme préliminaire ; rappelons que dans tout topos $\mathrm{true}_A : A \to \Omega$ désigne la composée $\top \circ !_A$, et que le morphisme diagonal $\delta_A = \langle \mathrm{id}_A, \mathrm{id}_A \rangle : A \to A \times A$ est un monomorphisme : on notera dans la suite $\chi_A : A \times A \to \Omega$ sa caractéristique.

Lemme 6. *Soit $\mathcal{C}$ un topos. Pour tout $A \in \mathsf{obj}(\mathcal{C})$ et $f, g : X \to A$, $\chi_A \circ \langle f, g \rangle = \mathrm{true}_X$ si et seulement si $f = g$.*

Démonstration. Le sens réciproque est immédiat : par définition, $\chi_A \circ \langle f, f \rangle = \chi_A \circ \delta_A \circ f = \mathrm{true}_A \circ f = \mathrm{true}_X$. Pour l'autre sens, il suffit d'observer que le carré interne du diagramme suivant

116. Malgré l'absence de réalisation, une analyse syntaxique permet néanmoins de détecter les occurrences du phénomène. Il est même possible de distinguer syntaxiquement des cas distincts d'usage de l'article zéro, comme p. ex. dans l'analyse proposée par [17].

est par définition un carré de produit fibré : il s'en suit l'existence d'un morphisme $h : X \to A$ tel que le diagramme entier commute.

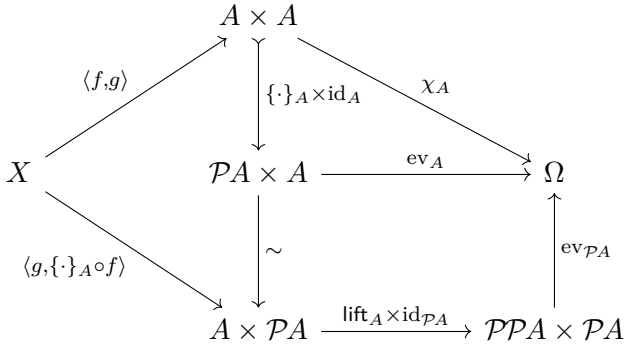


On a alors $\langle f, g\rangle = \delta_A \circ h = \langle h, h\rangle$, ce qui donne par composition avec les projections du produit $A \times A$ l'égalité $h = f = g$. □

Nous pouvons alors utiliser ce lemme pour établir notre résultat principal :

Proposition 43. *Pour tout topos $\mathcal{C}$ et tout objet $A \in \mathsf{obj}(\mathcal{C})$, le morphisme lift_A est mono.*

Démonstration. La première étape est de reconnaître que pour tous morphismes $f, g : X \to A$, le diagramme suivant commute :



Le triangle en haut à droite commute en effet par définition de $\{\cdot\}_A$, et le carré en bas à droite par définition de lift_A. De plus, le triangle de gauche commute par inspection, ce qui assure la commutativité de tout le diagramme. Soient maintenant $f, g : X \to A$ tels que $\mathsf{lift}_A \circ f = \mathsf{lift}_A \circ g$. Alors, par construction, on a l'égalité $\mathrm{ev}_{\mathcal{P}A} \circ \langle \mathsf{lift}_A \circ f, \{\cdot\}_A \circ f\rangle = \mathrm{ev}_{\mathcal{P}A} \circ \langle \mathsf{lift}_A \circ g, \{\cdot\}_A \circ f\rangle$; chacun des côtés de cette égalité correspondant au chemin extérieur bas dans le diagramme ci-dessus, avec deux fois f dans le premier cas, et f et g dans le second. Par commutativité, on peut substituer chacun de ces termes par le chemin extérieur haut, menant à $\chi_A \circ \langle f, f\rangle = \chi_A \circ \langle f, g\rangle$; or le terme de gauche est égal à true_X par le sens réciproque du lemme précédent. Par application du sens direct, on en déduit $f = g$. □

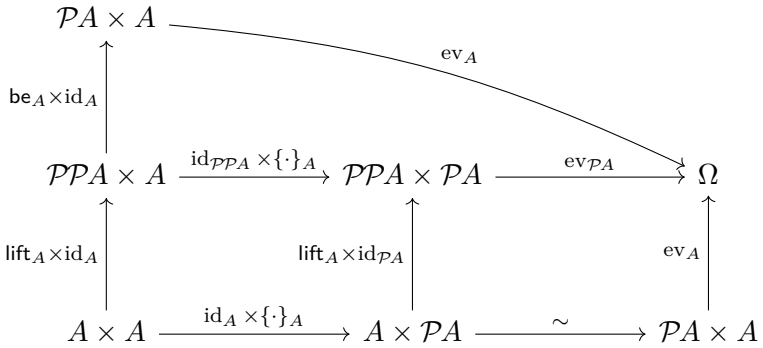
En dépit de leur correspondance en terme de domaine et codomaine, il ne sera pas surprenant de constater que les morphismes lift_A et $\{\cdot\}_{\mathcal{P}A} \circ \{\cdot\}_A$ sont en général distincts ; c'est par exemple le cas dans Set.

Une autre transformation présentée par Partee [129] est $\mathsf{be} : ((e \to t) \to t) \to (e \to t)$, une réduction du second au premier ordre, pensée notamment pour être un inverse des transformations données par les interprétations montagoviennes des déterminants *the* et *a*, que nous avons reproduites au début de cette section. En termes traditionnels, be est le lambda-terme $\lambda u \lambda x.u(\lambda y.x = y)$, qui est généralisable à tout topos par le morphisme $\mathsf{be}_A : \mathcal{P}\mathcal{P}A \to \mathcal{P}A$ obtenu comme nom de la composée $\mathrm{ev}_{\mathcal{P}A} \circ (\mathrm{id} \times \{\cdot\}_A) : \mathcal{P}\mathcal{P}A \times A \to \Omega$. Si ce morphisme n'est

pas mono, il dispose d'interactions intéressantes avec les morphismes précédemment considérés, comme déjà remarqué par Partee dans le cas ensembliste : le résultat suivant est donc une généralisation de cette observation à tout topos.

Proposition 44. *Soit $\mathcal{C}$ un topos. Pour tout $A \in \mathsf{obj}(\mathcal{C})$, $\mathsf{be}_A \circ \mathsf{lift}_A = \{\cdot\}_A$.*

Démonstration. Le cœur de cette démonstration consiste à reconnaître que le diagramme suivant commute :



En effet, le triangle supérieur commute par définition de be_A, le carré en bas à gauche commute par inspection, et le carré en bas à droite commute par définition de lift_A. Il suffit alors d'observer que le morphisme $A \times A \to \Omega$ obtenu en suivant le chemin extérieur droit est exactement la caractéristique χ_A du morphisme diagonal, dont $\{\cdot\}_A$ est le nom. La propriété universelle des exponentielles fait alors de $\{\cdot\}_A$ l'unique morphisme tel que $\mathsf{ev}_A \circ (\{\cdot\}_A \times \mathrm{id}_A) = \chi_A$, or la composée obtenue en suivant le chemin extérieur gauche indique que $\mathsf{be}_A \circ \mathsf{lift}_A$ vérifie la même propriété ; le résultat s'en déduit par unicité. □

Mais les propriétés de be_A ne s'arrêtent pas là : par sa définition même et par naturalité en A de l'isomorphisme $\mathcal{C}(X \times A, \Omega) \cong \mathcal{C}(X, \mathcal{P}A)$ pour tout X (cf. section 4.3), on peut en fait établir l'identité $\mathsf{be}_A = \mathcal{P}\{\cdot\}_A$. Il s'agit donc de la version interne du foncteur de produit fibré $\{\cdot\}_A^{-1}$, qui agit comme la fonction d'image inverse dans le cas ensembliste. De là la remarque de Partee concernant le fait de nombreux déterminants comme *a* que nous avons déjà vus ainsi que *exactly one* qui correspondrait au morphisme $\exists\{\cdot\}_A$, sont des sections de be_A : intuitivement, be_A associe à toute partie de $\mathcal{P}A$ l'union des singletons qu'elle contient, or les images d'une partie de A par les deux rétractions mentionnées contient bien les singletons des élements de cette partie. Ceci implique d'ailleurs que be_A est split épi. Quant à l'interprétation de *the*, elle satisfait une propriété similaire uniquement sur sa restriction aux singletons de A ; autrement dit, $\mathsf{be}_A \circ [\![\mathit{the}]\!] \circ \{\cdot\}_A = \{\cdot\}_A$, ce qui n'est pas étonnant au vu de la proposition précédente puisqu'on peut établir $[\![\mathit{the}]\!] \circ \{\cdot\}_A = \mathsf{lift}_A$. Nous observons par conséquent que les observations de Partee autour des changements de types se généralisent bien aux coercions transstructurelles et conservent leurs propriétés.

Au-delà des transformations étudiées par Partee, nous en considérons une dernière que nous estimons utile à nos objectifs linguistiques : la promotion d'un prédicat du premier ordre en modificateur de prédicat intersectif, que l'on retrouve pour les adjectifs, et qui prend la forme générale $\lambda u \lambda v \lambda x.ux \wedge vx$. Comme cette forme le suggère, le morphisme adj_A associé à cette transformation n'est autre que le nom associé à une certaine utilisation de la conjonction généralisée, donnée par la composée suivante :

$$\mathcal{P}A \times (\mathcal{P}A \times A) \xrightarrow{\sim} (\mathcal{P}A \times \mathcal{P}A) \times A \xrightarrow{\wedge_A \times \mathrm{id}_A} \mathcal{P}A \times A \xrightarrow{\mathsf{ev}_A} \Omega$$

Dans Set, cette transformation admet pour rétraction le morphisme $\exists\pi_A$, avec $\pi_A : \mathcal{P}A \times A \to A$ la seconde projection du produit, et est donc split mono. La généralisation de ce résultat à tout topos nous paraît tout à fait envisageable, mais requiert une preuve que nous n'avons pas eu le temps de formaliser. Par ailleurs, il est possible que d'autres transformations de ce type soient possibles, selon les formes observables des interprétations des adjectifs.

Par ce tour d'horizon des transformations structurelles, nous voyons donc que des changements de types issus la théorie montagovienne traditionnelle sont aisément généralisables à toute composante ontologique pour des composantes structurelles équivalentes, autorisant ainsi le déploiement de coercions transstructurelles dans des formalismes basés sur CPTT. En pratique, cette extension de CPTT peut être construite à l'aide d'un ensemble de coercions structurelles S définies par des jugements indépendants $\vdash_S c : \sigma \to \tau$ comme par exemple dans les règles suivantes : [117]

$$\frac{\sigma \in \mathbb{T}}{\vdash_S \mathsf{ident}_\sigma : \sigma \to \mathcal{P}\sigma} \qquad \frac{\sigma \in \mathbb{T}}{\vdash_S \mathsf{lift}_\sigma : \sigma \to \mathcal{P}\mathcal{P}\sigma} \qquad \frac{\sigma \in \mathbb{T}}{\vdash_S \emptyset_\sigma : \mathcal{P}\sigma \to \mathcal{P}\mathcal{P}\sigma}$$

$$\frac{\sigma \in \mathbb{T}}{\vdash_S \mathsf{be}_\sigma : \mathcal{P}\mathcal{P}\sigma \to \mathcal{P}\sigma} \qquad \frac{\sigma \in \mathbb{T}}{\vdash_S \mathsf{adj}_\sigma : \mathcal{P}\sigma \to \mathcal{P}(\mathcal{P}\sigma \times \sigma)}$$

Il est alors envisageable d'enrichir l'ensemble des termes avec des termes structurellement coercés en posant $\Lambda ::= \ldots \mid S\Lambda$, et d'autoriser pour leur introduction, à la différence des autres coercions, une application libre :

$$\frac{\Gamma \vdash u : \sigma \qquad \vdash_S s : \sigma \to \tau}{\Gamma \vdash su : \tau}$$

Les résultats précédents peuvent enfin s'exporter en règles équationnelles sur ces nouveaux termes, avec notamment les conversions suivantes :

$$\frac{\Gamma \vdash u : \sigma}{\Gamma \vdash \mathsf{be}_\sigma(\mathsf{lift}_\sigma u) = \mathsf{ident}_\sigma u : \mathcal{P}\sigma} \qquad \frac{\Gamma \vdash u : \mathcal{P}\sigma}{\Gamma \vdash \mathsf{adj}_\sigma u = \wedge^\sigma u : \mathcal{P}(\mathcal{P}\sigma \times \sigma)}$$

La seconde de ces règles équationnelles illustre la proximité entre coercions structurelles et prédicats : fondamentalement, chacune de ces coercions est définissable comme un terme de CPTT, et des égalités peuvent être ajoutées pour le souligner. Si nous les introduisons de façon séparée ici, c'est pour mettre en exergue la distinction conceptuelle que nous portons entre ceux-ci et les autres termes : les premiers sont des opérateurs servant à la bonne composition, et les seconds sont des interprétations sémantiques. Le passage d'une analyse en syntaxe profonde d'une phrase à sa représentation logique passe par plusieurs étapes à travers CPTT : chaque application syntaxique est convertie en une application dans CPTT en mobilisant les formes sémantiques minimales des unités lexicales considérées depuis le lexique ; mais les types de ces formes minimales n'ont pas nécessairement la bonne structure, qui nous est connue grâce aux catégories syntaxiques. C'est à ce moment là que les coercions transstructurelles entrent en œuvre : à chaque fois qu'une interprétation sémantique minimale ne respecte pas la structure attendue, une coercion appropriée est sélectionnée et appliquée. Une fois les structures de types ajustées, il ne reste plus qu'à mener la dérivation dans CPTT comme dans les sections précédentes, que ce soit directement ou via un détour par la théorie contrainte, selon les objectifs. Toute cette démarche est illustrée dans l'exemple qui suit, et qui concluera cette section.

117. La constante $\emptyset_\sigma$ correspond à la discussion sur l'article zéro du début de cette section ; sa définition précise est laissée non spécifiée ici afin de ne pas alourdir le propos avec des considérations sur la pluralité, qui dépassent malheureusement le cadre du présent travail.

Exemple 21. *Nous allons considérer ici une composition très simple pour interpréter la phrase « heavy books are boring ». Nous supposons disposer d'une analyse syntaxique simplifiée de cette phrase, dans laquelle are_boring est traitée comme une seule unité lexicale de catégorie syntaxique VP, associée à l'interprétation minimale* **boring** : $\mathcal{P}\mathtt{i}$. *Le lexique associe également à l'entrée heavy de catégorie Adj la constante* **heavy** : $\mathcal{P}\mathtt{p}$, *et à l'entrée books de catégorie N la constante* **book** : $\mathcal{P}\mathtt{ic}$ *; les types ontologiques* `p`, `i` *et l'hétérotype* `ic` *sont définis comme dans l'exemple 18. L'analyse syntaxique dégage alors deux opérateurs d'applications $F : Adj \times N \to N$ et $G : NP \times VP \to S$ et un opérateur d'ajustement $O : N \to NP$ correspondant à l'article zéro, réduisant la structure profonde de la phrase à la construction suivante :*

$$G(O(F(heavy, books)), are\_boring) : S$$

Nous considérons alors un homomorphisme $\mathscr{H}$ qui transforme cette structure syntaxique en une structure pré-sémantique non typée, selon la typologie illustrative donnée à la section 1.3. Soit E la fonction d'effacement de la composante ontologique des types sémantiques, définie inductivement sur $\mathbb{T}$ selon les égalités :

$$E(x) = \star \text{ pour } x \in \mathbb{B} \cup \mathbb{H}, \qquad E(1) = 1,$$
$$E(\sigma \times \tau) = E(\sigma) \times E(\tau), \qquad E(\mathcal{P}\sigma) = \mathcal{P}E(\sigma),$$

où $\star$ est un symbole distinct de tous les autres symboles utilisés jusqu'ici. L'image $E(\mathbb{T})$ est appelée ensemble des structures de types. *À l'instar du formalisme PTQ de Montague, nous pouvons alors définir une fonction T qui envoie chaque catégorie syntaxique sur une structure de types dans $E(\mathbb{T})$; nous en retiendrons les valeurs suivantes, qui correspondent à la traduction montagovienne traditionnelle :*

$$T(S) = t \qquad T(N) = \mathcal{P}\star \qquad T(NP) = \mathcal{P}\mathcal{P}\star$$
$$T(VP) = \mathcal{P}\star \qquad T(Adj) = \mathcal{P}(\mathcal{P}\star \times \star)$$

Pour toutes structures $\sigma, \tau \in E(\mathbb{T})$, nous posons de plus $A(\mathcal{P}(\sigma, \tau), \sigma) = \mathcal{P}\tau$, qui induit une fonction partielle $A : E(\mathbb{T}) \times E(\mathbb{T}) \rightharpoonup E(\mathbb{T})$. Enfin, si u désigne une unité lexicale, $L(u)$ désigne son interprétation sémantique minimale donnée comme un terme dans Λ, et $\mathfrak{l}(u)$ est le type sémantique de cette interprétation, qui est également fourni par le lexique.

Afin de construire la pré-sémantique associée à la représentation syntaxique de la phrase, nous allons considérer un langage intermédiaire composé de lambda-termes et d'opérateurs externes portant des annotations de structure de types. Ce langage, noté M, est défini inductivement selon les grammaires suivantes :

$$M ::= \Psi^{E(\mathbb{T})}_{E(\mathbb{T})} \qquad \Psi ::= \Lambda \mid \uparrow M \mid @(M, M)$$

Cette notation utilise un opérateur $\uparrow$ pour tous les ajustements unaires, et un opérateur $@$ pour toutes les formes d'applications. De plus, elle distingue en exposant et en indice de chaque terme deux structures de types dans $E(\mathbb{T})$. Celle en indice correspondra à la structure de types « attendue », c.-à-d. celle qu'est supposé avoir le terme en vertu de la catégorie syntaxique de son antécédent. En pratique, cet indice sera toujours un élément dans l'image de la fonction T définie plus haut. Quant à l'exposant, il représentera la structure de types « fournie », c.-à-d. celle obtenue soit directement depuis le lexique pour un lambda-terme, soit comme résultat d'une application compte tenu des structures fournies par l'abstraction et son argument.

L'idée derrière cette notation est de construire l'information des coercions transstructurelles à appliquer : lorsque les structures en exposant et en indice diffèrent, cela sera le signe qu'une telle

coercion allant du premier au second est nécessaire. Nous définissons alors l'homomorphisme $\mathscr{H}$ *à valeurs dans* M *comme suit :*

- *si* u *est une unité lexicale de catégorie syntaxique* C*, alors* $\mathscr{H}(u) = L(u)^{E(\mathfrak{l}(u))}_{T(C)}$;
- *si* X *est un opérateur* $C \to C'$ *et si* $\mathscr{H}(s) = v^{\sigma}_{T(C)}$*, alors* $\mathscr{H}(X(s)) = \uparrow\big(\mathscr{H}(s)\big)^{T(C)}_{T(C')}$;
- *si* Y *est un opérateur* $C \times C' \to C''$*, si* $\mathscr{H}(s) = v^{\sigma}_{T(C)}$*, et si* $\mathscr{H}(s') = v'^{\tau}_{T(C')}$*, alors* $\mathscr{H}(Y(s,s')) = @\big(\mathscr{H}(s), \mathscr{H}(s')\big)^{A(T(C),T(C'))}_{T(C'')}$

Dans le cas de figure de notre exemple donné plus haut, l'application de cet homomorphisme donne lieu à la construction suivante, qui constitue donc la pré-sémantique non typée de notre phrase de départ :

$$@\big(\uparrow\big(@\big(\mathbf{heavy}^{\mathcal{P}\star}_{\mathcal{P}(\mathcal{P}\star\times\star)}, \mathbf{book}^{\mathcal{P}\star}_{\mathcal{P}\star}\big)^{\mathcal{P}\star}_{\mathcal{P}\star}\big)^{\mathcal{P}\star}_{\mathcal{P}\mathcal{P}\star}, \mathbf{boring}^{\mathcal{P}\star}_{\mathcal{P}\star}\big)^{t}_{t}$$

Nous pouvons alors constater que deux sous-termes dans cette pré-sémantique ont un type fourni différent de celui qui est attendu : ceci marque les emplacements où une coercion transstructurelle devra s'appliquer. Nous définissons pour cela un nouveau morphisme $\mathscr{P}$ *qui envoie une pré-sémantique non typée vers un lambda-terme dans* Λ*, insérant des coercions lorsque les types attendus et fournis divergent, effaçant les opérateurs* $\uparrow$ *et remplaçant les opérateurs* @ *par des applications dans* CPTT*. Appliquée à la pré-sémantique ci-dessus, ce morphisme permet d'obtenir le lambda-terme non typé*

$$\emptyset\big((\mathsf{adj}\,\mathbf{heavy})\,\mathbf{book}\big)\,\mathbf{boring}.$$

Il ne nous reste alors plus qu'à produire un jugement de typage pour ce terme, dont la dérivation permettra d'insérer les coercions ontologiques nécessaires. Nous pouvons produire cette dérivation directement dans CPTT *si l'on veut écarter les coercions libres, ou bien dans la théorie contrainte dans le cas général ; nous allons illustrer cette seconde option. On supposera alors que l'application des coercions transstructurelles préserve les variables de types et leurs contraintes. La dérivation obtenue est alors la suivante, où les conventions utilisées dans l'exemple 20 sont maintenues :*

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \mathbf{heavy} : (\mathcal{P}\alpha, \{\alpha \sqsubseteq \mathtt{p}\})}}{\vdash \mathsf{adj}\,\mathbf{heavy} : (\mathcal{P}(\mathcal{P}\alpha \times \alpha), \{\dots\})} \quad \dfrac{}{\vdash \mathbf{book} : (\mathcal{P}\beta, \{\beta \sqsubseteq \mathtt{ic}\})}}{c \vdash (c\,(\mathsf{ajd}\,\mathbf{heavy}))\,\mathbf{book} : (\mathcal{P}\beta, \{\dots, \beta \sqsubseteq \alpha\})}}{c \vdash \emptyset\big(c\,(\mathsf{adj}\,\mathbf{heavy})\,\mathbf{book}\big) : (\mathcal{P}\mathcal{P}\beta, \{\dots\})} \quad \dfrac{}{\vdash \mathbf{boring} : (\mathcal{P}\delta, \{\delta \sqsubseteq \mathtt{i}\})}}{c, c' \vdash \emptyset\big(c\,(\mathsf{adj}\,\mathbf{heavy})\,\mathbf{book}\big)\,(c'\,\mathbf{boring}) : (t, \{\dots, \delta \sqsubseteq \beta\})}$$

La solution maximale à l'ensemble de contraintes final substitue α *par* $\mathtt{p}$*,* β *et* δ *par* $\mathtt{ic}$*. En notant* $p : \mathtt{ic} \to \mathtt{p}$ *et* $p' : \mathtt{ic} \to \mathtt{i}$ *les projections de l'hétérotype, on établit alors les inférences de coercions* $c = \mathcal{P}(\exists p \times p) : \mathcal{P}(\mathcal{P}\mathtt{p} \times \mathtt{p}) \to \mathcal{P}(\mathcal{P}\mathtt{ic} \times \mathtt{ic})$ *et* $c' = \mathrm{id}_{\mathtt{ic}}$*. De plus, on observe que le type de la constante* **boring** *a été réinterprété, ce qui suggère l'insertion d'une coercion cachée de la forme* $\mathcal{P}p' : \mathcal{P}\mathtt{i} \to \mathcal{P}\mathtt{ic}$*. Le terme final, typable dans* CPTT*, est donc*

$$\emptyset_{\mathtt{ic}}\big(\mathcal{P}(\exists p \times p)\,(\mathsf{adj}_{\mathtt{p}}\,\mathbf{heavy})\,\mathbf{book}\big)\,\big(\mathcal{P}p'\,\mathbf{boring}\big) : t.$$

Troisième partie

Études empiriques

Chapitre 7

Détermination empirique de types ontologiques

— Absolument impossible !!! Je vous avais confié une mission qui devait vous prendre trois ou quatre mois de travail !

— Ben justement ! Quand j'ai vu la somme de boulot que ça représentait, j'ai mis au point un logiciel qui a tout fait à ma place ! Tenez !

Léonard, t.39, p.21

Le présent chapitre rassemble les principales idées relatives à l'acquisition pratique de types ontologiques pertinents. Jusqu'à présent, les approches des types sémantiques introduites dans notre thèse ont été globalement théoriques : dans la partie I, nous avons étudié les propriétés linguistiques des types de sorte à pouvoir en façonner une caractérisation générale, établissant les lois comportementales auxquelles ils obéissent et quel est le rapport à la langue ; la conclusion principale étant alors que les types sémantiques *existent*, ou du moins présentent des raisons suffisantes pour motiver le postulat de leur existence. Cette existence implique naturellement celle des types ontologiques, dont nous avons pu alors énoncer la structure hiérarchique et diverses autres propriétés. Puis, dans la partie II, nous avons constitué étape par étape un modèle formel de ces types, expliquant comment les utiliser concrètement pour réaliser des interprétations sémantiques. Les types sémantiques sont intégrés à ce modèle suivant les principes identifiés dans la première partie, de sorte à fournir une implémentation mathématique cohérente de la théorie linguistique.

Comme nous avons pu le voir, cette implémentation repose sur une hypothèse : la donnée d'une ontologie de types — et incidemment, de celle d'un ensemble d'hétérotypes. Toutes les formalisations menées dans cette partie, et en particulier au chapitre 6, s'appuient sur cette hypothétique donnée de $(\mathbb{B}, \leq, e)$, dont on a établi l'existence, mais pour laquelle nous n'avons pas fourni d'instanciation. L'objectif de cette partie III, et donc du présent chapitre qui constitue son unique composante, est de pallier l'absence d'information sur le contenu concret d'une telle ontologie en proposant des pistes d'études adaptées sur corpus. En effet, la théorie linguistique veut que les types sémantiques se manifestent dans le langage par notre capacité à reconnaître des énoncés déviants, qui sont généralement écartés de la langue courante. Un relevé des prédications utilisées empiriquement serait donc en mesure de fournir des indications précieuses sur la répartition de ces types, qui pourront alors corroborer ou non les intuitions qui régissent jusqu'ici

les exemples de types ontologiques utilisés dans les démonstrations de formalismes sémantiques comme ceux introduits au chapitre 1, mais aussi comme celui construit au chapitre 6.

Afin de collecter des informations supplémentaires sur les types ontologiques, nous avons entrepris l'écriture d'un algorithme d'analyse des données textuelles auquel nous nous réfèrerons dans la suite sous le nom d'ATHICE, acronyme pour l'anglais « Acquisition of Type Hierarchies through Corpus Exploration ». Cet algorithme assez général a été implementé en Python afin de tester empiriquement son intérêt et d'identifier concrètement les difficultés auxquelles l'entreprise de collecte de types ontologique sur une base textuelle est confrontée. Dans la section 7.1, nous donnerons une brève introduction à la linguistique de corpus et plus précisément à des tâches qui se rapprochent de nos objectifs, ainsi qu'une présentation des éléments sur lesquels nous appuyons notre propre approche ; avant d'introduire en section 7.2 les bases théoriques qui ont inspiré la conception d'ATHICE. Nous entrerons ensuite dans les détails de l'algorithme et de son implémentation en section 7.3, et concluerons en section 7.4 par la présentation des premiers résulats ainsi qu'une discussion sur les leçons à en tirer pour de futures améliorations.

7.1 Linguistique formelle et données empiriques

Depuis une vingtaine d'années environ au moment où nous écrivons ces lignes, le domaine du traitement automatique des langues (TAL) a développé de nombreuses études expérimentales à partir de corpus de textes informatisés. Ces corpus offrent en effet des fragments de la langue en conditions réelles qui donnent une bonne idée d'un usage courant de celle-ci, permettant à la fois d'explorer des phénomènes langagiers nouveaux avec la manière de les modéliser, et d'entraîner, tester et vérifier les programmes dédiés à des tâches syntaxiques ou sémantiques concrètes. La nature de ces tâches présente une grande diversité, et inclut de manière non exhaustive l'étiquetage automatique de rôles sémantiques, la résolution d'ambiguïtés de sens, la classification lexicale, et la résolution de métaphores. Ces tâches sont souvent considérées au travers du prisme de l'*apprentissage statistique*, dont le principe très général est d'emmagasiner des informations en parcourant une certaine quantité de données annotées afin de pouvoir les réutiliser sur des données nouvelles. Ce principe général englobe diverses approches algorithmiques, et il est probable que notre objectif et l'algorithme ATHICE que nous introduirons dans les pages suivantes puissent s'y rattacher.

Un autre domaine proche est celui des bases de données lexicographiques, qui peuvent être décrites comme des collections d'informations sur les mots d'une langue et sur les relations syntaxiques et sémantiques qu'ils entretiennent entre eux. Notre objectif de détermination de types ontologiques se rapproche beaucoup de telles bases de données dans la mesure où, comme développé dans le chapitre 3, les composantes ontologiques sont avant tout caractéristiques des expressions, et en particulier des unités lexicales : une ontologie de types peut donc se ramener à une classification de ces unités en fonction d'une propriété sémantique basique qui régit leur capacité à se combiner de manière sensée. Des exemples bien plus complets de bases lexicographiques ou sémantiques dans un ordre d'idées proche incluent WordNet [118, 58] que nous avions déjà évoqué au chapitre 1, mais aussi FrameNet [12], qui propose des représentations en sémantique des cadres, ainsi que le Réseau Lexical du Français (RLF) [111] ; chacun implémentant divers degrés d'informations syntaxiques et de relations sémantiques entre unités lexicales. Bien entendu, la nature des types sémantiques telle que proposée dans la partie I du présent texte rend notre objectif bien plus « primitif » que les exemples sus-cités, puisque toutes les propriétés sémantiques qui y sont listées impliquent des types sémantiques sous-jacents mais non nécessairement identifiés comme tels.

Cela signifie en particulier que la tâche envisagée ici est probablement nouvelle en comparaison des tâches et des bases de données qui ont été proposées jusqu'ici pour le TAL. En effet, la plupart des propriétés sémantiques abordées par ces initiatives sont généralement plus complexes et plus détaillées que le cadre des types sémantiques, incluant notamment des critères de sélection argumentale plus subtils que ceux dont les types sémantiques ont besoin ; or, il convient de rappeler ici que nous avons vu en section 3.1 que les types sémantiques ne sauraient être aussi détaillés que de telles propriétés, y compris en comparaison de relations très générales comme la synonymie ou l'hyperonymie. En revanche, nous nous devons de reconnaître que le problème de peupler concrètement une ontologie de types comme entendue dans notre théorie a une bien moindre portée applicative que les données collectées par les projets susmentionnés ; de fait, son unique intérêt serait a priori d'instancier les ontologies utilisées par les théories formelles comme celle que nous avons développée dans la partie II afin de les fonder empiriquement et d'en faire ds modèles aussi fidèles à la langue que possible. On pourrait éventuellement imaginer des applications connexes à ces ontologies, par exemple dans le cadre de l'étude des métaphores, mais des approches avancées existent déjà dans ce domaine (cf. p. ex. [168]), et la question de savoir si nos ontologies de types apporteraient une quelconque aide de ce point de vue-là reste sujette à débat.

Revenons maintenant aux corpus de textes, qui constituent le matériau de base pour un grand nombre d'études en TAL. Afin d'exploiter pleinement le potentiel d'un corpus, il est souvent nécessaire de lui adjoindre de l'information linguistique supplémentaire, qui prend la forme d'*annotations* (cf. [68]). Ces annotations sont de natures variées, incluant généralement des renseignements syntaxiques comme les parties du discours des unités lexicales ou bien les relations structurelles des phrases, et parfois des informations d'ordre sémantique, comme par exemple en utilisant la notion de *champ sémantique* [186] héritée de la sémantique structuraliste (cf. section 2.4), ou encore des indications de sens comme produites par une méthode de désambiguïsation lexicale [121]. Dans une certaine mesure, nous pourrions d'ailleurs imaginer que de futures tâches pourraient bénéficier d'informations sur les types ontologiques des unités lexicales, dont la précision est plus diffuse que pour les autres sortes d'annotation sémantique ; notre objectif pourrait alors à terme permettre la production d'annotations de ce genre. Mais pour l'heure, notre algorithme sera surtout intéressé par l'utilisation d'annotations déjà réalisées pour effectuer sa propre tâche et construire une ontologie de types. Au vu de la relation entre les types et des autres propriétés sémantiques, il est probable que nous ayons à utiliser davantage des annotations syntaxiques que sémantiques, mais toute information disponible pourra être bénéfique si elle est utilisée correctement.

Nous allons en particulier nous focaliser sur un schéma d'annotation spécifique, appelé *Universal Dependencies* (UD) [52, 122]. Ce schéma d'annotation très détaillé décompose les phrases mots par mots en les numérotant et en décrivant pour chacun leur forme réalisée et leur lemme de base, leur partie du discours et diverses autres informations lexicales et flexionnelles, et surtout une structure détaillée de *dépendances* qui représentent les liens syntaxiques entre les mots, chaque lien portant une étiquette qui décrit la nature de la relation qu'il exprime. Ces relations incluent diverses formes de couples prédicats-arguments, comme le sujet du verbe nsubj, l'objet direct obj et l'objet indirect iobj, la modification adjectivale amod, la modification adverbiale advmod, et bien d'autres formes de liens syntaxiques. L'exploitation de ces liens pour caractériser les relations de prédication entre mots, dont nous avons eu l'occasion de voir qu'elle était essentielle à la compositionnalité et donc à l'établissement des types sémantiques, semble être prometteuse pour mener à bien notre objectif, et c'est pourquoi nous prenons ce schéma d'annotation comme point de départ de notre réflexion sur la construction empirique de notre ontologie. Mais pour poursuivre sur cette voie, nous devons également mobiliser un peu plus d'éléments théoriques

quant aux liens entre prédications et types ontologiques : la section suivante va donc traiter de cette question.

7.2 De la théorie sommersienne à l'analyse de corpus

Après avoir posé les bases des études linguistiques empiriques, nous en venons maintenant au cœur du problème. Notre objectif est de déterminer quels sont les types ontologiques utilisés par la langue courante, ceux qui viendront instancier l'ontologie de types qui est au centre de CPTT (cf. chapitre 6). Comment des données textuelles pourraient nous en apprendre plus sur ces types ? Pour le comprendre, nous allons entrer des les détails de la théorie de Fred Sommers, dont nous avions déjà donné un aperçu en section 2.5. Nous nous étions alors contenté de rassembler les idées majeures et les résultats les plus importants de Sommers, afin de mettre en lumière les principes fondamentaux qui ont par la suite inspiré nos propres principes, énoncés au chapitre 3 ; mais son travail repose sur des bases qu'il nous est possible de formaliser et de mettre en relation avec des éléments disponibles dans les corpus annotés, faisant ainsi émerger les idées qui ont guidé la conception d'ATHICE. Nous développons par conséquent dans la suite de cette section les théories de Sommers telles qu'introduites principalement dans [170, 171], avant d'expliciter le lien possible entre ces théories et la linguistique de corpus.

Comme évoqué en section 2.5, la théorie de Sommers est fondée sur la notion de *relation de prédication*, qui caractérise les paires d'expressions dont la combinaison prédicative a du sens, c.-à-d. ne présente pas d'erreur de catégorie. Cela suppose donc de distinguer en premier lieu les expressions de la langue (unités lexicales ou syntagmes) en fonction de leur propension à dépendre ou non d'autres expressions. Pour formaliser cela, on distinguera un ensemble $\mathcal{P}$ de *prédicats*, constitué de toutes les expressions de la langue qui peuvent occuper une position prédicative en syntaxe profonde, et un ensemble $\mathcal{A}$ d'*arguments*, contenant de même les expressions pouvant occuper une position argumentale. Cette distinction repose sur des distinctions syntaxiques, et se recoupe assez bien avec la répartition des catégories synatxiques : ainsi, les noms communs et propres seront quasi-systématiquement considérés comme des arguments, alors que les adjectifs et les adverbes seront plutôt des prédicats. Certaines expressions ont cependant le comportement dual de prédicats et d'arguments, comme les syntagmes verbaux : ces derniers prennent un argument sujet, mais peuvent être arguments de certains verbes ou d'adverbes. Fondamentalement, il n'y a ainsi pas de raison de considérer $\mathcal{P}$ et $\mathcal{A}$ comme disjoints.

Cependant, la prédication en elle-même est une relation asymétrique et paramétrée par les différentes positions de la structure argumentale : si un argument donné est acceptable pour un prédicat, le prédicat en question ne pourra jamais être considéré comme argument de la première expression, même si cette dernière dispose aussi d'une forme prédicative. Par ailleurs, les arguments sujets et les arguments objets d'un verbe transitif sont très différents, parfois même disjoints. Dès lors, établir une relation qui rend compte de ces paramétrages et de ces distinctions suppose un « découpage » plus fin de la répartition des expressions, en distinguant ces dernières sur deux aspects. Le premier aspect est la séparation selon la position : pour toute expression $w \in \mathcal{P} \cap \mathcal{A}$, on introduira deux copies distinctes $w_p \in \mathcal{P}$ et $w_a \in \mathcal{A}$ et on effacera simplement w. Le second aspect est la distinction des arguments possibles dans la structure argumentale : pour toute expression $w \in \mathcal{P}$, si w admet plusieurs arguments caractérisés par des rôles $a_1, \ldots, a_n$, on introduira des copies multiples $w_{a_1}, \ldots, w_{a_n} \in \mathcal{P}$, et on effacera ensuite w. Nous exigeons par ailleurs que ces traitements soient effectués dans cet ordre, de sorte qu'une expression à la fois argumentale et multiplement prédicative ne donne lieu qu'à une seule copie dans $\mathcal{A}$. Nous pouvons y voir une généralisation de l'approche de Sommers, qui ne considère dans

les exemples qu'il traite que des prédicats monadiques : notre découpage permet de s'y ramener et d'exploiter malgré tout ses résultats ; cette démarche est par ailleurs justifiée par le fait que Sommers ne tient pas compte de critères structurels autre que cette prédication monadique, ce qui nous permet donc de découper les prédicats polyadiques sans perte de généralité.

Nous disposons donc d'expressions parfois annotées et réparties en deux ensembles disjoints $\mathcal{P}$ et $\mathcal{A}$. Sommers propose alors de définir une relation de sens binaire sur l'ensemble $\mathcal{P} \cup \mathcal{A}$ des expressions, notée $U \subset (\mathcal{P} \cup \mathcal{A}) \times (\mathcal{P} \cup \mathcal{A})$, qui répond aux caractéristiques suivantes : si un prédicat $P \in \mathcal{P}$ et un argument $t \in \mathcal{A}$ peuvent être composés par application et former une expression qui ne présente pas d'erreur de catégorie, alors $U(P, t)$; et si un argument $t \in \mathcal{A}$ peut être conjointement accepté par deux prédicats $P, P' \in \mathcal{P}$, alors $U(P, P')$. Ces caractéristiques ne constituent cependant pas l'entièreté de la relation, car deux éléments de $\mathcal{A}$ peuvent tout à fait être en relation U : seulement, nous ne disposons pas de critères suffisants pour tester directement si c'est le cas, à l'opposé des relations entre éléments de $\mathcal{P}$ ou entre un élément de $\mathcal{A}$ et un élément de $\mathcal{P}$. On note par ailleurs N la négation de U, c.-à-d. la relation complémentaire $(\mathcal{P} \cup \mathcal{A}) \times (\mathcal{P} \cup \mathcal{A}) \setminus U$ obtenue pour les paires de mots dont la combinaison donne une prédication absurde. La relation U est réflexive et symétrique, mais comme Sommers s'est appliqué à montrer dans [171], elle n'est pas transitive en général.

Cette relation U peut être raffinée en une relation de prédication $\blacktriangleright \subset \mathcal{P} \times \mathcal{A}$ entre un prédicat et un argument : on dira que $P \in \mathcal{P}$ est *prédicable* de $t \in \mathcal{A}$, ou de façon équivalente que t est dans la *portée* de P, lorsque $U(P, t)$; auquel cas on privilégiera la notation $P \blacktriangleright t$. Sommers utilise alors cette seconde relation pour définir plusieurs notions de types, préfixées par les lettres grecques α et β dans les deux casses minuscules et majuscules. Ces types représentent des ensembles d'expressions, obtenus selon les propriétés prédicatives que leurs éléments manifestent. Les deux types les plus élémentaires à définir sont les α-types, qui incarnent la notion de portée évoquée précédemment, et les B-types, qui leur sont symétriques ; pour tout prédicat $P \in \mathcal{P}$ et pour tout argument $t \in \mathcal{A}$, on pose ainsi les définitions suivantes :

$$\alpha(P) = \{t \in \mathcal{A} \mid P \blacktriangleright t\} \qquad B(t) = \{P \in \mathcal{P} \mid P \blacktriangleright t\}$$

Les deux autres sortes de types, nommés β-types et A-types, ne sont pas élémentaires car ils se construisent à partir des définitions précédentes ; pour tout α-type α et tout B-type B, on a :

$$\beta(B) = \{t \in \mathcal{A} \mid B(t) = B\} \qquad A(\alpha) = \{P \in \mathcal{P} \mid \alpha(P) = \alpha\}$$

Il s'agit donc de classes d'équivalences pour la relation « définir le même α-type » ou « B-type », selon les circonstances. La nomenclature utilisée ici est celle définie par Sommers lui-même ; elle est organisée de sorte que les minuscules correspondent à des sous-ensembles d'arguments et les majuscules à des sous-ensembles de prédicats, et qu'une bijection soit définissable entre une instance en minuscule et une en majuscule pour la même lettre. En effet, chaque α-type par exemple détermine un unique A-type, et réciproquement ; et une propriété similaire lie les deux autres sortes de types. Les α-types définissent exactement ce que Sommers considère comme les catégories ontologiques : ils sont à la base de notre ontologie de types.

La relation U et les types ainsi définis présentent plusieurs propriétés d'intérêt que Sommers s'est appliqué à démontrer dans ses articles. On commencera par remarquer une première propriété qui, bien que non explicitement énoncée par Sommers, semble sous-entendu dans ses travaux :

(T0) Pour tout $t \in \mathcal{A}$, il existe un prédicat canonique $/t/ \in \mathcal{P}$ tel que pour toute expression $w \in \mathcal{P} \cup \mathcal{A}$, $U(t, w)$ si et seulement si $U(/t/, w)$.

L'idée derrière ce principe est d'assimiler les expressions arguments aux prédicats formés par leur combinaison avec le verbe être : par exemple, le mot anglais *philosopher* est un nom commun et est donc considéré comme un argument, mais *is a philosopher* est un prédicat qui lui est très proche et qui simplifie l'établissement de relations U entre arguments. L'usage de ce prédicat canonique permet alors de définir la catégorie d'une expression quelle qu'elle soit : pour toute expression w, on notera $|w|$ l'ensemble dénoté par $\alpha(w)$ si $w \in \mathcal{P}$, ou par $\alpha(/w/)$ si $w \in \mathcal{A}$. Cette notation permet alors d'énoncer le principe général suivant, qui est l'une des lois les plus importantes de Sommers ; on en trouvera une démonstration dans [171].

(T1) Pour toutes expressions $w, w' \in \mathcal{P} \cup \mathcal{A}$, $U(w, w')$ si et seulement si $|w| \subseteq |w'|$ ou $|w'| \subseteq |w|$.

Une conséquence immédiate de ce résultat est le fait que deux catégories ontologiques ont une intersection non vide si et seulement si l'une est incluse dans l'autre ; autrement dit, $N(w, w')$ si et seulement si $|w| \cap |w'| = \varnothing$.[118] Il est donc impossible pour des catégories de se chevaucher partiellement. Cette propriété explique notamment la nécessité d'introduire les hétérotypes comme des composés externes à l'ontologie.

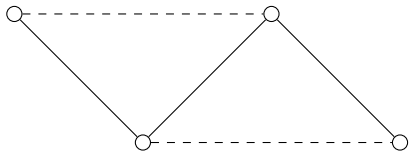
Nous pouvons voir dans les relations d'inclusion de ce principe les prémisses d'une idée de hiérarchisation des types : chaque α-type détermine en effet une classe de prédicats associée, qui est un A-type, ces classes étant par construction disjointes. Ceci permet de alors d'importer l'ordre d'inclusion sur les α-types et d'en faire un ordre partiel sur les A-types. Si l'on assimile ces types aux critères ontologiques des types sémantiques de ces prédicats, nous commençons alors à voir poindre l'ébauche d'une ontologie ordonnée. Cette impression est complétée par d'autres principes énoncés par Sommers, qui découlent directement du principe (T1) et lui sont même équivalents : il s'agit simplement de reformulations permettant de mieux cerner les effets de ce principe sur la structure des catégories ontologiques. La première version détermine la hiérarchisation en affirmant que la catégorie d'une expression qui est en relation U avec deux autres expressions qui ne sont pas reliées entre elles contient les catégories associées à celles-ci :

(T2) Pour toutes expressions $p, q, r \in \mathcal{P} \cup \mathcal{A}$, on a $U(p, q)$, $U(q, r)$ et $N(p, r)$ si et seulement si $|p| \subset |q|$, $|r| \subset |q|$, et aucune autre inclusion n'est valide.

La seconde reformulation est encore plus importante : parfois mentionné sous le nom anglais *tree rule*, ce principe impose à la hiérarchie ainsi engendrée une structure très particulière qui, comme son nom l'indique, se ramène à celle d'un arbre :

(T3) Pour toutes expressions p, q, r, s, la formule $\neg(U(p, q) \wedge U(q, r) \wedge U(p, s) \wedge N(p, r) \wedge N(r, s))$ est valide.

Pour comprendre pourquoi, il peut être utile de visualiser la relation U comme un graphe dont les sommets sont les expressions de $\mathcal{P} \cup \mathcal{A}$.[119] Alors, le principe (T3) interdit l'existence dans ce graphe du motif ci-dessous, où les pointillés représentent l'absence d'arête :



118. En effet, s'il existe $t \in |w| \cap |w'|$, alors t est un argument commun aux prédicats associés à w et w' ; d'après les propriétés générales de U, cela implique $U(w, w')$ et donc une contradiction.

119. On pourra se restreindre aux unités lexicales pour avoir un graphe fini ; cette restriction est supposée pouvoir être menée sans perte de généralité.

On notera que l'existence d'une arête entre les sommets aux extrémités du motif n'est pas importante ici. En théorie des graphes, ce motif capture deux formes bien connues de sous-graphes induits : le chemin à quatre sommets (ou P_4) et le cycle à quatre sommets (ou C_4). [120] Les graphes qui ne comportent pas de sous-graphes induits de la forme P_4 ou C_4 sont généralement désignés sous le nom de *graphes trivialement parfaits* et possèdent de nombreuses propriétés, parmi lesquelles le fait de représenter le graphe de comparabilité d'un arbre [77, 190]. Autrement dit, il est possible à partir d'un tel graphe de reconstruire un ordre partiel qui est essentiellement un arbre. Ce résultat permet donc d'appuyer l'organisation de l'ontologie en arbre, et justifie au passage de manière théorique l'existence d'une catégorie maximale : la seule condition est que le graphe soit connexe, ce qui est inévitable dans le cas des unités lexicales d'une même langue puisque deux composantes connexes distinctes définiraient intuitivement deux langues différentes [171, §III].

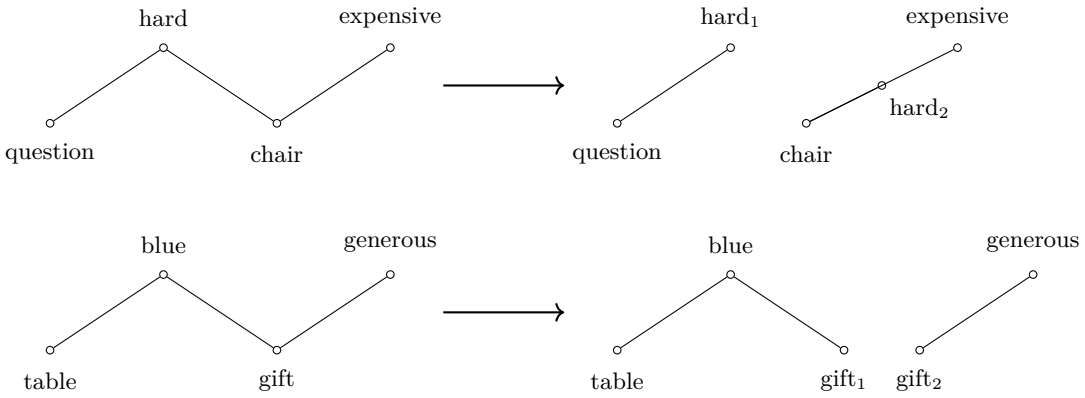
Cette structure sur le graphe des unités lexicales se ramène alors à une hiérarchie de catégories ontologiques par la formations de classes d'équivalence en appliquant les définitions sommersiennes des différentes sortes de types. L'arbre résultant comporte les β-types en guise de feuilles et les A-types comme nœuds internes ; les α-types correspondent aux sous-arbres, et sont donc formés de l'union des β-types à leurs feuilles ; et les B-types correspondent aux branches de l'arbre, et sont formés de l'union de tous les A-types rencontrés en remontant le long de ces dernières. La correspondance bijective observée entre sortes de même lettre devient alors assez naturelle : chaque sous-arbre détermine un α-type par l'union de ses feuilles et un A-type par sa racine, et chaque branche détermine un B-type par l'union de ses nœuds internes et un β-type par son unique feuille. Ainsi, la résolution des différents liens de prédicabilité détermine entièrement l'arbre des classes d'arguments et de prédicats, qui forme l'ontologie finale. On insistera alors sur le fait qu'une telle ontologie répartit toujours les arguments aux feuilles et les prédicats dans les nœuds internes ; il sera donc impossible de trouver un argument plus haut dans la hiérarchie. Cela signifie aussi que la composante ontologique des arguments est toujours plus petite que celle des prédicats, confirmant ainsi le principe (O2) énoncé en sous-section 3.3.2.

C'est alors que commence à s'esquisser les possibilités d'application empirique de ces principes. Comme nous venons de le voir, l'ontologie des types sémantiques est une hiérarchie fondée sur les unités lexicales de la langue et la manière dont elles interagissent ; sa construction est entièrement déterminée par la valeur empirique de la relation de sens U, et encore plus par la relation de prédication $\blacktriangleright$ qu'elle engendre. Pour résoudre notre problème de déterminer l'ontologie, nous pouvons donc nous ramener au problème plus simple de déterminer ces relations, et les données issues de corpus de textes peuvent aider en ce sens. L'idée générale est alors de considérer que toutes les combinaisons de mots dans le corpus sont correctes, et que les prédications qui y sont observables sont donc des exemples valides de la relation U. Il y a bien entendu deux problèmes immédiats qui surgissent de cette hypothèse : premièrement le fait qu'elle est fondamentalement fausse, puisque les figures de styles comme les métonymies et les métaphores sont fréquentes dans le langage — et donc dans les corpus — alors qu'elles doivent théoriquement être considérées comme des erreurs de catégories, et deuxièment le fait qu'un corpus ne peut donner qu'un aperçu partiel de U, y compris parmi les unités lexicales observables puisque deux mots peuvent parfaitement avoir du sens lorsque utilisés conjointement sans jamais qu'une telle utilisation soit observée dans le corpus d'étude — c'est notamment le cas pour les paires d'arguments, étant donné qu'aucun moyen direct d'établir U dans ce cas n'existe a priori.

120. cf. par exemple [184] pour une introduction à théorie des graphes, notamment §1.2 pour la notion de *sous-graphe induit* et §2.1 pour une définition formelle des chemins et des cycles ; [21] fournit les mêmes informations au §1.1 et définit en plus au §1.4 la notion de graphe de comparabilité évoquée plus bas.

Si l'on souhaite travailler avec cette hypothèse, il est donc nécessaire de fournir des réponses suffisamment convaincantes à ces problèmes. Le second est un problème commun de rareté des données : ne serait-ce que par sa finitude, aucun corpus ne peut démontrer de manière exhaustive tous les comportements possibles du langage ; et le remède le plus simple à cet état de fait est l'inférence statistique. Étant données les prédications observées dans le corpus, il nous faudra construire les classes de prédicats et d'arguments non comme des objets formels directement, mais comme des *regroupements* inférés à l'aide d'une certaine mesure de similarité entre les unités lexicales, fondée sur leur comportement prédicatif. [121] Cette approche inférentielle nous permettra ainsi d'approximer U en reconstruisant les relations de sens probables qui ne sont pas directement observées, dont les relations entre paires d'arguments qui auraient de toute façon posé problème, sachant qu'a priori seuls les arguments d'un même β-type sont censés démontrer ce lien.

Quant au premier de ces deux problèmes, nous estimons pouvoir lui donner une réponse satisfaisante à travers la réponse à une autre question. Le principe (T3) prévient en effet de manière théorique l'existence d'un certain motif dans le graphe de comparabilité de la relation U. Mais que se passe-t-il en pratique si au cours de notre inférence nous obtenons réellement un tel motif ? Sommers, notamment dans [171, 174], nous explique que ce cas peut effectivement être rencontré en pratique, et doit être pris comme un signe d'*équivocité*, c.-à-d. d'ambiguïté parmi différents sens attribuables à une même unité lexicale. L'ambiguïté en question porte nécessairement sur l'un des deux sommets du milieu du motif, et ne peut être résolue qu'en séparant l'un de ces sommets en deux sommets distincts représentant chacun un sens différent de l'unité concernée. Les deux schémas ci-dessous, inspirés directement des exemples de Sommers, montrent comment le prédicat *hard* et l'argument *gift* peuvent être séparés pour détruire le motif non souhaité :



On remarquera notamment que l'un des sens nouvellement établis pour *hard* entre en relation U avec *expensive*. Ainsi, *hard* est désambiguïsé pour son sens physique de « solide, résistant » et son sens intangible de « difficile, demandant des efforts » ; et *gift* est désambiguïsé de même pour son sens concret d'« objet offert » et son sens abstrait d'« acte d'offrir ».

La règle de Sommers nous encourage ainsi à clarifier les sens des mots par la désambiguïsation, et ce pour tous les types d'équivocité possibles : outre la polysémie concernée par les exemples précédents, on pourra retrouver dans la même situation des cas d'homonymie notamment, mais

121. Signalons ici que, par souci de rigueur dans la langue utilisée et afin de faire vivre les équivalents français de termes anglais, nous utiliserons le terme de *regroupement* comme double traduction des termes anglais *cluster* et *clustering*.

aussi des coercions liées à des figures de style. Ainsi, les usages métaphoriques et métonymiques présents dans le corpus pourraient être détectés comme levant pareil cas d'ambiguïté : en effet, l'usage des mots concernés ne ressemblera pas à celui des arguments réguliers des prédicats qui leur sont métaphoriquement appliqués, ni aux autres arguments de leurs prédicats de base pour lesquels la métaphore n'est pas observée. De la sorte, nous pouvons espérer que la présence de ces métaphores n'ait pas d'impact sur la structure globale de la hiérarchie ontologique : seules des copies de certains mots désambiguïsés apparaîtraient dans des classes non conventionnelles pour eux, et la nature stylistique de leur usage pourra être déduite par inspection manuelle des résultats. Ainsi, la présence d'éléments inattendus dans la relation U inférée pourra être amortie par les relations régulières que ces mêmes mots entretiennent et par une procédure adaptée d'élimination des ambiguïtés et donc des motifs problématiques de la relation inférée.

Tout ceci démontre par conséquent que nous disposons de réponses potentielles aux problèmes soulevés par notre hypothèse de travail, rendant alors envisageable de tester effectivement cette hypothèse par l'étude pratique d'un corpus. D'un intérêt tout particulier pour cette tâche sont les corpus annotés syntaxiquement, en UD par exemple, car ces annotations fournissent directement les relations de prédications dont nous avons besoin au travers des relations de dépendance : il suffirait alors de sélectionner les catégories syntaxiques et les étiquettes de dépendances que nous souhaiterions considérer, et utiliser les occurrences de ces relations comme matériau de base pour inférer U et construire algorithmiquement une hiérarchie de types ontologiques. Dans le reste de cette section, nous allons nous appliquer à décrire des idées pour une implémentation pratique d'une telle tâche, en soulignant les aspects les plus complexes qu'une telle procédure implique et les problèmes supplémentaires qu'il resterait à résoudre pour la mener à bien.

7.3 Algorithmes et éléments d'implémentation

Grâce à la section précédente, nous avons pu nous faire une vague idée de comment passer d'un corpus à une hiérarchie de types ontologiques, grâce à l'étude des relations de prédication et à l'inférence de la relation de sens associée, permettant par regroupement de reconstruire des A-types et β-types, et la hiérarchie elle-même. Dans la présente section, nous présentons ATHICE, un algorithme expérimental développé en Python et destiné à implémenter cette procédure. Nous introduisons notamment des détails théoriques de l'algorithme souhaité, ses différentes variations possibles et les choix que nous avons faits jusqu'ici pour l'implémentation. Il nous faut cependant souligner que l'implémentation d'ATHICE qui sera discutée dans ces pages est un portrait figé du programme au moment où les présentes lignes sont rédigées, alors que le programme pourra être amené à changer y compris après la finalisation de notre thèse. Nous décrirons donc ici l'implémentation de la version 0.1, datée du 23 juin 2022 et présentant la toute première version fonctionnelle de l'algorithme — limitations et erreurs comprises. [122]

L'état embryonnaire d'ATHICE à ce stade, dû à son implémentation tardive au cours de la présente thèse, ne constitue donc en aucun cas un travail de recherche complètement achevé. Nous estimons cependant que les leçons tirés au cours de la construction de ce programme sont précieuses, dans la mesure où elles ont permis de consolider l'idée générale théorique d'un algorithme dédié à la recherche empirique d'ontologies de types, et ont permis d'identifier au sein de cet algorithme les difficultés les plus importantes et les points délicats sur lesquels toute

122. Notre implémentation est disponible à l'adresse `https://gitlab.crans.org/babonnaud/athice`, et est versionné à l'aide de l'outil GIT. Comme l'on peut s'y attendre pour un programme en cours d'élaboration, il est possible que l'implémentation disponible au moment de la lecture de ces lignes diffère grandement des descriptions qui en sont faites dans le présent chapitre, y compris dans son architecture globale, bien que son objectif de production de hiérarchies de types est prévu pour rester stable dans le temps.

tentative d'implémentation devra porter une attention particulière. Il serait dommageable que l'expérience acquise au cours de l'écriture du programme ainsi qu'au cours des premiers tests soit perdue, alors que la question de la population empirique de l'ontologie des types sémantiques est soulevée par d'autres chercheurs (cf. p. ex. la discussion de Retoré [149] à ce sujet) et que l'algorithme présenté ici dispose du potentiel théorique pour y proposer une réponse. C'est pourquoi nous avons choisi d'évoquer ici notre propre tentative, en commençant par décrire les principes algorithmiques généraux d'ATHICE en sous-section 7.3.1, puis en présentant les choix particuliers de notre implémentation au regard des possibilités multiples théoriquement offertes en sous-section 7.3.2 et les difficultés d'implémentation requérant une attention particulière en sous-section 7.3.3. Les enseignements des tests et des premiers résultats seront abordés dans la section 7.4.

7.3.1 Architecture globale

Commençons par formaliser l'organisation d'un algorithme inspiré des méthodes décrites dans la section précédente. Comme nous l'avons vu, notre premier objectif est d'approximer U, la relation de sens entre les unités lexicales du corpus, sachant qu'une partie de cette relation est directement observable à partir des données annotées et que le reste doit être déduit par des méthodes d'inférence statistique. Nous distinguons dans cette sous-tâche deux étapes essentielles. La première est la *collecte*, ou l'analyse préliminaire des données du corpus afin de rassembler toutes les paires d'unités lexicales en relation de prédication. Avec des annotations comme en UD, cette étape est très rapide : il suffit de décomposer les données pour en extraire les relations de dépendance, et les filtrer en fonction des types de dépendance et des catégories syntaxiques souhaitées ; afin d'offrir une plus grande versatilité dans les résultats et de permettre une complexité graduelle dans la quantité de données considérée, nous paramétrons cette étape par des listes de catégories syntaxiques et de types de dépendances que nous souhaitons conserver, avec une liste minimum comprenant les noms (communs et propres) et les verbes, ainsi que les relations de sujet et d'objet direct. Dans la mesure du possible, il conviendra de privilégier une annotation en syntaxe profonde si disponible, prenant davantage en compte les rôles thématiques dans l'établissement des dépendances ; par exemple en considérant les *Enhanced Universal Dependencies* (EUD) [164]. En plus de mieux restituer la distinction entre sujets de surface et objets profonds, et entre objets de surface et sujets profonds (nécessaire pour éviter les inversions en cas de forme passive par exemple), cette approche a le potentiel d'augmenter la quantité de données disponibles : en EUD, par exemple, on enregistrera davantage de relations de sujet nsubj. Cette étape est également celle où sera réalisée la production des copies des unités lexicales (que nous prendrons sous forme de lemmes) en fonction de leur position argumentale ou prédicative et, pour les prédicats, en fonction du type de dépendance que nous considérons, afin de nous ramener à un ensemble uniquement constitué de prédicats monadiques.

La seconde étape à considérer est l'*inférence*. Comme évoqué plus haut, la relation U obtenue à l'issue de l'étape de collecte est nécessairement partielle et incomplète, puisque toutes les combinaisons possibles des lemmes du corpus ne sont pas observées. De plus, si la collecte n'est basée que sur l'observation des dépendances dans le corpus d'entrée, alors chaque relation observée comprend a priori un prédicat et un argument, faisant du graphe de comparabilité de U à ce stade un graphe biparti. L'inférence à réaliser a donc pour but de compléter U en devinant quelles relations non-observées sont possibles, avec un accent tout particulier mis sur les relations internes à chaque classe de lemmes. Nous réalisons cette étape par regroupement des lemmes en fonction de leur voisinage dans le graphe de comparabilité de U, c.-à-d. en mesurant

la similarité entre les ensembles d'autres lemmes avec lesquels ils sont en relation observée : [123] lorsque ces voisinages sont suffisamment proches, on peut supposer que les lemmes testés ont le même comportement au regard de la prédication, et ajouter une relation U entre eux. Ces similarités doivent être testées entre arguments ou entre prédicats séparément puisque le graphe de comparabilité est biparti, rendant les voisinages des deux classes nécessairement disjoints. Cependant, chaque inférence ne génère pas que des relations internes à une classe : lorsqu'une nouvelle relation est établie, il conviendra de mettre également en relation les prédicats ou arguments qu'ils n'auraient pas déjà en commun dans leurs voisinages respectifs, afin de retrouver les propriétés complète de la relation de Sommers. Le regroupement ainsi effectué produit directement les A-types et β-types dont nous avons besoin pour la suite de l'algorithme. À l'issue de cette étape, nous pouvons donc considérer avoir acquis toutes les paires pertinentes de la relation de sens.

La seconde tâche à effectuer par notre algorithme est alors la reconstitution de l'ontologie des types. Mais là encore, une étape intermédiaire s'avère nécessaire : nous l'appelons la *désambiguïsation*. En effet, comme évoqué à la fin de la section 7.2, il y a fort à parier que les inférences menées à l'étape précédente conduisent à des configurations où peut être observé le motif proscrit par le principe (T3) de Sommers, signalant une ambiguïté à résoudre. Le but de cette étape est donc de sélectionner les lemmes considérés comme ambigus et de les séparer en de multiples copies, chaque copie étant reliée à une partition distincte du voisinage du lemme initial, de sorte à ce que tout motif problématique soit éliminé. Cela implique formellement d'avoir une sous-routine capable de parcourir la relation U inférée pour identifier ces motifs, et une autre capable de décider quels sont les lemmes à séparer, sachant qu'un chemin P_4 induit offre deux candidats potentiels et qu'un cycle C_4 en offre quatre. Enfin, une troisième sous-routine aura pour rôle de réaliser la séparation du lemme choisi au sein de la relation, en s'assurant que les autres propriétés soient maintenues ; en particulier, il est essentiel à la terminaison de l'algorithme que cette sous-routine n'introduise pas un nouveau motif interdit qui n'existait pas dans l'itération précédente, et en élimine au moins une à chaque application. Il est à noter que comme nous disposons à ce stade d'indications sur les A-types et β-types, il est possible de réaliser ces séparations de manière globale sur les regroupement plutôt que sur les lemmes isolés : les lemmes regroupés ensemble présentent en effet le même comportement, y compris au regard de leurs ambiguïtés possibles. En supposant que tout ceci puisse être mené à bien, cette étape conduit alors à une relation qui répond à tous les impératifs de Sommers... sauf peut-être un.

En effet, la répartition des données dans un corpus de texte est telle qu'une seule des hypothèses de Sommers pourrait ne pas être remplie à ce stade : la connexité du graphe de comparabilité de U. Il est envisageable et probablement très commun qu'au sein d'une quantité d'énoncé très limitée il y ait des groupes de lemmes entre lesquels aucune relation n'est observée en pratique. Si cela est le cas, la relation obtenue à l'issue de l'étape de désambiguïsation n'est pas le graphe de comparabilité d'un arbre mais d'une forêt ; conformément aux principes de Sommers, il convient alors d'introduire un A-type vide, représentant la catégorie maximale, et de le relier à tous les autres types obtenus au cours de l'élaboration de la relation. Dès lors, une

123. Une question soulevée par cette idée est de savoir si la fréquence des relations observées doit jouer un rôle ou non dans la mesure de cette similarité. Bien que cette question reste ouverte à discussion, nous sommes d'avis que cette fréquence ne doit pas intervenir. En effet, la relation qui nous intéresse ici est purement binaire, décrivant si oui ou non les mots ont du sens lorsque utilisés conjointement, et la notion de préférence sélectionnelle introduite par la mesure des fréquences ne devrait pas avoir d'influence sur la possibilité de combinaison sensée : une seule observation suffit à prouver que la combinaison est possible. On pourrait arguer que la fréquence pourrait aider à détecter plus tôt les cas de figure de style ; nous objectons cependant que de tels emplois ne sont pas nécessairement plus rares que les autres, et que cela introduirait le risque que des emplois parfaitement valides soit injustement étiquetés comme des erreurs de catégorie à cause de leur faible fréquence dans le corpus d'étude.

Algorithme : ATHICE

Données : un corpus C et deux listes P de parties du discours et L d'étiquettes de relations.
Résultat : une hiérarchie de types sémantiques T.
$U \leftarrow \mathsf{collect}(C, P, L)$
$U \leftarrow \mathsf{infer}(U)$
$U \leftarrow \mathsf{desambiguate}(U)$
$T \leftarrow \mathsf{reconstruct}(U)$

FIGURE 7.1 – Schéma général de l'algorithme ATHICE

dernière étape de *reconstruction* pourra traiter la relation inférée et désambiguïsée comme un graphe trivialement parfait, et en déduire l'arbre hiérarchique des catégories qu'elle engendre, concluant ainsi l'algorithme.

Nous récapitulons alors le schéma général de l'algorithme ATHICE en figure 7.1, restituant les quatre étapes décrites ci-dessus avec les paramètres minimaux attendus. Bien évidemment, il est possible qu'une implémentation pratique de cet algorithme nécessite l'ajout d'hyperparamètres supplémentaires, comme un nombre de regroupements à calculer ou des seuils d'acceptabilité statistique, qui peuvent être ajouté à l'envi ; mais la structure présentée dans cette figure doit être comprise comme le schéma minimal qui subsume toute implémentation.

7.3.2 Choix d'implémentation

Maintenant que nous avons le schéma général de l'algorithme et les principales idées sous-tendant chaque étape, nous pouvons nous pencher plus en détail sur les différentes méthodes d'implémentation de celles-ci, et sur les choix que nous avons faits pour notre propre version, avec quelques justifications idoines. Comme évoqué plus haut, notre version d'ATHICE est developpée en Python 3, et suit une organisation de *package* Python standard. Nous décomposons la suite de cette sous-section en quatre parties portant sur les quatre étapes identifiées précédemment.

Collecte. Nous avons pris le parti d'étudier des corpus annotés en UD pour nos premières expérimentations, aussi nous fallait-il un analyseur d'entrées capable d'extraire des informations organisées selon le format `CoNLL-U` [124] Plutôt que de réécrire entièrement un tel analyseur, nous nous sommes appuyé sur le logiciel GREW [79], dont un portage en Python est disponible. Le recours à GREW a grandement facilité le traitement du corpus d'entrée : en effet, ce programme dédié à la réécriture de graphes pour le TAL est capable de décomposer les données d'entrée en une liste de graphes de dépendances associés à chaque phrase du corpus, et offre la possibilité d'effectuer des requêtes sur des motifs donnés. Grâce à cette fonctionnalité, nous n'avons eu qu'à effectuer une série de requêtes correspondant aux parties du discours et aux types de relations choisies en paramètres de notre programme, et d'accumuler les résultats dans une structure de donnée indépendante — en l'occurrence un dictionnaire, manipulé comme une représentation simple du graphe de comparabilité de la relation U. La standardisation directionnelle des dépendances au sein du schéma d'annotation UD permet au passage de déterminer pour chaque observation quel lemme est prédicat et quel lemme est argument, permettant la séparation à la volée des différentes facettes d'un même lemme. Ces distinctions sont incarnées dans le maintien

124. cf. `https://universaldependencies.org/format.html`.

de deux dictionnaires distincts, l'un associant chaque prédicat à l'ensemble de ses arguments observés, et l'autre faisant l'inverse. Chaque lemme est annoté par sa partie générale du discours, et pour les prédicats par la position argumentale concernée par la copie considérée. Ainsi, un verbe comme *manger* pourra avoir jusqu'à trois représentations : un argument `manger_v` (pour des cas comme « je veux manger »), et deux prédicats `manger_v_nsubj` et `manger_v_obj` distanguant les positions sujet et objet.

Inférence. Pour cette seconde étape, il était nécessaire d'appliquer des algorithmes de regroupement. C'est alors ici que plusieurs possibilités s'offrent à nous, dans le choix de l'algorithme qui effectuera le regroupement d'une part, et dans le choix des données à regrouper d'autre part. Nous avons vu juste au-dessus que, conformément aux hypothèses de la section 7.2, les arguments et les prédicats étaient maintenus séparés dans notre programme : il est donc possible d'effectuer un regroupement indépendant sur chaque ensemble séparément. En pratique, le regroupement des prédicats par exemple est effectué en utilisant les listes d'arguments associées à chacun d'eux par notre dictionnaire comme des vecteurs binaires dans un espace de dimension égale au nombre d'arguments collectés. [125] Dans l'implémentation que nous décrivons, nous avons fait le choix de regrouper uniquement les prédicats, à l'aide des algorithmes proposés par la bibliothèque SCIKIT-LEARN [126] ; les premiers tests ont été effectués avec la méthode classique dite des *k-moyennes* [84], ajoutant de ce fait un hyperparamètre supplémentaire à l'algorithme pour déterminer le nombre de regroupements à calculer.

Pour les arguments, nous avons commencé par former les portées (les α-types) de chaque regroupement de prédicats (un A-type) par l'union des voisins de chaque membre de l'A-type. Ensuite, les β-types ont été obtenus par partition des arguments en fonction de leur A-types antécédents ; autrement dit, si les regroupements de prédicats sont notés $C_1, \ldots, C_n$ et que la relation de prédication est étendue sous la forme $C \blacktriangleright t$ si et seulement si $\forall P \in C.P \blacktriangleright t$, alors les arguments sont partitionnés en classes d'équivalence pour l'égalité de leur image par la fonction $t \mapsto \{C_i \mid C_i \blacktriangleright t\}$. En parallèle, une pré-hiérarchie est formée en reliant deux A-types entre eux lorsque l'on peut estimer leurs portées comme « suffisamment incluses l'une dans l'autre » : en pratique, cela revient à comparer la valeur de la fonction $\mathrm{incl}(C_i, C_j)$ définie par

$$\mathrm{incl}(C_i, C_j) = \frac{|s(C_i) \cap s(C_j)|}{|s(C_i)|} \qquad \text{avec } s(C) = \{t \mid C \blacktriangleright t\}$$

avec une valeur de seuil fixée proche de 1 : lorsque la valeur dépasse le seuil, on considère que C_i est inférieur à C_j dans la hiérarchie. La valeur du seuil est alors ajoutée aux hyperparamètres de l'algorithme. À l'issue de cette partie du programme, la relation U inférée est encodée sous la forme d'un graphe (toujours via un dictionnaire), dont les sommets représentent les regroupements des prédicats et des arguments, et dont les arêtes sont orientées de telle sorte que les regroupements supérieurs de la hiérarchie soient sources et les inférieurs soient cibles. Au sein d'un même regroupement, il est supposé que tous les lemmes sont en relation U entre eux, et lorsqu'une arête existe entre deux, tous leurs lemmes sont en relations entre eux également. Par construction, les β-types forment bien des feuilles, à l'opposé des A-types.

Désambiguïsation. Comme vu précédemment, cette étape fait intervenir trois sous-routines principales. La première est destinée à la détection des motifs interdits par la théorie de Sommers.

125. Nous n'avons pas considéré de méthodes de réduction des dimensions ici, il pourrait cependant être intéressant d'étudier par la suite si l'application d'un tel procédé avant regroupement a une influence positive sur la production des résultats.

126. cf. `https://scikit-learn.org/`.

Comme nous avons représenté U sous la forme d'un graphe, il nous a paru naturel de mobiliser à nouveau GREW pour cette tâche. Comme la recherche de motifs est une opération distincte de la recherche de sous-graphes induits et qu'un cycle C_4 « contient » de ce point de vue quatre P_4 distincts, une simple recherche sur ces derniers suffit à obtenir l'intégralité des données nécessaires à l'évaluation des ambiguïtés de U. La seconde sous-routine est celle qui permet de sélectionner un nœud du graphe à désambiguïser. La démarche est ici factorisée par l'usage des regroupements : plutôt que d'avoir à traiter tous les lemmes similairement ambigus un par un, nous pouvons les traiter en une seule fois ; aussi la recherche des motifs comme la sélection du nœud à traiter portent-elles sur le graphe des regroupements. Cependant, la construction d'une méthode fiable pour choisir un nœud ambigu est loin d'être aisée, puisqu'il s'agit de deviner par observation des contextes quels lemmes d'un même groupe présentent ce type de phénomène alors que plusieurs possibilités s'offrent à nous.

Afin de laisser libre place à l'expérimentation sous toutes ses formes, nous avons pris le parti d'implémenter le choix d'une telle méthode sous la forme d'un hyperparamètre appelé *heuristique*, qui se présente formellement sous la forme d'une fonction prenant en entrée le graphe des regroupements et produisant le nom d'un nœud à traiter. À l'heure où ces lignes sont écrites, seules deux heuristiques ont été implémentées : une première qui choisit systématiquement un nœud au hasard parmi ceux qui apparaissent le plus fréquemment parmi les motifs d'ambiguïtés, et une seconde similaire sauf que le nœud retenu est systématiquement sélectionné parmi les β-types. Évidemment, ces heuristiques sont loin d'être les seules possibles, et ne sont que des essais destinés à vérifier la pertinence d'heuristiques de ce type qui ne dépendent que du graphe d'entrée ; des heuristiques pouvant faire appel à des données extérieures sont également envisageables ici.

La troisième et dernière sous-routine de cette étape est celle qui effectue la séparation proprement dite. Étant donné le nœud sélectionné, elle effectue un regroupement des nœuds dans son voisinage immédiat en fonction des relations qu'ils ont entre eux, différents sens étant caractérisés par des voisinages sans liens directs entre eux. En pratique, considérant que des sens différents peuvent avoir des supertypes communs, l'algorithme qui partitionne le voisinage étudie en priorité les regroupements parents du nœud traité en commençant par ceux présentant le plus faible degré de sortie,[127] et en déduit des ensembles pertinents formés de parents et d'enfants du nœud traité étant tous connectés entre eux ; chaque ensemble ainsi obtenu représente un sens distinct. La sous-routine efface alors le nœud initial et insère un nombre de nouveaux nœuds égal au nombre d'ensembles identifiés, et chacun de ces nouveaux nœud est marqué comme étant une copie du nœud initial (se propagant aux lemmes qu'il contient) et est relié aux élements de l'ensemble qui lui correspond. Un nœud supplémentaire est par ailleurs ajouté lorsque des enfants du nœud initial n'ont pas été récupérés par l'un de ces ensembles. La suite de ces trois sous-routines est alors répétée jusqu'à ce que plus aucun motif problémetique ne soit trouvé, auquel cas l'étape de désambiguïsation est considérée comme terminée.

Reconstruction. Cette dernière partie sera très brève, car la reconstruction de l'arbre à partir du graphe obtenu à l'issue de l'étape précédente, qui est trivialement parfait par construction, est assez simple. On introduit tout d'abord un élément supplémentaire à ce graphe qui se relie à tous les autres s'il n'en existe pas déjà un ; il représente le type maximal. Puis, il suffit d'ordonner les nœuds du graphe par degré de sortie décroissante, car les plus haut degrés sont en effet les plus hauts dans la hiérarchie finale : le nœud de plus haut degré, généralement celui introduit manuellement, devient la racine ; on le note comme tel dans une autre structure et on

127. C.-à-d. le plus petit nombre d'arêtes ayant le regroupement en question pour source.

le supprime du graphe initial. Les nœuds de plus haut degrés dans chacune des composantes connexes résultantes sont marqués comme ses enfants directs, et ainsi de suite récursivement jusqu'à ce que les feuilles soient toutes atteintes. Notre implémentation exécute ceci à l'aide d'une boucle qui vide l'ensemble des nœuds du graphe au fur et à mesure qu'ils sont traités, et renvoie l'arbre final lorsque cet ensemble est vide. Les nœuds de cet arbre sont alors les types ontologiques, et les lemmes associés à chaque regroupement sont alors traités comme recevant la composante ontologique associée.

7.3.3 Difficultés techniques

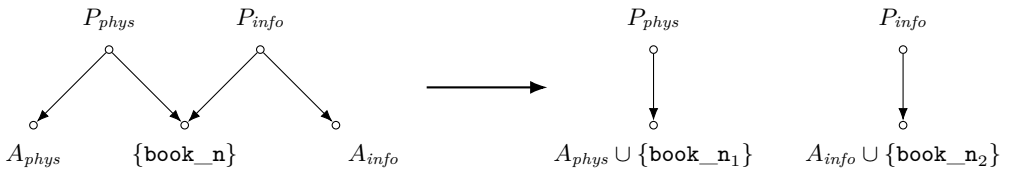
Aussi simple que l'algorithme puisse paraître sur le papier, la description plus détaillée des choix d'implémentation réalisée dans la sous-section précédente indique que des difficultés d'ordre pratique sont inévitables. Nous abordons dans la suite quelques difficultés majeures ainsi rencontrées au cours de notre exercice d'implémentation d'ATHICE, et soulignons les problèmes techniques encore présents dans la version disponible au moment de l'écriture de ces lignes.

Regroupement des lemmes. Comme décrit au-dessus, nous avons choisi par défaut l'algorithme des k-moyennes pour effectuer le regroupement initial des prédicats, et nous nous basons sur ce regroupement pour regrouper les arguments. La difficulté posée par ce choix est la nécessité de déterminer une valeur d'hyperparamètre k pour le nombre de regroupement à produire qui soit pertinente pour nos résultats. Lorsque le paramètre est trop bas, il en résulte l'existence d'un regroupement « défaut » dans lequels s'entassent des prédicats sans grand lien entre eux ; à l'inverse, un nombre trop grand risque de séparer de force des prédicats dont le comportement pourrait être considéré autrement comme similaire. Au vu des premières expériences menées, une heuristique satisfaisante serait de choisir pour k une fraction de petit dénominateur du nombre total de prédicats, de l'ordre du tiers ou du quart ; mais idéalement, l'esprit exploratoire que nous souhaiterions insuffler dans ce projet ATHICE s'épanouïrait davantage d'une méthode de regroupement qui ne reposerait pas sur ce type d'hyperparamètre.

Construction d'heuristiques pertinentes. La détermination d'une heuristique efficace est également un problème épineux pour l'implémentation. Existe-t-il une méthode qui permettrait de sélectionner un nœud ambigu de manière fiable ? Comme nous le verrons dans la section suivante, les premières heuristiques testées donnent des résultats contrastés, et rien ne peut garantir leur légitimité au regard du comportement observable. Il est probable d'ailleurs qu'il soit impossible de deviner correctement un nœud ambigu sans utiliser des informations supplémentaires, a minima depuis le corpus d'entrée que ce soit à l'aide de données issues de l'annotation ou via la prise en compte de relations supplémentaires, voire à l'aide de bases de données externes comme des réseaux lexicaux. Il est indéniable que l'introduction de données externes induirait un biais dans la production de notre résultat, aussi la question porte-t-elle sur le fait de savoir si un tel biais est effectivement souhaitable. Dans une certaine mesure, le choix d'inclure ces données risquerait d'effacer des ambiguïtés propres au corpus d'étude, au profit d'ambiguïtés conventionnellement reconnues — du moins dans les premières étapes. Ce choix dépend alors de l'intention que nous souhaitons placer dans la production de notre résultat : pour concevoir une ontologie de types générale d'une langue donnée, s'appuyer sur des données déjà existantes semble pertinent et peut-être même souhaitable ; mais nous avons également l'intuition qu'une méthode basée uniquement sur les données du corpus pourrait permettre la découverte d'emplois contrastés de mots qui ne sont habituellement pas considérés comme ambigus, ce qui pourrait ouvrir la voie à des applications dans des domaines comme les langues de spécialité et la terminologie.

Dans les deux cas cependant, les contours d'une heuristique fiable demeurent à déterminer : nos premières implémentations ne sont en effet pas suffisamment stables pour cela.

Séparation des nœuds. La procédure de séparation des nœuds sélectionnés comme ambigus est certainement l'une des étapes les plus délicates de l'implémentation d'ATHICE, principalement à cause du grand nombre de situations à prendre en compte lors de l'opération. Un aspect de cette procédure que la description de la sous-section précédente a passé sous silence est le cas où l'on doit fusionner un nœud nouvellement créé avec un nœud déjà existant. À titre d'exemple intuitif, supposons qu'à une étape donnée de l'algorithme il existe un A-type correspondant aux critères d'un type *phys*, et un second correspondant aux critères d'un type *info*, chacun ayant un lien avec sa propre feuille d'arguments dont il est l'unique parent ; et supposons qu'il existe un β-type contenant le lemme `book_n` et relié aux deux A-types précédents. Si nous séparons ce β-type, il ne suffit pas de maintenir les copies comme des feuilles séparés : en tant qu'arguments physique et informationnels valides, les aspects fraîchement identifiés doivent rejoindre les arguments déjà existants ; la transformation à mener est donc la suivante :



Mais la complexité des transformations augmente avec le nombres de nœuds dans le voisinage de celui à traiter. Des A-types peuvent également nécessiter d'être fusionnés avec d'autres A-types lorsque leurs copies ont exactement les mêmes propriétés que ceux-là. Une autre difficulté se pose lorsque, après séparation, un nœud se retrouve comme enfant d'un nœud sans parents dont les seuls enfants sont le nœud fraîchement séparé et son propre enfant, comme dans l'exemple de gauche ci-dessous : dans ce cas, les deux parents devraient également être fusionnés. De plus, la procédure de séparation doit veiller à ne pas produire de configurations où un β-type serait enfant de deux A-types non reliés entre eux comme à droite ci-dessous : en effet, si cette configuration satisfait les règles de Sommers, elle implique que le β-type doit être plus haut dans la hiérarchie finale que les prédicats au-dessus de lui, ce qui n'est absolument pas souhaitable.

Cette diversité de cas particuliers à prendre en compte rend l'écriture d'un algorithme de séparation assez difficile, suffisamment du moins pour multiplier drastiquement les risques d'erreurs dans le programme final. Ainsi, lors des tests réalisés avec la version d'ATHICE décrite dans ces pages, quelques rares cas de lancements ont été observés au cours desquels une séparation a commencé à boucler sur elle-même, en étant incapable d'éliminer le motif problématique parce que le nœud effacé était recréé à l'identique. Les configurations précises qui causent ce phénomène n'ont pas encore été identifiées. En dépit de ce problème, il convient de souligner que la plus grande part des tests ont terminé sans difficulté. Mais ceci illustre bien la quantité de précautions nécessaires à l'élaboration de cette procédure, qui est assurément le point névralgique d'ATHICE.

7.4 Premiers résultats

Malgré les difficultés d'implémentation et le déroulement tardif de ces expériences au regard de la durée de la thèse défendue dans ces pages, nous avons été en mesure de proposer une version d'ATHICE suffisamment complète pour mener à bien une poignée de tests en conditions réelles, offrant par conséquent un premier aperçu des capacités du programme, mais également de ces limites actuelles. Cette dernière section a ainsi pour but d'illustrer les premiers résultats obtenus, de discuter leur pertinence et surtout d'établir les pistes d'amélorations potentielles et les problèmes inhérents à l'approche que nous avons déployée dans ce chapitre.

7.4.1 Résultats des premiers tests

La finalisation tardive de la première version fonctionnelle d'ATHICE n'a laissé que peu de temps pour des tests poussés, mais n'a pas empêché les premières manipulations d'aboutir à des résultats tantôt prometteurs et tantôt mitigés, dont nous offrons ici un bref aperçu. L'entrée utilisée pour ces tests est le sous-corpus de test (laissant de côté ceux de développement et d'entraînement) du *Universal Dependencies English Web Treebank* (UD-EWT), version 2.9.[128] Les paramètres de filtre des relations pertinentes se sont concentrés sur les données suivantes : pour les parties du discours, les noms (NOUN) et les verbes (VERB) uniquement, et pour les relations, les étiquettes sujet (nsubj) et objet (obj) uniquement. Pour rappel, les autres hyperparamètres ajoutés par notre implémentation sont le nombre $n \in \mathbb{N}^*$ de regroupements à calculer pour les prédicats, le seuil minimal $\sigma \in [0, 1]$ de reconnaissance d'inclusions entre portées, ainsi que l'heuristique h permettant de sélectionner un nœud à séparer étant donné un graphe en entrée, cette heuristique variant entre deux valeurs `MC` (« most critical », le nœud le plus fréquent au sein des motifs procrits), et `MCAO` (« most critical, arguments only », idem mais limité aux arguments) qui sont les seules implémentées pour cette version 0.1 d'ATHICE.

Malheureusement, la manipulation de ces hyperparamètres n'a eu que peu d'impact sur la forme finale des résultats. Quels que soient les valeurs prises en compte, les arbres ontologiques finaux obtenus ont été invariablement plats, c.-à-d. de hauteur 1, distribuant de nombreux petits types comme enfants d'un type maximal vide. Comme on pouvait s'y attendre, de faibles valeurs du seuil σ conduisent des comportements non-désirés, du fait que cela peut conduire à reconnaître deux A-types distincts comme mutuellement supérieurs l'un à l'autre, ce qui rompt avec les règles de Sommers. Par ailleurs, les défauts actuels de la procédure de séparation des nœuds provoquent de plus en plus d'erreurs à mesure que le paramètre n est augmenté, remettant à un certain degré de hasard l'obtention de résultats exploitables pour $n > 100$. De même, la valeur $h =$ `MCAO` augmente significativement le taux d'erreur, principalement parce que la condition interdisant d'avoir deux A-types non reliés au-dessus d'un même β-type ne parvient pas à être maintenue. Nous gageons néanmoins que les résultats utilisant cette heuristique sont fondamentalement moins pertinents, puisqu'elle implique qu'aucune ambiguïté ne pourra être reconnue au sein des prédicats. Les tests ayant terminé sans erreur ont permis de dégager deux tendances comportementales notables de l'algorithme : pour des valeurs de n entre 20 et 100, le nombre de séparations effectuées est généralement très proche n, et pour les valeurs testées entre 100 et 150, ce nombre tend à stagner autour des 100 ; et de plus, l'heuristique `MC` a tendance à ne sélectionner que des prédicats au tout début de l'étape de désambiguïsation, mais à ne sélectionner plus que des arguments vers la fin, avec au milieu un certain équilibre entre les deux sortes de lemmes.

128. `https://universaldependencies.org/treebanks/en_ewt/index.html`.

Quant aux types eux-mêmes, nous proposons dans la suite quelques exemples obtenus pour $n = 80$, $\sigma = 0.85$ et $h = \texttt{MC}$.[129] Nous passons outre les types générés à partir du regroupement « défaut » mentionné en sous-section 7.3.3, dont le contenu est malheureusement trop vaste et trop disparate pour lui donner du sens. Des exemples notables de types obtenus sont listés et commentés ci-après :

— **{recommend_v_obj, place_n, service_n, office_n}** : un exemple simple et cohérent, les trois noms étant acceptables comme arguments objets du prédicats. On notera cependant que contrairement aux deux autres, *service* n'est pas utilisable comme lieu.

— **{love_v_obj, champagne_n, atmosphere_n, place_n, guy_n, fact_n}** : un exemple simple et cohérent à première vue ; on notera cependant que les arguments sont indifférenciés entre physiques et abstraits.

— **{do_v_nsubj, try_v_nsubj, assume_v_nsubj, take_v_nsubj, cure_v_obj, cook_v_nsubj, fix_v_nsubj, rally_n, university_n,people_n, guy_n, leader_n}** : un exemple typique de groupe présentant une certaine cohérence, bien que certaines combinaisons interrogent, comme ?*cure university*. On relèvera que *university* semble être pris sous l'aspect des personnes qui l'administrent ; aucune autre occurrence de ce mot n'apparaît dans nos résultats.

— **{fight_v_obj, view_v_obj, brew_v_nsubj, terrorism_n, war_n}** : on observe dans ce groupe un cas d'usage métaphorique du prédicat *brew* ; il n'apparaît qu'une seule fois dans le corpus d'entrée en combinaison avec *war*.

— **{lift_v_obj, change_v_obj, raise_v_obj, shake_v_obj, hand_n, worth_n, question_n, capital_n}** : la présence de *hand* dans ce groupe détone avec les autres noms présents, mais s'explique par des usages figuratifs comme *lift a hand* ; en revanche, on constatera qu'aucune distinction n'a été faite entre *raise a hand* et *raise a question*.

— **{give_v_obj, call_n}** et **{make_v_obj, call_n}** : deux exemples qui démontrent que certains types sont trop décomposés par rapport à ce qu'on pourrait attendre. Il semblerait que les deux verbes présents ici aient été initialement classés dans deux groupes séparés.

Nous passons sous silence d'autres types qui n'apportent rien de très différent des exemples ci-dessus. Mentionnons cependant que outre les types démesurés suite à leur combinaision avec le regroupement « défaut », de nombreux cas de types à deux éléments — un prédicat et un argument — apparaissent dans nos résultats. Ceux-ci s'expliquent généralement par le fait que toutes les occurrences de ces mots dans le corpus étudié sont systématiquement combinées entre elles.

7.4.2 Leçons tirées de l'expérience

Les résultats présentés ci-dessus sont certainement un peu maigres au vu de nos objectifs initiaux, mais nous estimons néanmoins que l'expérience déjà acquise nous apporte des enseignements précieux sur les limites et les perspectives de la tâche que nous avons humblement essayé d'accomplir. Il convient avant tout de souligner qu'une grande partie des difficultés relevées quant à l'obtention de résultats satisfaisants est imputable aux imperfections de notre implémentation dans la version 0.1 d'ATHICE, ainsi qu'à certains choix sous-tendant cette dernière, à l'image des problématiques liées à l'algorithme de regroupement. Le cadre de ce travail

129. Notons qu'en conséquence des variables aléatoires induites par l'algorithme de regroupement, les résultats présentés ici pour les mêmes paramètres ne sont pas totalement reproductibles. Nous avons néanmoins constaté une certaine constance dans la majeure partie des types obtenus d'une itération à l'autre, ce qui assurera dans les mêmes conditions des résultats comparables aux nôtres.

de thèse n'a pas permis de présenter une version plus complète dans ces pages ; néanmoins, il est certain que des travaux futurs sauront contribuer à améliorer cette implémentation, et par conséquent la qualité de ses résultats.

Mais au-delà des problèmes pratiques causés par notre implémentation, la persistence de certaines caractéristiques dans nos résultats en dépit des modifications dans les hyperparamètres leur suggère une origine indépendante, et probablement inhérente à l'approche elle-même. Et nous avons tout particulièrement en tête le fait que nous obtenons systématiquement des arbres plats, et des types parfois réduits à deux lemmes isolés : ces phénomènes mettent en lumière le fait qu'en dépit des inférences qui sont faites pour tenter de compléter la relation U, les données récoltées sont tout de même insuffisantes pour obtenir un fragment d'ontologie qui ait une structure intéressante. En effet, la platitude observée témoigne de l'absence de prédicats « dominants », dont la quantité de relations inférée dépasserait la moyenne et les placerait donc plus haut dans la hiérarchie, tandis que les types isolés sont une preuve indéniable du manque de relations observables pour certains lemmes du corpus. Mais nous ne pouvons être surpris que seule une quantité limitée de données puisse être disponible dans un corpus quel qu'il soit ; en conséquence, nos résultats révèlent — ou plutôt confirment, car ce n'est pas réellement une surprise en soi — que la tâche de construire une ontologie de types reflétant le comportement ne serait-ce que d'un fragment d'une langue requiert une quantité faramineuse de données, qu'il n'est pas nécessairement aisé de rassembler.

En particulier, il faudrait au minimum un corpus présentant une très forte diversité d'usage des mêmes mots dans des combinaisons variés ; mais une telle ambition se heurte aux collocations et autres préférences d'usage qui modèlent la langue courante : si deux mots peuvent être théorisés comme ayant du sens dans une utilisation conjointe, il est quelquefois possible qu'une telle utilisation ne soit jamais observée parce que conceptuellement trop rare. Le point critique de ce phénomène est à trouver dans la distinction sommersienne entre fausseté et absurdité, et entre négation et démenti : les occurrences de prédications fausses, ou de démentis catégoriellement corrects, sont loin d'être aussi fréquentes que les prédications affirmatives ; d'aucuns pourraient même s'avancer à dire qu'elles sont aussi rares que les erreurs de catégories, si ce n'est plus puisque les figures de style semblent somme toute assez fréquentes dans les corpus. À titre d'exemple, quand bien même *chien* et *rugir* ne formeraient pas une erreur de catégorie, la probabilité de rencontrer cette combinaison précise dans un corpus paraît intuitivement faible ; or, même si ces deux mots apparaissent dans un corpus donné, il n'est pas même dit que les relations observables suffiront à inférer cette relation théorique. Les résultats obtenus à l'issue de la procédure, à moins d'une intervention manuelle, ne satisferont certainement jamais pleinement notre intuition. Mais s'ils peuvent tout du moins fournir une base conséquente de travail pour la réalisation d'une ébauche d'ontologie de types, cela devrait en valoir la peine.

Profitons de ce moment pour évoquer une autre caractéristique d'ATHICE qui pourra être améliorée à l'avenir : la non incrémentalité de ses résultats ; autrement dit, l'impossibilité d'enrichir les résultats obtenus sur un corpus avec ceux obtenus sur un autre corpus distinct. En effet, il n'est pas aisé de fusionner deux ontologies lorsque celles-ci sont basées sur des relations U qui n'ont pas été inférées de la même manière, et dépendent des fois de lemmes observés d'un côté et qui n'apparaissent pas de l'autre. Par ailleurs, le mélange de résultats finaux ne tiendrait pas compte des variations de similarité entre portées qui pourraient être engendrées par la fusion des données initiales ; de fait, le meilleur moyen de cumuler les données des deux corpus est pour l'instant de reprendre l'intégralité de l'algorithme à partir de l'étape d'inférence, en se basant sur l'union des observations de chaque source. La qualité des résultats finaux pourrait être améliorée en trouvant un moyen de réutiliser l'ontologie produite sur un premier corpus pour en traiter un second : de futurs travaux sur ATHICE pourront être orientés dans cette direction.

Un autre point d'amélioration à mentionner est la détermination des nœuds ambigus, qui non content d'être un problème d'implémentation, peut plus largement être considéré comme une limite à l'algorithme général tant qu'une solution satisfaisante n'est pas trouvée. Comme nous l'avons évoqué dans la section précédente, il n'existe a priori pas de méthode fiable permettant de discerner des lemmes ambigus sur la base de la relation U seule, et cela n'est d'ailleurs pas souhaitable puisqu'une composante linguistique se doit d'entrer d'une manière ou d'une autre dans cette décision. Qu'il s'agisse d'exploiter les annotations du corpus d'entrée ou de faire appel à une base de donnée externe, il convient cependant de rester prudent dans l'utilisation de ces informations. Si l'on dispose par exemple d'une liste des différents sens lexicographiques d'un lemme, il pourrait être tentant de s'en servir directement pour construire les copies subséquentes ; cependant, nous avons établi en sous-section 3.3.3 que les consituants d'un type complexes sont toujours de composantes ontologiques incomparable, ce qui n'est pas toujours le cas des différents sens d'un mots, certains pouvant en effet partager la même composante. Par conséquent, la détection des lemmes ambigus peut s'appuyer sur des données externes, mais doit néanmoins se reposer sur les données internes disponibles pour la séparation proprement dite. Une étude approfondie sera nécessaire pour résoudre ce problème.

Nous terminerons par l'évocation d'une ultime difficulté, et non des moindres, qui concerne l'exploitation des potentiels résultats fournis par ATHICE. En effet, il est commun dans les tâches en TAL comme dans beaucoup d'autres domaines d'appuyer la validité des résultats obtenus par le biais d'une évaluation, comparant par exemple les données produites à des exemples considérés comme des objectifs standard. Seulement, ATHICE a ici pour but de mener une tâche qui relève davantage de l'entreprise exploratoire : notre objectif serait en effet de proposer un standard d'ontologie de types sémantiques là où il n'en existe pas encore, puisque seules des hiérarchies fragmentaires et des exemples issus de considérations intuitives mais elles-mêmes non vérifiées sont actuellement disponibles. Mais l'absence d'ontologie standard pose réciproquement la question de l'évaluation de la pertinence des propositions de hiérarchies, qu'elles soient produites par des moyens humains ou automatiques. Comment peut-on confirmer qu'une telle ontologie s'approche effectivement d'un modèle idéal de la langue ? Est-il même possible de proposer une telle procédure de vérification ? Nous n'avons pour l'instant pas de réponse concrète à apporter à cette question, mais nous tenons à souligner à quel point sa résolution est importante pour la poursuite des travaux en élaboration d'ontologie de types : même si un algorithme plus performant parvenait à réaliser l'objectif d'ATHICE, ses résultats ne pourraient être sérieusement considérés en l'absence d'une évaluation reconnue. Là encore, de futurs travaux sauront peut-être y apporter un réponse satisfaisante.

Bilan. Que retenir des expériences réalisées avec ATHICE ? La tâche de construire une ontologie de types n'est pas une mince affaire, y compris avec des éléments théoriques adaptables à une utilisation empirique, des corpus annotés disponibles et des outils d'implémentation qui facilitent notre tâche. En particulier, outre les difficultés techniques posées par l'implémentation même de l'algorithme, celui-ci requiert par nature une énorme quantité de données pour pouvoir espérer approcher l'organisation concrète de la langue, et en déterminer les types ontologiques réels. En l'absence d'une densité suffisante de combinaisons observables, il existe un risque inhérent à notre objectif et à notre algorithme de révéler une structure trop incomplète pour satisfaire nos ambitions au regard de la langue entière. Nous avons cependant l'intuition que même sur un domaine restreint comme celui d'un petit corpus, les résultats produits par ATHICE peuvent servir des ambitions plus locales, comme étudier la langue spécifique à ce corpus en découvrant

de possibles ambiguïtés internes qui lui seraient propres, et en formant des catégories de termes offrant des indications potentielles sur la manière dont la thématique traitée par le texte est conceptuellement organisée par ses auteurs. Nous conclurons cette discussion par une brève comparaison avec les autres tâches sémantiques menées sur corpus : le fait que nous nous limitons à un ensemble de relations de dépendance qui ont du sens linguistiquement parlant plutôt qu'à des contextes plus généraux pour déterminer les vecteurs représentant les mots et mener nos regroupements distinguent ATHICE de bon nombre d'autres algorithmes ; et cette distinction est importante car nous avons notamment l'intuition que cette restriction nous permet d'effectuer des regroupements moins précis, et donc plus adaptés aux caractéristiques propres aux types sémantiques qui, comme expliqué dans notre partie I, forment la plus petite condition nécessaire à toute autre propriété sémantique.

Conclusion

So long, and thanks for all the ghoti!

Anonyme

La présente thèse s'est donnée pour objectif de fournir des éléments fondamentaux nécessaires à l'appréhension et à la construction de théories de types spécifiquement conçues pour l'analyse sémantique formelle. Comme mis en évidence dans la partie préliminaire, la construction de ces théories se repose préalablement sur la compréhension de la notion de même de type sémantique, dans ce qu'elle rassemble de propriétés linguistiques et dans les comportements dont elle fait preuve et que nos théories se doivent de modéliser. Sur la base de ce constat, nous avons divisé notre problématique en trois axes distincts, chacun étant abordé dans l'une des trois parties principales qui constituent cette thèse.

Dans la partie I, nous avons ainsi exploré la notion de type sémantique selon un point de vue linguistique : en examinant les différentes contributions historiques qui ont pu aboutir à l'usage de types comme observé dans les formalismes actuels, nous avons cherché à concevoir une théorie générale des types sémantiques décrivant d'où provient cette notion, à quelles propriétés elle renvoie, et à quel comportement on doit s'attendre au sein du langage. La partie II avait quant à elle pour but de replacer les types sémantiques dans un cadre mathématique et formel. En exploitant le domaine de la théorie des catégories et plus spécifiquement la théorie des topos, choisis notamment pour leur capacité à agir comme modèles des théories d'ordre supérieur, nous avons essayé de construire un schéma général des théories de types sémantiques capable d'offrir une grande flexibilité de composition, répondant ainsi aux impératifs issus des observations linguistiques. Enfin, la partie III était dédiée à l'examen des fondations concrètes de telles théories, en fournissant des pistes pour répondre à la question de savoir quels types de base sont réellement nécessaires à une modélisation fidèle de la langue ; les pistes explorées s'appuyant sur la recherche de données issues de corpus de textes.

Contributions

Nous résumons ici les différentes contributions de la présente thèse, suivant l'ordre des parties. En dépit de la diversité des axes abordés, nous réaffirmons ici que la structure de nos travaux est cohérente : son cœur est bien entendu la construction de théories de types formelles, mais un tel travail ne saurait être adéquatement réalisé dans une optique linguistique sans une compréhension profondes des phénomènes linguistiques à l'œuvre, ni ne saurait être complet sans indication de comment instancier une théorie concrète utilisable dans des applications pratiques d'analyse sémantique, qui est bel et bien l'objectif général de la linguistique informatique auquel la présente thèse espère avoir contribué.

Axe linguistique. Nous avons exploré dans le chapitre 2 diverses contributions historiques à la sémantique dont des fragments apparaissent comme ayant enrichi notre perception commune des types sémantiques. Si nous avons pu retracer la première construction de cette notion à la théorie des types de Russell, dont l'objectif premier était la prévention des paradoxes logiques dans la langue, les nombreuses critiques et idées émises à partir de cette construction ont été un terreau fertile sur lequel diverses variations théoriques ont vu le jour. Nous retenons en particulier la nécessité de restreindre les domaines d'entités pour une meilleure précision sélectionnelle, dont nous retrouvons les prémisses dans les travaux de Turing mais qui trouvent leur véritable essor dans les apports de la sémantique structuraliste et générativiste, sous l'influence notable de Katz et Fodor, entre autres. Mais la caractérisation des types la plus pertinente pour nos exigences nous a semblé être la théorie des catégories ontologiques initiée par Ryle et surtout perfectionnée par Sommers.

Nous estimons en effet que la théorie sommersienne des types fournit un comportement satisfaisant des types sémantiques au regard de la compositionnalité et surtout des problèmes de coercions et d'erreurs sémantiques qui peuvent être observés dans la langue. D'une importance majeure ici est l'idée d'une construction des types basée non pas sur ce qui est *vrai* pour les prédicats, mais sur ce qui est *non-absurde*, c.-à-d. vrai ou faux d'une manière sémantiquement cohérente. Étant admise le peu d'impact observable des travaux de Sommers dans la littérature autour de la sémantique formelle, une première contribution de la présente thèse est donc d'avoir remis en lumière une perspective méconnue sur la conception des types sémantiques, qui s'accorde parfaitement avec les exigences établies au fil du temps pour les théories qui y font appel. Par conséquent, si la théorie de Sommers n'est pas en elle-même nouvelle, son application à un cadre sémantique formel nous paraît novateur et contribue à fonder linguistiquement les modèles que nous avons développé par la suite.

Dans le chapitre 3, nous avons examiné plus en détail les liens que peuvent entretenir les diverses propriétés sémantiques définissables dans la langue avec les types sémantiques, et avons établi sur la base de nos observations la nécessité d'un typage correct avant toute autre considération d'ordre sémantique, ces dernières ne pouvant être raisonnablement construites sans vérification préalable des types sémantiques. En élaborant à partir de cette remarque, nous avons construit et proposé un ensemble de principes destinés à servir de fondements généraux pour les théories de types sémantiques. Ces principes, qui postulent l'existence des types sémantiques en tant que valeurs sémantiques hypothétiques permettant de modéliser certains comportements compositionnels de la langue et décrivent leurs principales propriétés — parmi lesquelles la distinction entre composantes ontologiques et composantes structurelles, ainsi que l'existence de complexes hétérotypiques —, constituent le second apport majeur de la présente thèse selon son axe linguistique ; leur objectif est de fournir un cadre scientifique adaptable à toute approche des types sémantique et qui permet de faire le pont entre les observations linguistiques et les modèles mathématiques.

Axe formel. La seconde partie avait pour but de mettre en application les principes énoncés à la fin de la partie linguistique. Dans le chapitre 5, nous avons mis en évidence le lien entre théories des types et catégories, soulignant notamment la relation entre les théories d'ordre supérieur, incluant la théorie des types simples utilisée initialement par Montague, et les topos. Alors que notre introduction à cette relation se fondait sur la théorie pour en dériver un topos, nous avons alors entrepris d'inverser cette approche en fixant les propriétés d'un topos bien choisi, et en étudiant comment ses propriétés pouvaient mener naturellement à une théorie des types sémantiques qui corresponde à nos attentes. À la fin du chapitre 5, nous avons ainsi

posé les bases de cette construction en montrant comment la composante ontologique des types sémantiques pouvait être étendue en un topos plus général, dans lequel pourront ensuite être ajoutés les autres propriétés nécessaires au respect des principes linguistiques susmentionnés.

Mais le véritable cœur de notre développement est le chapitre 6, qui propose à partir de cette base initiale la construction d'une théorie de type générale nommée CPTT, obéissant aux principes linguistiques précédemment énoncés et présentant une grande flexiilité compositionnelle pour les applications en sémantique formelle. CPTT présente des caractéristiques qui la distingue de d'autres approches basées sur les types sémantiques comme UTT, GL, TCL ou MGL, à commencer par le remplacement des types flèches associés aux lambda-abstractions par des types prédicats représentés par un nouveau constructeur $\mathcal{P}$ afin de mettre en évidence le fait que les fonctions considérées dans les applications sémantiques formelles sont quasi-systématiquement à valeurs dans le type des propositions t. Par ailleurs, ce type des prédicats nous a aussi permis de revisiter la notion de sous-typage, en présentant une version covariante des transformations ontologiques de types qui parvient néanmoins à préserver la sûreté des compositions grâce à des règles de typages spécifiquement ajustées à cet effet.

Il convient cependant de souligner que, par le minimalisme qui entoure sa conception et par la grande flexibilité des compositions qu'elle autorise, CPTT n'a pas pour vocation d'offrir une enième théorie alternative pour la sémantique formelle, mais plutôt un exemple de théorie minimale et générale, capable d'unifier les différentes théories évoquées ci-dessus en dépit de leurs différences manifestes. En particulier, CPTT offre une expressivité logique équivalente à celle d'UTT puisque les deux admettent des topos pour modèles, tout en permettant la manipulation de prédicats comme dans les systèmes d'inspiration montagovienne, sans pour autant entrer en conflit avec la gestion du sous-typage ; de plus, CPTT repose également sur un lexique du style de GL et offre une grande capacité de gestion de coercions de types. Par conséquent, une autre grande contribution de la présente thèse est la définition d'une théorie de types dans un style montagovien, incluant la gestion des coercions de types ontologiques, du sous-typage, de la polysémie et autres phénomènes hétérotypiques, ainsi que des changements de types introduits par Partee, et possiblement capable de rivaliser en expressivité avec des théories de types non-montagoviennes.

Corollairement à cette précédente contribution, il convient de souligner un autre aspect auquel la présente thèse participe : nous avons en effet proposé dans les dernières sections du chapitre 6 une variante de CPTT reposant sur un principe inspiré du polymorphisme borné, qui lui offre la capacité d'inférer des coercions et des composés hétérotypiques à partir des types des constantes considérés et de la manière dont elles se combinent. Cette variante permet donc de générer des dérivations de types en CPTT en automatisant le calcul des coercions à utiliser, ouvrant la voie à une implémentation concrète de la théorie au sein d'analyseurs sémantiques. Rappelons d'ailleurs que les coercions explicites qui sont laissées par CPTT dans les formules logiques finales sont pensées comme des indicateurs pour les analyses sémantiques subséquentes sur les changements sémantiques induits par la composition. Ce calcul étendu avec inférence de coercions et d'hétérotypes constitue ainsi un apport supplémentaire de la présente thèse, renforçant la théorie de type précédemment discutée.

Axe empirique. La troisième et dernière partie de notre travail, constituée de l'unique chapitre 7, était dédiée à l'étude de la population de l'ontologie de types sur une base empirique. Nous avons dans ce but exploité la théorie de Sommers pour proposer un algorithme d'exploration de corpus annoté en sémantique, plus particulièrement en UD. Si malheureusement l'ampleur de la tâche s'est révélée bien plus importante que ce que le temps imparti pour notre

travail ne nous a permis, compte tenu des autres contributions que nous y avons développées, nous estimons tout de même que la présente thèse a contribué à ouvrir la voie à de nouvelles questions de recherche autour de la constitution de hiérarchies de types sémantiques. En dépit de l'absence de réponse concrète à ces questions au terme de notre étude, les difficultés mises en évidence au cours de l'élaboration et de la mise en œuvre d'ATHICE sont des sources précieuses d'informations sur les limites et écueils d'une telle entreprise, informations qui pourront être prises en compte par des futures tentatives d'exploration. Parmi les écueils les plus importants à retenir, on notera en particulier le problème de la quantité de données nécessaires à la construction d'un modèle fidèle au comportement de la langue, malgré l'intégration d'éléments d'inférence au sein de notre algorithme, ainsi que le problème de l'évaluation des résultats, dans la mesure où aucun standard n'existe encore dans ce domaine.

Perspectives et travaux futurs

La présente thèse, bien qu'apportant un certain nombre de contributions autour de la notion de type sémantique, est loin de répondre à toutes les questions possibles ; nous détaillons ci-dessous plusieurs pistes supplémentaires qui s'incrivent dans la continuité du travail présenté dans ces pages et qui mériteraient de s'y pencher davantage.

Acquisition empirique d'ontologies. Au vu des conclusions de la partie III, il est évident qu'il reste beaucoup à faire pour espérer obtenir des hiérarchies de type à partir de corpus de textes. Le travail restant peut se diviser en deux tâches principales, portant sur des niveaux conceptuels différents. La première tâche s'intéresse au problème le plus concret, celui de l'obtention des résultats : comment corriger les problèmes d'implémentation d'ATHICE, ou comment construire un algorithme différent et plus efficace, et comment surmonter la difficulté posée par la quantité de données nécessaire à la constitution de résultats fiables ? Quant à la seconde tâche, elle a pour dessein d'interroger la finalité et la validité des résultats : comment évaluer les ontologies produites par les algorithmes, comment confirmer ou infirmer qu'ils constituent effectivement de bons modèles du comportement de la langue naturelle ? Ce double travail, bien que conséquent, serait un complément nécessaire à la présente thèse puisque, comme nous avons pu le souligner à plusieurs reprises dans nos pages, la question de l'instanciation précise de l'ontologie de types porte en elle la majorité des propriétés qui font de CPTT un bon ou un mauvais modèle : en dépit de tous ses constituants formels (constructeurs de types, règles de typage, etc.), cette théorie de types ne saura être pertinente si son ontologie n'est pas adaptée ; c'est pourquoi la question de l'acquisition empirique demeure importante à approfondir.

Intégration aux grammaires formelles. Nous avons terminé la partie II sur un exemple général, démontrant toute l'étendue des capacités de CPTT et de sa variante inférentielle pour la dérivation de formules sémantiques logiques avec coercions. Cet exemple a mobilisé plusieurs transformations introduites de façon ad hoc pour convertir des représentations intermédiaires, faisant le pont entre la représentation syntaxique initiale et la représentation sémantique finale. Il n'est certainement pas un hasard que ces différentes étapes de transformation renvoient au schéma général du système d'interface syntaxe-sémantique que nous avions décrit en section 1.3 : nous pensons en effet que cet exemple pourrait être généralisé en une procédure formelle d'interface syntaxe-sémantique dont CPTT serait l'un des rouages. L'appel à des objets proches d'homomorphismes nous laisse à penser qu'il serait possible en particulier d'intégrer CPTT dans une variante de grammaire catégorielle abstraite où le lexique appliquerait les principes

d'inférence permettant l'insertion des coercions dans la formule objet. Il serait alors nécessaire de décrire les écarts de cette variante d'ACG avec la version originale, d'étudier ses propriétés et de vérifier si une implémentation dans un cadre similaire à celui de l'outil ACGtk [133] est possible. Cette perspective permettrait alors l'automatisation de l'utilisation de CPTT, dont l'ontologie de types serait le principal paramètre.

Comparaison de CPTT et d'UTT. Comme explicité dans le chapitre 6, CPTT est une théorie de types ancrée dans la tradition montagovienne où les noms sont interprétés comme des prédicats, quand bien même notre conception des prédicats avec un constructeur de type à part diverge des standards de cette tradition. Cette approche est donc elle-même opposée à l'alternative de l'interprétation des noms comme types défendue par Luo et Chatzikyriakidis pour UTT. Pourtant, les deux théories reposent sur une logique d'ordre supérieur et admettent par conséquent des topos pour modèles catégoriques. Par ailleurs, la construction de CPTT exclut cette dernière des critiques sur l'interaction entre sous-typage et prédicats, laissant par conséquent ouverte la possibilité que les pouvoirs expressifs de CPTT et d'UTT soient similaires ; en particulier, il pourrait être envisagé l'existence d'une équivalence entre jugements dérivables dans UTT et jugements dérivables en CPTT avec inférence de coercions. Explorer cette possibilité pourrait ainsi permettre de renforcer les deux types d'approche en mettant en évidence leurs points communs, et en rendant accessible dans chacun les constructions plus aisées dans l'autre : cela pourrait être l'étape finale de l'unification des théories sémantiques dont nous souhaitons faire de CPTT le porte-étendard.

Annexe A

Récapitulatif des règles de CPTT

Le chapitre 6 propose une construction étape par étape d'une théorie générale des types nommée CPTT, comprenant un sous-typage covariant et un constructeur de types prédicats. La présente annexe a pour objectif de rassembler en un seul et même endroit les divers éléments constitutifs de la théorie distribués au fil des sections, afin d'en fournir un récapitulatif complet.

A.1 Types

Soit $(\mathbb{B}, \leq, e)$ une ontologie de types, et soit $(\mathbb{H}, \mathfrak{a})$ un ensemble d'hétérotypes sur $\mathbb{B}$. Le système $\mathbb{T}$ des types de CPTT est défini par la grammaire suivante :

$$\mathbb{T} ::= 1 \mid t \mid \mathbb{B} \mid \mathbb{H} \mid \mathbb{T} \times \mathbb{T} \mid \mathcal{P}\mathbb{T}$$

Pour les coercions, nous disposons de deux notations de typage supplémentaires, avec le symbole $\leq$ pour les coercions de sous-typage et le symbole $\to$ pour les autres coercions. Ces notations sont construites sur $\mathbb{T}$ mais n'en font pas partie, et ne doivent en conséquence pas servir à typer des termes. Par souci de lisibilité, nous appellerons $\mathbb{K}$ cet ensemble des types de coercions, donné comme suit :

$$\mathbb{K} ::= \mathbb{T} < \mathbb{T} \mid \mathbb{T} \to \mathbb{T}$$

A.2 Coercions

Nous supposons l'existence d'un ensemble K de symboles de coercion isomorphe à la restriction stricte $<$ de l'ordre $\leq$ vu comme un sous-ensemble de $\mathbb{B} \times \mathbb{B}$, équipé de deux fonctions $\kappa_1, \kappa_2 : K \to \mathbb{B}$ représentant les fonctions de domaine et codomaine, respectivement. L'ensemble des coercions de sous-typages est alors $\mathcal{K}$ défini par la grammaire à gauche ci-dessous. On dispose de même d'un ensemble de coercions hétérotypiques H défini selon la grammaire à gauche.

$$\mathcal{K} ::= K \mid \mathrm{id}_{\mathbb{B}} \mid \mathcal{K} \times \mathcal{K} \mid \exists\mathcal{K} \qquad\qquad H ::= p_{\mathbb{H},\mathbb{B}} \mid \mathrm{id}_{\mathbb{B}} \mid H \times H \mid \exists H$$

On dispose aussi d'un ensemble Z de coercions inverses, formé à partir des coercions de $\mathcal{K}$ et H selon la grammaire donnée ci-dessous. On rappelle que pour tout $c \in \mathcal{K} \cup H$, si c est de la forme $\exists c'$, alors $\bar{c}$ est la coercion $\mathcal{P}c'$ correspondante dans Z.

$$Z ::= \mathcal{P}\mathcal{K} \mid \mathcal{P}H$$

Enfin, on supposera l'existence d'un ensemble S de coercions transstructurelles équipé de deux fonctions $s_1, s_2 : S \to \mathbb{B}$ de domaine et codomaine, comprenant au moins les données suivantes pour tout $\sigma \in \mathbb{T}$:

ident_σ	$s_1(\mathsf{ident}_\sigma) = \sigma$	$s_2(\mathsf{ident}_\sigma) = \mathcal{P}\sigma$
lift_σ	$s_1(\mathsf{lift}_\sigma) = \sigma$	$s_2(\mathsf{lift}_\sigma) = \mathcal{P}\mathcal{P}\sigma$
$\emptyset_\sigma$	$s_1(\emptyset_\sigma) = \mathcal{P}\sigma$	$s_2(\emptyset_\sigma) = \mathcal{P}\mathcal{P}\sigma$
be_σ	$s_1(\mathsf{be}_\sigma) = \mathcal{P}\mathcal{P}\sigma$	$s_2(\mathsf{be}_\sigma) = \mathcal{P}\sigma$
adj_σ	$s_1(\mathsf{adj}_\sigma) = \mathcal{P}\sigma$	$s_2(\mathsf{adj}_\sigma) = \mathcal{P}(\mathcal{P}\sigma \times \sigma)$

Les quatres ensembles $\mathcal{K}$, Z, H et S disposent chacune de règles qui permettent d'évaluer le type des coercions ; du fait des restrictions de sûreté qui pèsent sur l'utilisation des coercions, on notera que le jugement des coercions inverses n'est en pratique jamais utilisé. Les règles sont données ci-dessous.

Règles pour $\mathcal{K}$:

$$\frac{c \in K}{\vdash c : \kappa_1(c) < \kappa_2(c)}\,(\text{base}) \qquad \frac{\vdash c : \sigma < \tau \qquad \vdash c' : \sigma' < \tau'}{\vdash c \times c' : \sigma \times \sigma' < \tau \times \tau'}\,(\text{prod}) \qquad \frac{\vdash c : \sigma < \tau}{\vdash \exists c : \mathcal{P}\sigma < \mathcal{P}\tau}\,(\text{pred})$$

$$\frac{\vdash c : \sigma < \tau \qquad \xi \in \mathbb{T}}{\vdash c \times \mathrm{id}_\xi : \sigma \times \xi < \tau \times \xi}\,(\text{prid}) \qquad \frac{\vdash c : \sigma < \tau \qquad \xi \in \mathbb{T}}{\vdash \mathrm{id}_\xi \times c : \xi \times \sigma < \xi \times \tau}\,(\text{idpr})$$

Règles pour H :

$$\frac{a \in \mathfrak{a}(r)}{\vdash_{\mathrm{H}} p_{r,a} : r \to a}\,(\text{h-base}) \quad \frac{\vdash_{\mathrm{H}} p : \sigma \to \tau}{\vdash_{\mathrm{H}} \exists p : \mathcal{P}\sigma \to \mathcal{P}\tau}\,(\text{h-pred}) \quad \frac{\vdash_{\mathrm{H}} p : \sigma \to \tau \qquad \vdash_{\mathrm{H}} p' : \sigma' \to \tau'}{\vdash_{\mathrm{H}} p \times p' : \sigma \times \sigma' \to \tau \times \tau'}\,(\text{h-prod})$$

$$\frac{\vdash_{\mathrm{H}} p : \sigma \times \tau}{\vdash_{\mathrm{H}} p \times \mathrm{id}_\xi : \sigma \times \xi \to \tau \times \xi}\,(\text{h-prid}) \qquad \frac{\vdash_{\mathrm{H}} p : \sigma \times \tau}{\vdash_{\mathrm{H}} \mathrm{id}_\xi \times p : \xi \times \sigma \to \xi \times \tau}\,(\text{h-idpr})$$

Règles pour Z :

$$\frac{\vdash c : \sigma < \tau}{\vdash \mathcal{P}c : \mathcal{P}\tau \to \mathcal{P}\sigma}\,(\text{rev}) \qquad \frac{\vdash_{\mathrm{H}} p : \sigma \to \tau}{\vdash \mathcal{P}p : \mathcal{P}\tau \to \mathcal{P}\sigma}\,(\text{h-rev})$$

Règle pour S :

$$\frac{k \in S}{\vdash_{\mathrm{S}} k : s_1(k) \to s_2(k)}\,(\text{s-struct})$$

A.3 Termes

Nous supposons disposer d'un ensemble V de variables ainsi que d'une signature de constantes $(Q, \mathfrak{t})$. Nous imposons pour cette signature l'existence des constantes logiques suivantes, pour tous types $\sigma, \tau \in \mathbb{T}$:

$\top : t$	$\wedge^\sigma : \mathcal{P}(\mathcal{P}\sigma \times \mathcal{P}\sigma \times \sigma)$	$\exists^{\sigma,\tau} : \mathcal{P}(\mathcal{P}(\sigma \times \tau) \times \tau)$
$\bot : t$	$\vee^\sigma : \mathcal{P}(\mathcal{P}\sigma \times \mathcal{P}\sigma \times \sigma)$	$\forall^{\sigma,\tau} : \mathcal{P}(\mathcal{P}(\sigma \times \tau) \times \tau)$
$\neg^\sigma : \mathcal{P}(\mathcal{P}\sigma \times \sigma)$	$\Rightarrow^\sigma : \mathcal{P}(\mathcal{P}\sigma \times \mathcal{P}\sigma \times \sigma)$	

L'ensemble Λ des termes se construit alors selon la grammaire suivante :

$$\Lambda ::= * \mid V \mid Q \mid (\Lambda)\Lambda \mid \lambda V.\Lambda \mid \langle \Lambda, \Lambda \rangle \mid \pi_i \Lambda \mid \mathcal{K}\Lambda \mid H\Lambda \mid Z\Lambda \mid S\Lambda \qquad (i \in \{1,2\})$$

Comme il est usage en lambda-calcul, les parenthèses de l'application pourront être omises lorsqu'aucune ambiguïté n'est possible ; en particulier, le terme uvw est une abréviation pour $((u)v)w$. Rappelons également qu'étant donnés $u, v \in \Lambda$ et $x \in V$, $u[v/x]$ est le terme u dans lequel toutes les occurrences de x ont été substituées par v.

A.4 Jugements de typage

Un jugement est de la forme $\Gamma \vdash u : \sigma$ avec Γ un contexte isomorphe à un élément de $(V \times \mathbb{T})^*$, $u \in \Lambda$ et $\sigma \in \mathbb{T}$. Étant donnés un type de base $a \in \mathbb{B} \cup \mathbb{H}$, un type ontologique $b \in \mathbb{B}$, des contextes de type $d_1, \ldots, d_n$ et une coercion $c : a < b$, on utilisera dans les règles ci-dessous les abréviations de coercions suivantes, toujours constructibles à partir de c :

$$\begin{aligned} c' &: \mathcal{P}(\textstyle\prod_{\ell=1}^{n} d_\ell[a]) < \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b]) \\ c'' &: (\textstyle\prod_{\ell=1}^{i} d_\ell[a]) < (\prod_{\ell=1}^{i} d_\ell[b]) \\ c''' &: \mathcal{P}(\textstyle\prod_{\ell=i+1}^{n} d_\ell[a]) < \mathcal{P}(\prod_{\ell=i+1}^{n} d_\ell[b]) \end{aligned}$$

Règles sans coercion :

$$\frac{}{\Gamma, x : \tau, \Gamma' \vdash x : \tau}\,(\text{ax}) \qquad \frac{q \in Q}{\Gamma \vdash q : \mathfrak{t}(q)}\,(\text{const}) \qquad \frac{}{\Gamma \vdash * : 1}\,(\text{unit})$$

$$\frac{\Gamma \vdash u : \sigma \quad \Gamma \vdash v : \tau}{\Gamma \vdash \langle u, v \rangle : \sigma \times \tau}\,(\text{pair}) \qquad \frac{\Gamma \vdash u : \sigma \times \tau}{\Gamma \vdash \pi_1 u : \sigma}\,(\text{proj}_1) \qquad \frac{\Gamma \vdash u : \sigma \times \tau}{\Gamma \vdash \pi_2 u : \tau}\,(\text{proj}_2)$$

$$\frac{\Gamma, x : \alpha \vdash u : t}{\Gamma \vdash \lambda x.u : \mathcal{P}\alpha}\,(\text{lam}_1) \qquad \frac{\Gamma, x : \alpha \vdash u : \mathcal{P}\beta \quad y \notin \text{fv}(u) \quad y \notin \Gamma}{\Gamma \vdash \lambda y.(u(\pi_2 y))[\pi_1 y/x] : \mathcal{P}(\alpha \times \beta)}\,(\text{lam}_2)$$

$$\frac{\Gamma \vdash u : \mathcal{P}\alpha \quad \Gamma \vdash v : \alpha}{\Gamma \vdash uv : t}\,(\text{app}_t) \qquad \frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} \alpha_\ell) \quad \Gamma \vdash v : \prod_{\ell=1}^{i} \alpha_\ell}{\Gamma \vdash uv : \mathcal{P}(\prod_{\ell=i+1}^{n} \alpha_\ell)}\,(\text{app}_p)$$

Règles avec coercions :

$$\frac{\Gamma \vdash u : \sigma \quad \vdash_{\text{S}} s : \sigma \to \tau}{\Gamma \vdash su : \tau}\,(\text{s-app}) \qquad \frac{\Gamma \vdash u : \mathcal{P}\mathcal{P}a \quad \Gamma \vdash v : \mathcal{P}b \quad c : a < b}{\Gamma \vdash u((\mathcal{P}c)v) : t}\,(\text{abais})$$

$$\frac{\Gamma \vdash u : \mathcal{P}d[b] \quad \Gamma \vdash v : d[a] \quad \vdash c : d[a] < d[b]}{\Gamma \vdash u\,(cv) : t}\,(\text{act})$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b]) \quad \Gamma \vdash v : \prod_{\ell=1}^{i} d_\ell[a] \quad \vdash c : a < b}{\Gamma \vdash (\vec{c}'u)v : \mathcal{P}(\prod_{\ell=i+1}^{n} d_\ell[a])}\,(\text{acp})$$

$$\frac{\Gamma \vdash u : \mathcal{P}d[b] \quad \Gamma \vdash v : d[r] \quad \vdash_{\text{H}} p : d[r] \to d[a] \quad \vdash c : d[a] < d[b]}{\Gamma \vdash u(c(pv)) : t}\,(\text{h-app}_t)$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b]) \quad \Gamma \vdash v : \prod_{\ell=1}^{i} d_\ell[r] \quad \vdash_{\text{H}} p : r \to a \quad \vdash c : a < b}{\Gamma \vdash (\vec{p}'(\vec{c}'u))v : \mathcal{P}(\prod_{\ell=i+1}^{n} d_\ell[r])}\,(\text{h-app}_p),$$

A.5 Théorie équationelle

Les jugements sont de la forme $\Gamma \vdash u = v : \sigma$ avec les mêmes conditions que précédemment, et $v \in \Lambda$. Nous omettons ici les règles classiques qui permettent de fonder la relation $=$ comme une congruence, mais elles sont néanmoins applicables à CPTT.

$$\frac{\Gamma \vdash u : 1}{\Gamma \vdash u = * : 1}\,(\text{1-eq}) \qquad \frac{\Gamma \vdash u : \sigma \times \tau}{\Gamma \vdash \langle \pi_1 u, \pi_2 u\rangle = u : \sigma \times \tau}\,(\text{rectr})$$

$$\frac{\Gamma \vdash u : \sigma \quad \Gamma \vdash v : \tau}{\Gamma \vdash \pi_1\langle u, v\rangle = u : \sigma}\,(\text{preq}_1) \qquad \frac{\Gamma \vdash u : \sigma \quad \Gamma \vdash v : \tau}{\Gamma \vdash \pi_2\langle u, v\rangle = v : \tau}\,(\text{preq}_2)$$

$$\frac{\Gamma, x : \sigma \vdash u : t \quad \Gamma \vdash v : \sigma}{\Gamma \vdash (\lambda x.u)v = u[v/x] : t}\,(\beta_t) \qquad \frac{\Gamma \vdash u : \mathcal{P}\sigma \quad x \notin \mathrm{fv}(u)}{\Gamma \vdash \lambda x.ux = u : \mathcal{P}\sigma}\,(\eta)$$

$$\frac{\Gamma \vdash \lambda x.u : \mathcal{P}(\prod_{\ell=1}^{n} \alpha_\ell) \quad \Gamma \vdash v : \prod_{\ell=1}^{i} \alpha_\ell \quad y \notin \mathrm{fv}(u, v) \quad y \notin \Gamma}{\Gamma \vdash (\lambda x.u)v = \lambda y.u[\langle \pi_1 v, \ldots, \pi_i v, y\rangle/x] : \mathcal{P}(\prod_{\ell=i+1}^{n} \alpha_\ell)}\,(\beta_p)$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} \alpha_\ell) \quad \Gamma \vdash v : \prod_{\ell=1}^{i} \alpha_\ell \quad \Gamma \vdash w : \prod_{\ell=i+1}^{j} \alpha_\ell}{\Gamma \vdash (uv)w = u\langle \pi_1 v, \ldots, \pi_i v, \pi_1 w, \ldots, \pi_{j-i} w\rangle : \mathcal{P}(\prod_{\ell=j+1}^{n} \alpha_\ell)}\,(\text{factor})$$

$$\frac{\Gamma \vdash u : \mathcal{P}d[b] \quad \Gamma \vdash v : d[a] \quad \vdash c : d[a] < d[b]}{\Gamma \vdash ((\mathcal{P}c)u)v = u(cv) : t}\,(\text{eqct})$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\prod_{\ell=1}^{n} d_\ell[b]) \quad \Gamma \vdash v : \prod_{\ell=1}^{i} d_\ell[a] \quad \vdash c : a < b}{\Gamma \vdash (\vec{c}'u)v = \vec{c}'''(u(c''v)) : \mathcal{P}(\prod_{\ell=i+1}^{n} d_\ell[a])}\,(\text{eqcp})$$

$$\frac{\Gamma \vdash u : \mathcal{P}\sigma \quad \Gamma \vdash v : \mathcal{P}\sigma \quad \vdash c : \mathcal{P}\sigma < \mathcal{P}\tau}{\Gamma \vdash \vee^\tau(c \times c)\langle u, v\rangle = c(\vee^\sigma\langle u, v\rangle) : \mathcal{P}\tau}\,(\vee\text{-eq}),$$

$$\frac{\Gamma \vdash u : \mathcal{P}\sigma \quad \Gamma \vdash v : \mathcal{P}\sigma \quad \vdash c : \mathcal{P}\sigma < \mathcal{P}\tau}{\Gamma \vdash \wedge^\tau(c \times c)\langle u, v\rangle = c(\wedge^\sigma\langle u, v\rangle) : \mathcal{P}\tau}\,(\wedge\text{-eq})$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\sigma \times \xi) \quad \vdash c : \mathcal{P}(\sigma \times \xi) < \mathcal{P}(\tau \times \xi)}{\Gamma \vdash \exists^{\tau,\xi}(cu) = \exists^{\sigma,\xi} u : \mathcal{P}\xi}\,(\exists_1\text{-eq})$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\sigma \times \xi) \quad \vdash c : \mathcal{P}(\sigma \times \xi) < \mathcal{P}(\sigma \times \zeta) \quad \vdash c' : \mathcal{P}\xi < \mathcal{P}\zeta}{\Gamma \vdash \exists^{\sigma,\zeta}(cu) = c'(\exists^{\sigma,\xi} u) : \mathcal{P}\zeta}\,(\exists_2\text{-eq})$$

$$\frac{\Gamma \vdash u : \mathcal{P}(\sigma \times \xi) \quad \vdash c : \mathcal{P}(\sigma \times \xi) < \mathcal{P}(\sigma \times \zeta) \quad \vdash c' : \mathcal{P}\xi < \mathcal{P}\zeta}{\Gamma \vdash \forall^{\sigma,\zeta}(cu) = c'(\forall^{\sigma,\xi} u) : \mathcal{P}\zeta}\,(\forall\text{-eq})$$

$$\frac{\Gamma \vdash u : \sigma}{\Gamma \vdash \mathsf{be}_\sigma(\mathsf{lift}_\sigma u) = \mathsf{ident}_\sigma u : \mathcal{P}\sigma}\,(\text{partee}) \qquad \frac{\Gamma \vdash u : \mathcal{P}\sigma}{\Gamma \vdash \mathsf{adj}_\sigma u = \wedge^\sigma u : \mathcal{P}(\mathcal{P}\sigma \times \sigma)}\,(\text{intersect})$$

Bibliographie

También alegó un hecho que todos los viajeros han confirmado:
No hay en la vasta Biblioteca, dos libros idénticos.

Jorge Luis Borges, *La biblioteca de Babel*

[1] Martín Abadi et Luca Cardelli. *A Theory of Objects*. Springer, New York, 1996.

[2] Jiří Adámek, Horst Herrlich et George E. Strecker. *Abstract and Concrete Categories: The Joy of Cats*. [Auto-édité en ligne], 2004.

[3] Juri Derenikovič Apresjan. « Analyse distributionnelle des significations et champs sémantiques structurés ». *Langages*, **1**, pp. 44–74, 1966. Traduit par Yvan Mignot.

[4] Nicholas Asher. *Lexical Meaning in Context: A Web of Words*. Cambridge University Press, 2011.

[5] Nicholas Asher et James Pustejovsky. « A Type Composition Logic for Generative Lexicon ». *Journal of Cognitive Science*, **7**(1), pp. 1–38, 2006.

[6] Jenny Audring et Geer Booij. « Cooperation and coercion ». *Linguistics*, **54**(4), pp. 617–637, 2016.

[7] William Babonnaud. « A Topos-Based Approach to Building Language Ontologies ». Dans Raffaella Bernardi, Gregory M. Kobele et Sylvain Pogodalla (réds.), *Proceedings of the 24th International Conference on Formal Grammar (FG 2019)*, pp. 18–34. Springer, Berlin, 2019.

[8] William Babonnaud. « Covariant Subtyping Applied to Semantic Predicate Calculi ». Dans *Proceedings of the 10th International Conference on Logical Aspects of Computational Linguistics (LACL 2021)*. 2021.

[9] William Babonnaud. « On the Dual Interpretation of Nouns as Types and Predicates in Semantic Type Theories ». Dans *Proceedings of the 2nd Workshop on Computing Semantics with Types, Frames and Related Structures*. 2021.

[10] William Babonnaud et Philippe de Groote. « Lexical selection, coercion, and record types ». Dans *Proceedings of the 17th Workshop on Logic & Engineering of Natural Language Semantics (LENLS17)*. 2020.

[11] William Babonnaud, Laura Kallmeyer et Rainer Osswald. « Polysemy and Coercion – A Frame-Based Approach Using LTAG and Hybrid Logic ». Dans Maxime Amblard, Philippe de Groote, Sylvain Pogodalla et Christian Retoré (réds.), *Proceedings of the 9th International Conference on Logical Aspects of Computational Linguistics (LACL 2016)*, pp. 18–33. Springer, 2016.

[12] Collin F. Baker, Charles J. Fillmore et John B. Lowe. « The Berkeley FrameNet Project ». Dans Christian Boitet et Pete Whitelock (réds.), *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and of the 17th International Conference on Computational Linguistics*, pp. 86–90. Association for Computational Linguistics, 1998.

[13] Yehoshua Bar-Hillel. « Syntactical and semantical categories ». Dans Borchert [18].

[14] Hendrik Pieter Barendregt. *The Lambda-Calculus: Its Syntax and Semantics*. Elsevier, 1984.

[15] Christian Bassac, Bruno Mery et Christian Retoré. « Towards a Type-Theoretical Account of Lexical Semantics ». *Journal of Logic, Language, and Information*, **19**(2), pp. 229–245, 2010.

[16] Jonathan Bennett. *Events and their Names*. Clarendon Press, Oxford, 1988.

[17] Claire Beyssade et Carmen Dobrovie-Sorin. « A Syntax-Based Analysis of Predication ». Dans Efthymia Georgala et Jonathan Howell (réds.), *Proceedings of SALT 15*, pp. 44–61. Cornwell University, Ithaca, 2005.

[18] Donald M. Borchert (réd.). *Encyclopedia of Philosophy*. Thompson Gale, 2^{e} édition, 2006.

[19] Daniel Bourquin. « Les catégories syntaxico-sémantiques : petite histoire d'un grand problème ». *Travaux de logique*, **11**, pp. 1–34, 1996.

[20] John Boyland et Giuseppe Castagna. « Type-Safe Compilation of Covariant Specialization: A Practical Case ». rap. tech. UCB/CSD-95-890, University of California, 1996.

[21] Andreas Brandstädt, Van Bang Le et Jeremy P. Spinrad. *Graph Classes: A Survey*. Society for Industrial and Applied Mathematics, Philadelphia, 1999.

[22] Val Breazu-Tannen, Thierry Coquand et Carl A. Gunter. « Inheritance as Implicit Coercion ». *Information and Computation*, **93**, pp. 172–221, 1991.

[23] Edward J. Briscoe, Ann Copestake et Branimir K. Boguraev. « Enjoy the paper: lexical semantics via lexicology ». Dans Hans Karlgren (réd.), *Proceedings of the 13th International Conference on Computational Linguistics (COLING 1990)*, pp. 42–47. Association for Computational Linguistics, 1990.

[24] Peer F. Bundgaard. « Husserl and Language ». Dans Daniel Schmicking et Shaun Gallagher (réds.), *Handbook of Phenomenology and Cognitive Sciences*, pp. 368–399. Springer, Dordrecht, 2010.

[25] Heather Burnett et Peter Sutton. « Vagueness and Natural Language Semantics ». Dans Daniel Gutzmann, Lisa Matthewson, Cécile Meier, Hotze Rullmann et Thomas Ede Zimmermann (réds.), *The Wiley Blackwell Companion to Semantics*. John Wiley & Sons, 2020.

[26] Luca Cardelli et Peter Wegner. « On Understanding Types, Data Abstraction, and Polymorphism ». *Computing Surveys*, **17**(4), pp. 471–522, 1985.

[27] Rudolf Carnap. *Abriß der Logistik*. Springer, Vienne, 1929.

[28] Rudolf Carnap. « Überwindung der Metaphysik durch logische Analyse der Sprache ». *Erkenntnis*, **2**, pp. 219–241, 1931.

[29] Rudolf Carnap. « Empiricism, Semantics, and Ontology ». *Revue Internationale de Philosophie*, **4**, pp. 20–40, 1950.

[30] Rudolf Carnap. *Meaning and Necessity: A Study in Semantics and Modal Logics*. The University of Chicago Press, 1956.

[31] Pierre Cartier. « Un pays dont on ne connaîtrait que le nom (Grothendieck et les 'motifs') », 2000. Prépublication IHES.

[32] John Cartmell. « Generalised algebraic theories and contextual categories ». *Annals of Pure and Applied Logic*, **32**, pp. 209–243, 1986.

[33] Claudia Casadio. « Semantic categories and the development of categorial grammars ». Dans Richard T. Oehrle, Emmon Bach et Deirdre Wheeler (réds.), *Categorial Grammars and Natural Language Structures*, pp. 95–123. Springer, Dordrecht, 1988.

[34] Stergios Chatzikyriakidis et Zhaohui Luo. « On the Interpretation of Common Nouns: Types Versus Predicates ». Dans Stergios Chatzikyriakidis et Zhaohui Luo (réds.), *Modern Perspectives in Type-Theoretical Semantics*, vol. 98 de *Studies in Linguistics and Philosophy*, pp. 43–70. Springer, 2017.

[35] Stergios Chatzikyriakidis et Zhaohui Luo. *Formal Semantics in Modern Type Theories*. ISTE, 2020.

[36] Gennaro Chierchia. « Formal semantics and the grammar of predication ». *Linguistic Inquiry*, **16**(3), pp. 417–443, 1985.

[37] Gennaro Chierchia. « Mass nouns, vagueness and semantic variation ». *Synthese*, **174**, pp. 99–149, 2010.

[38] Noam Chomsky. *The Logical Structure of Linguistic Theory*. MIT Microfilm, 1955.

[39] Noam Chomsky. *Syntactic Structures*. Mouton, 1957.

[40] Noam Chomsky. « Some Methodological Remarks on Generative Grammar ». *WORD*, **17**(2), pp. 219–239, 1961.

[41] Noam Chomsky. *Aspects of the theory of syntax*. MIT Press, 1965.

[42] Alonzo Church. « A Formulation of the Simple Theory of Types ». *The Journal of Symbolic Logic*, **5**(2), pp. 56–68, 1940.

[43] Leon Chwistek. « Antynomje logiki formalnej ». *Przegląd Filozoficzny*, **24**(3-4), pp. 164–171, 1921.

[44] Bob Coecke, Mehrnoosh Sadrzadeh et Stephen Clark. « Mathematical Foundations for a Compositional Distributional Model of Meaning ». *Linguistic Analysis*, **36**, pp. 345–384, 2010.

[45] L. Jonathan Cohen. « On a concept of degree of grammaticalness ». *Logique et Analyse*, **8**(30), pp. 141–153, 1965.

[46] Ann Copestake et Edward J. Briscoe. « Lexical Operations in a Unification-based Framework ». Dans James Pustejovsky et Sabine Bergler (réds.), *Lexical Semantics and Knowledge Representation*, pp. 88–101. Association for Computational Linguistics, 1991.

[47] Brian A. Davey et Hilary A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2^{e} édition, 2002.

[48] Donald Davidson. *Essays on Actions and Events*. Clarendon Press, Oxford, 1980.

[49] Philippe de Groote. « Towards Abstract Categorial Grammars ». Dans Bonnie L. Webber (réd.), *ACL '01: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pp. 252–259. Association for Computational Linguistics, 2001.

[50] Philippe de Groote. « Type raising, continuations, and classical logic ». Dans Martin J. B. Stokhof et Robert A. M. van Rooij(réds.), *Proceedings of the Thirteenth Amsterdam Colloquium*, pp. 97–101. ILLC, Amsterdam, 2001.

[51] Philippe de Groote et Makoto Kanazawa. « A Note on Intensionalization ». *Journal of Logic, Language and Information*, **22**(2), pp. 173–194, 2013.

[52] Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre et Christopher D. Manning. « Universal Stanford Dependencies: A cross-linguistic typology ». Dans Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk et Stelios Piperidis (réds.), *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pp. 4585–4592. Association for Computational Linguistics, 2014.

[53] Samuel Eilenberg et Saunders MacLane. « General Theory of Natural Equivalences ». *Transactions of the American Mathematical Society*, **58**(2), pp. 231–294, 1945.

[54] George Englebretsen. « Vacuosity ». *Mind*, **81**(2), pp. 273–275, 1972.

[55] George Englebretsen. « La théorie des catégories de Sommers : une nouvelle introduction ». *Dialogue*, **27**(3), pp. 451–473, 1988. Traduit par Odette Mercure.

[56] George Englebretsen. *Robust Reality: An Essay in Formal Ontology*. Ontos Verlag, Frankfurt, 2012.

[57] William M. Farmer. « The seven virtues of simple type theory ». *Journal of Applied Logic*, **6**(3), pp. 267–286, 2008.

[58] Christiane Fellbaum (réd.). *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, 1998.

[59] Nora Fieder, Lyndsey Nickels et Britta Biedermann. « Representation and processing of mass and count nouns: a review ». *Frontiers in Psychology*, **5**, p. 589, 2014.

[60] Charles J. Fillmore. « The Case for Case ». Dans Emmon W. Bach et Robert T. Harms (réds.), *Universals in Linguistic Theory*, pp. 1–88. Holt, Rinehart and Winston, 1968.

[61] John Rupert Firth. « A synopsis of linguistic theory 1930-1955 ». Dans John Rupert Firth (réd.), *Studies in Linguistic Analysis*, pp. 1–32. Blackwell, Oxford, 1957.

[62] Michael P. Fourman. « The Logic of Topoi ». Dans Jon Barwise (réd.), *Handbook of Mathematical Logic*, vol. 90 de *Studies in Logic and the Foundations of Mathematics*, pp. 1053–1090. Elsevier, 1977.

[63] Gottlob Frege. *Function und Begriff*. Verlag von Hermann Phole, Jena, 1891. Traduit comme “Function and concept” dans [69].

[64] Gottlob Frege. « Über Sinn und Bedeutung ». *Zeitschrift für Philosophie und philosophische Kritik*, **100**, pp. 25–50, 1892. Traduit comme “On Sense and Reference” dans [69].

[65] Robin O. Gandy. « The Simple Theory of Types ». Dans Robin O. Gandy et John M. E. Hyland (réds.), *Logic Colloquium 76*, pp. 173–181. North-Holland Publishings, 1977.

[66] Robin O. Gandy. « General Introduction to Turing's work on Type Theory ». Dans Gandy et Yates [67], pp. 151–154.

[67] Robin O. Gandy et C. E. M. Yates (réds.). *Collected Works of A. M. Turing, vol. 4: Mathematical Logic*. Elsevier, 2001.

[68] Roger Garside, Geoffrey Leech et Tony McEnery (réds.). *Corpus Annotation*. Addison Wiley, 1997.

[69] Peter Geach et Max Black (réds.). *Translations from the Philosophical Writings of Gottlob Frege*. Basil Blackwell, Oxford, 1952.

[70] Dirk Geeraerts. « Prospects and problems of prototype theory ». *Linguistics*, **27**(4), pp. 587–612, 1989.

[71] Dirk Geeraerts. *Theories of Lexical Semantics*. Oxford University Press, 2010.

[72] Béatrice Godart-Wendling. « Le statut du mot dans les formalismes de Carnap et d'Ajdukiewicz ». Dans Bernard Fradin et Jean-Marie Marandin (réds.), *Mot et grammaires*, Études de sémantique lexicale, chap. 6, pp. 181–197. Didier Érudition, Paris, 1997.

[73] Béatrice Godart-Wendling. « Montague et les catégories d'Ajdukiewicz ». *Travaux de logique*, **13**, pp. 159–175, 1999.

[74] Kurt Gödel. « Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I ». *Monatshefte für Mathematik und Physik*, **38**, pp. 173–198, 1931. Traduit comme "On formally undecidable propositions of Principia mathematica and related systems I" dans [181].

[75] Robert Goldblatt. *Topoi: The Categorial Analysis of Logic*, vol. 98 de *Studies in logic and the foundations of mathematics*. North-Holland Publishings, 1979.

[76] Maria Gołębiewska. « Edmund Husserl's Semantics and the Critical Theses of Late Structuralism ». *Eidos. A Journal for Philosophy of Culture*, **3**(1), pp. 30–50, 2019.

[77] Martin Charles Golumbic. « Trivially perfect graphs ». *Discrete Mathematics*, **24**(1), pp. 105–107, 1978.

[78] Nicola Guarino, Daniel Oberle et Steffen Staab. « What Is an Ontology? » Dans Steffen Staab et Rudi Studer (réds.), *Handbook on Ontologies*, pp. 1–17. Springer, Berlin, 2009.

[79] Bruno Guillaume. « Graph Matching and Graph Rewriting: GREW tools for corpus exploration, maintenance and conversion ». Dans Dimitra Gkatzia et Djamé Seddah (réds.), *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations*, pp. 168–175. Association for Computational Linguistics, 2021.

[80] Alan G. Hamilton. *Logic for Mathematicians*. Cambridge University Press, 1978.

[81] Gilbert Harman. « Quine on Meaning and Existence, I. The Death of Meaning ». *The Review of Metaphysics*, **21**(1), pp. 124–151, 1967.

[82] Gilbert Harman. « Quine on Meaning and Existence, II. Existential committment ». *The Review of Metaphysics*, **21**(2), pp. 343–367, 1967.

[83] Bernard Harrison. « Category Mistakes and Rules of Language ». *Mind*, **74**(3), pp. 309–325, 1965.

[84] John A. Hartigan et M. A. Wong. « A K-Means Clustering Algorithm ». *Journal of the Royal Statistical Society: Series C*, **28**(1), pp. 100–108, 1979.

[85] Allen P. Hazen. « Type Theory ». Dans Borchert [18].

[86] Leon Henkin. « Completeness in the theory of types ». *The Journal of Symbolic Logic*, **15**(2), pp. 81–91, 1950.

[87] Laurence R. Horn. *A Natural History of Negation*. The University of Chicago Press, 1989.

[88] Bart Jacobs. *Categorical Logic and Type Theory*, vol. 141 de *Studies in Logic and the Foundations of Mathematics*. Elsevier, Amsterdam, 1999.

[89] Bart Jacobs et Jan Rutten. « A Tutorial on (Co)Algebras and (Co)Induction ». *EATCS Bulletin*, **62**, pp. 222–259, 1997.

[90] Peter T. Johnstone. *Sketches of an Elephant: A Topos Theory Compendium*. Oxford University Press, 2002.

[91] Aravind K. Joshi et Yves Shabes. « Tree-Adjoining Grammars ». Dans Grzegorz Rozenberg et Arto Salomaa (réds.), *Handbook of Formal Languages*, pp. 69–123. Springer, Berlin, 1997.

[92] Johan A. W. Kamp. « Two theories about adjectives ». Dans Edward L. Keenan (réd.), *Formal Semantics of Natural Languages*, pp. 123–155. Cambridge University Press, Cambridge, 1975.

[93] Jerrold J. Katz. « Semi-sentences ». Dans Jerry A. Fodor et Jerrold J. Katz (réds.), *The Structure of Language: Readings in the Philosophy of Language*, pp. 400–416. Prentice-Hall, 1964.

[94] Jerrold J. Katz. *Sense, reference, and philosophy*. Oxford University Press, 2004.

[95] Jerrold J. Katz et Jerry A. Fodor. « The Structure of a semantic theory ». *Language*, **39**(2), pp. 170–210, 1963.

[96] Jerrold J. Katz et Paul M. Postal. *An Integrated Theory of Linguistics Descriptions*, vol. 26 de *Research Monograph*. The MIT Press, Cambridge, 1964.

[97] Ferenc Kiefer. « Some Semantic Relations in Natural Language ». Dans Harry H. Josselson (réd.), *Proceedings of the Conference on Computer-related Semantic Analysis*, pp. VII/1–23. Wayne State University, Detroit, 1966.

[98] Jaegwon Kim. *Supervenience and Mind: Selected Philosophical Essays*. Cambridge University Press, 1993.

[99] Marie La Palme Reyes, John Macnamara et Gonzalos E. Reyes. « Reference, Kinds and Predicates ». Dans John Macnamara et Gonzalo E. Reyes (réds.), *The Logical Foundations of Cognition*, vol. 4 de *Vancouver Studies in Cognitive Science*, pp. 91–143. Oxford University Press, 1994.

[100] Joachim Lambek. « From λ-calculus to cartesian closed categories ». Dans J. Roger Hindley et Jonathan P. Seldin (réds.), *To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pp. 375–402. Academic Press, London, 1980.

[101] Joachim Lambek et Philip J. Scott. *Introduction to Higher Order Categorical Logic*. Cambridge University Press, 1986.

[102] Velma Joan Leeding. *Anindilyakwa Phonology and Morphology*. Thèse de doctorat, Sydney University, 1989.

[103] Ernest Lepore. « Philosophy of Language ». Dans Borchert [18].

[104] David Lewis. *Events*, pp. 241–269. Oxford University Press, 1986.

[105] Xiaobin Li, Stan Szpakowicz et Stan Matwin. « A WordNet-based algorithm for word sense disambiguation ». Dans *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, vol. II*, pp. 1368–1376. Morgan Kaufmann Publishers, 1995.

[106] John A. Lucy. « Sapir-Whorf Hypothesis ». Dans James D. Wright (réd.), *International Encyclopedia of the Social & Behavioral Sciences*, pp. 13486–13490. Elsevier, 2001.

[107] Zhaohui Luo. « Type-theoretical semantics with coercive subtyping ». Dans Nan Li et David Lutz (réds.), *Proceedings of SALT 20*, pp. 38–56. 2010.

[108] Zhaohui Luo. « Common Nouns as Types ». Dans Denis Béchet et Alexandre Dikovsky (réds.), *Logical Aspects of Computational Linguistics. 7th International Conference, LACL 2012, Nantes, France, July 2-4, 2012, Proceedings*, pp. 173–185. Springer, Berlin, 2012.

[109] Zhaohui Luo. « Formal semantics in modern type theories with coercive subtyping ». *Linguistics and Philosophy*, **35**(6), pp. 491–513, 2012.

[110] Zhaohui Luo, Sergei Soloviev et Tao Xue. « Coercive subtyping: Theory and implementation ». *Information and Computation*, **223**, pp. 18–42, 2013.

[111] Veronika Lux-Pogodalla et Alain Polguère. « Construction of a French Lexical Network: Methodological Issues ». Dans Benoît Sagot (réd.), *Proceedings of the First International Workshop on Lexical Resources (WoLeR 2011)*, pp. 54–61. 2011.

[112] John Lyons. *Semantics*. Cambridge University Press, 1977.

[113] Saunders MacLane. *Categories for the Working Mathematician*. Springer, 1971.

[114] Saunders MacLane et Ieke Moerdijk. *Sheaves in Geometry and Logic: A First Introduction to Topos Theory*. Springer, New York, 1992. Corr. 2nd edition 1994.

[115] Per Martin-Löf. *Intuitionistic type theory: Notes by Giovanni Sambin of a series of lectures given in Padua, June 1980*. Bibliopolis, Napoli, 1984.

[116] Bruno Mery, Richard Moot et Christian Retoré. « Solving the Individuation and Counting Puzzle with λ-DRT and MGL ». Dans Kazuhiro Kojima, Maki Sakamoto, Koji Mineshima et Ken Satoh (réds.), *New Frontiers in Artificial Intelligence: Proceedings of JSAI-isAI 2018 Workshops, Yokohama, Japan*, pp. 298–312. Springer, 2019.

[117] Bruno Mery et Christian Retoré. « Semantic Types, Lexical Sorts and Classifiers ». Dans Bernadette Sharp et Michael Zock (réds.), *Proceedings of NLPCS 2013, 10th InternaDonal Workshop on Natural Language Processing and Cognitive Science, Marseille, France, October 2013*, pp. 49–64. 2013.

[118] George A. Miller. « WordNet: a lexical database for English ». *Communication of the ACM*, **38**(11), pp. 39–41, 1995.

[119] Richard Montague. « Universal grammar ». *Theoria*, **36**, pp. 373–398, 1970.

[120] Richard Montague. « The Proper Treatment of Quantification in Ordinary English ». Dans Patrick Suppes, Julius Moravcsik et Jaakko Hintikka (réds.), *Approaches to Natural Language*, pp. 221–242. Reidel, Dordrecht, 1973.

[121] Roberto Navigli. « Word sense disambiguation: A survey ». *ACM Computing Surveys*, **41**(2), pp. 1–69, 2009.

[122] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Jan Hajič, Christopher D. Manning, Sampo Pyysalo, Sebastian Schuster, Francis Tyers et Daniel Zeman. « Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection ». Dans Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk et Stelios Piperidis (réds.), *Proceedings of the 12th International Conference on Language Resources and Evaluation*, pp. 4034–4043. Association for Computational Linguistics, 2020.

[123] Geoffrey Nunberg. « The Non-Uniqueness of Semantic Solutions: Polysemy ». *Linguistics and Philosophy*, **3**(2), pp. 143–184, 1979.

[124] Geoffrey Nunberg. « Transfers of Meaning ». *Journal of Semantics*, **12**(2), pp. 109–132, 1995.

[125] Steven Orey. « Model Theory for the Higher Order Predicate Calculus ». *Transactions of the American Mathematical Society*, **92**, pp. 72–84, 1959.

[126] Arthur Pap. « Types and meaninglessness ». *Mind*, **69**(1), pp. 41–54, 1960.

[127] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley Publishings, 1994.

[128] Terence Parsons. *Events in the Semantics of English: A Study in Subatomic Semantics*. The MIT Press, 1990.

[129] Barbara H. Partee. « Noun Phrase Interpretation and Type-shifting Principles ». Dans Jeroen Groenendijk, Dick de Jongh et Martin Stokhof (réds.), *Studies in discourse representation theory and the theory of generalized quantifiers*, pp. 115–143. De Gruyter, 1986.

[130] Barbara H. Partee. « Compositionality and coercion in semantics: The dynamics of adjective meaning ». Dans Gerlof Bouma, Irene Krämer et Joost Zwarts (réds.), *Cognitive Foundations of Interpretation*, pp. 145–161. Royal Netherlands Academy of Arts and Sciences, Amsterdam, 2007.

[131] Barbara H. Partee et Mats Rooth. « Generalized Conjuction and Type Ambiguity ». Dans Rainer Bäuerle, Christoph Schwarze et Armin von Stechow (réds.), *Meaning, Use, and the Interpretation of Language*, pp. 361–383. de Gruyter, Berlin, 1983.

[132] Benjamin C. Pierce. *Basic Category Theory for Computer Scientists*. The MIT Press, Cambridge, 1991.

[133] Sylvain Pogodalla. « ACGTK : un outil de développement et de test pour les grammaires catégorielles abstraites ». Dans Laurence Danlos et Thierry Hamon (réds.), *Actes de la 23ème Conférence sur le Traitement Automatique des Langues Naturelles, 31ème Journées d'Études sur la Parole, 18ème Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (JEP-TALN-RECITAL 2016)*, pp. 1–2. Association pour le Traitement Automatique des Langues, Paris, 2016.

[134] François Pottier. *Synthèse de types en présence de sous-typage : de la théorie à la pratique*. Thèse de doctorat, Université Paris VII, 1998.

[135] James Pustejovsky. *The Generative Lexicon*. MIT Press, Cambridge, 1995.

[136] James Pustejovsky. « Type Construction and the Logic of Concepts ». Dans Pierrette Bouillon et Federica Busa (réds.), *The Language of Word Meaning*, chap. 7, pp. 91–123. Cambridge University Press, 2001.

[137] James Pustejovsky. « A Survey of Dot Objects », 2005. Non publié.

[138] James Pustejovsky. « Coercion in a general theory of argument selection ». *Linguistics*, **49**(6), pp. 1401–1431, 2011.

[139] James Pustejovsky et Olga Batiukova. *The Lexicon*. Cambridge University Press, 2019.

[140] James Pustejovsky et Branimir K. Boguraev. « Lexical knowledge representation and natural language processing ». *Artificial Intelligence*, **63**(1-2), pp. 193–223, 1993.

[141] James Pustejovsky et Pierrette Bouillon. « Aspectual Coercion and Logical Polysemy ». *Journal of Semantics*, **12**(2), pp. 133–162, 1995.

[142] James Pustejovsky et Elisabetta Jezek. « Semantic Coercion in Language: Beyond Distributional Analysis ». *Rivista di Linguistica*, **20**(1), pp. 181–214, 2008.

[143] James Pustejovsky et Anna Rumshisky. « Mechanisms of sense extensions in verbs ». Dans Gilles-Maurice de Schryver (réd.), *A way with words: Recent advances in lexical theory and analysis – A Festschrift for Patrick Hanks*. Menha Publishers, Kampala, 2010.

[144] Hilary Putnam. « The Meaning of "Meaning" ». Dans Keith Gunderson (réd.), *Language, Mind, and Knowledge*, vol. 7 de *Minnesota Studies in the Philosophy of Science*, pp. 131–193. University of Minnesota Press, Minneapolis, 1975.

[145] Willard V. O. Quine. « On Carnap's views on ontology ». *Philosophical Studies*, **2**(5), pp. 65–72, 1951.

[146] Willard V. O. Quine. « Two Dogmas of Empiricism ». *The Philosophical Review*, **60**(1), pp. 20–43, 1951.

[147] Frank P. Ramsey. « The Foundations of Mathematics ». *Proceedings of the London Mathematical Society*, **25**, pp. 338–384, 1926.

[148] Tanya Reinhart. « The Theta System – An Overwiew ». *Theoretical Linguistics*, **28**(3), pp. 229–290, 2003.

[149] Christian Retoré. « The Montagovian Generative Lexicon ΛTy_n: a Type Theoretical Framework for Natural Language Semantics ». Dans Ralph Matthes et Aleksy Schubert (réds.), *Proceedings of the 19th International Conference on Types for Proofs and Programs*, vol. 26 de *LIPICS*, pp. 202–229. Dagstuhl Publishings, 2014.

[150] John C. Reynolds. *Theories of Programming Languages*. Cambridge University Press, 1998.

[151] Emily Riehl. *Category Theory in Context*. Dover Publications, 2017.

[152] Eleanor Rosch. « Natural Categories ». *Cognitive Psychology*, **4**, pp. 328–350, 1973.

[153] Eleanor Rosch. « Cognitive representations of semantics categories ». *Journal of Experimental Psychology*, **104**(3), pp. 192–233, 1975.

[154] Bertrand Russell. « On Denoting ». *Mind*, **14**(4), pp. 479–493, 1905.

[155] Bertrand Russell. « Mathematical Logic as Based on the Theory of Types ». *American Journal of Mathematics*, **30**(3), pp. 222–262, 1908.

[156] Bertrand Russell. « Logical Atomism ». Dans John Henry Muirhead (réd.), *Contemporary British Philosophy*, pp. 357–383. George Allen Unwin, London, 1924.

[157] Gilbert Ryle. « Categories ». *Proceedings of the Aristotelian Society*, **38**, pp. 189–206, 1938.

[158] Walid S. Saba. « Logical Semantics and Commonsense Knowledge: Where Did we Go Wrong, and How to Go Forward, Again », 2018. ArXiv preprint.

[159] Walid S. Saba. « Language and its Commonsense: Where Formal Semantics Went Wrong, and Where it Can (and Should) Go. » *Journal of Knowledge Structures and Systems*, **1**(1), pp. 40–62, 2020.

[160] Mehrnoosh Sadrzadeh, Dimitri Kartsaklis et Esma Balkır. « Sentence entailment in compositional distributional semantics ». *Annals of Mathematics and Artificial Intelligence*, **82**, pp. 189–218, 2018.

[161] Anne Salvesen et Jan M. Smith. « The Strength of the Subset Type in Martin-Löf's Type Theory ». Dans *Proceedings of the Third Annual Symposium on Logic in Computer Science (LICS '88)*, pp. 384–391. IEEE, 1988.

[162] Charles Sayward. « A Defense of Sommers ». *Philosophical Studies*, **29**, pp. 343–347, 1976.

[163] Charles Sayward et Stephen H. Voss. « Absurdity and Spanning ». *Philosophia*, **2**(3), pp. 227–238, 1972.

[164] Sebastian Schuster et Christopher D. Manning. « Enhanced English Universal Dependencies: An Improved Representation for Natural Language Understanding Tasks ». Dans Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk et Stelios Piperidis (réds.), *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pp. 2371–2378. Association for Computational Linguistics, 2016.

[165] Kurt Schütte. « Syntactical and Semantical Properties of Simple Type Theory ». *The Journal of Symbolic Logic*, **25**(4), pp. 305–326, 1960.

[166] Robert A. G. Seely. « Locally cartesian closed categories and type theory ». *Mathematical Proceedings of the Cambridge Philosophical Society*, **95**(1), pp. 33–48, 1984.

[167] Stuart M. Shieber. « Evidence against the context-freeness of natural language ». *Linguistics and Philosophy*, **8**(3), pp. 333–343, 1985.

[168] Ekaterina Shutova, Simone Teufel et Anna Korhonen. « Statistical Metaphor Processing ». *Computational Linguisitcs*, **39**(2), pp. 301–353, 2013.

[169] Scott Soames. « Ontology, Analyticity, and Meaning: the Quine–Carnap Dispute ». Dans David J. Chalmers, David Manley et Ryan Wasserman (réds.), *Metametaphysics: New Essays on the Foundations of Ontology*, chap. 14, pp. 424–443. Oxford University Press, 2009.

[170] Fred Sommers. « The Ordinary Language Tree ». *Mind*, **68**(2), pp. 160–185, 1959.

[171] Fred Sommers. « Type and Ontology ». *The Philosophical Review*, **72**(3), pp. 327–363, 1963.

[172] Fred Sommers. « A Program for Coherence ». *The Philophical Review*, **73**(4), pp. 522–527, 1964.

[173] Fred Sommers. « Predicability ». Dans Max Black (réd.), *Philosophy in America*, pp. 262–281. Cornell University Press, Ithaca, 1965.

[174] Fred Sommers. « Structural Ontology ». *Philosophia*, **1**(1-2), pp. 21–42, 1971.

[175] Jonathan Suzman. « The Ordinary language lattice ». *Mind*, **81**(3), pp. 434–436, 1972.

[176] Alfred Tarski. « Sur les ensembles définissables de nombres réels I ». *Fundamenta Mathematicæ*, **17**, pp. 210–239, 1931.

[177] Paul Taylor. *Practical Foundations of Mathematics*, vol. 59 de *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 1999.

[178] Robert D. Tennent. *Semantics of Programming Languages*. Prentice Hall, 1991.

[179] Manley Thompson. « Categories ». Dans Borchert [18].

[180] Alan M. Turing. « The Reform of Mathematical Notation and Phraseology ». Dans Gandy et Yates [67].

[181] Jean van Heijenoort (réd.). *From Frege to Gödel: A Source Book in Mathematical Logic, 1879-1931*. Harvard University Press, Cambridge, 1967.

[182] Loïc Vial, Benjamin Lecouteux et Didier Schwab. « Sense Vocabulary Compression through the Semantic Knowledge of WordNet for Neural Word Sense Disambiguation ». Dans Piek Vossen et Christiane Fellbaum (réds.), *Proceedings of the 10th Global Wordnet Conference*, pp. 108–117. Global WordNet Association, 2019.

[183] Piek Vossen et Ann Copestake. « Untangling Definition Structure into Knowledge Representation ». Dans Edward J. Briscoe, Valeria de Paiva et Ann Copestake (réds.), *Inheritance, Defaults and the Lexicon*, pp. 246–274. Cambridge University Press, 1993.

[184] Walter D. Wallis. *A Beginner's Guide to Graph Theory*. Birkhäuser, Boston, 2007.

[185] Howard Wettstein. « Frege-Russell Semantics? » *Dialectica*, **44**(1-2), pp. 113–135, 1990.

[186] Andrew Wilson et Jenny Thomas. « Semantic Annotation ». Dans Garside *et al.* [68], pp. 53–65.

[187] Ludwig Wittgenstein. *Tractatus Logico-Philosophicus*. Routlegde & Kegan Paul Ltd, London, 1921.

[188] Ludwig Wittgenstein. *Philosophical Investigations*. Macmillan, New York, 1953.

[189] Jan Wolenski. « Husserl and the development of semantics ». *Philosophia Scientiæ*, **2**(4), pp. 151–157, 1997.

[190] Elliot S. Wolk. « A note on "The comparability graph of a tree" ». *Proceedings of the American Mathematical Society*, **16**, pp. 17–20, 1965.

[191] Peter M. Worsley. « Noun-Classification in Australian and Bantu: Formal or Semantic? » *Oceania*, **24**(4), pp. 275–288, 1954.

[192] Tao Xue. « Definitional Extension in Type Theory ». Dans Ralph Matthes et Aleksy Schubert (réds.), *Proceedings of the 19th International Conference on Types for Proofs and Programs (TYPES 2013)*, vol. 26 de *LIPICS*, pp. 251–269. Dagstuhl Publishings, 2014.

[193] Wending Xue, Meichun Liu et Stephen Politzer-Ahles. « A study of complement coercion in Mandarin Chinese: evidence from an acceptability judgment task ». Dans Meichun Liu, Chunyu Kit et Qi Su (réds.), *Proceedings of the 21st Chinese Lexical Semantics Workshop, CLSW20, Hong Kong*, pp. 775–784. Springer, 2020.

Résumé

Cette thèse s'intéresse à l'utilisation des théories de types et des types eux-mêmes dans les formalismes sémantiques compositionnels en traitement automatique des langues. Les types sémantiques jouent un rôle essentiel dans la détection et la représentation de certains phénomènes sémantiques, incluant les usages créatifs de la langue, la polysémie et la coprédication, et nécessitent pour cela une certaine précision linguistique ainsi que des mécanismes théoriques capable de manipuler des structures complexes et des coercions de types. Afin de répondre à ces exigences, l'objectif de ce travail de thèse est de proposer une base minimaliste à la construction des théories de types sémantiques, qui soit capable dans une certaine mesure d'unifier les différentes approches actuelles à la sémantique lexicale et formelle. Une première partie est dédiée à l'examen des contraintes linguistiques qui pèsent sur la notion même de type sémantique, et aboutit à l'élaboration de principes généraux destinés à encadrer l'élaboration de théories de types sémantiques. Dans une seconde partie, ces principes sont confrontés aux fondements mathématiques de telles théories, conduisant à la construction d'une théorie de types dans un style montagovien, augmenté d'une relation de sous-typage et de coercions, à partir d'un modèle catégorique de topos. Enfin, une dernière partie traite du choix des types sémantiques de base, et tente d'évaluer expérimentalement si l'acquisition de tels types à partir de données empiriques est envisageable.

Mots-clés : sémantique formelle, sémantique lexicale, théorie des types, théorie des catégories, théorie des topos, coercion, compositionnalité, inférence de types.

Abstract

This thesis investigates the use of type theories and types with regards to compositionality in the formal semantic models of computational linguistics. Semantic types play a fundamental role in the detection and representation of semantic phenomena such as creative uses of language, polysemy and copredication. As a consequence, a formal account of types requires linguistic accuracy as well as theoretical mechanisms to manipulate complex structures and type coercions. In order to meet these expectations, the present thesis aims to propose a minimalist account of the construction of semantic type theories which would make it possible to unify the various current approaches to formal and lexical semantics. A first part is dedicated to the study of the linguistic constraints which influence the notion of semantic type itself, and leads to the formulation of general principles to supervise the production of semantic type theories. In a second part, these principles are interpreted in the light of the mathematical foundations of such theories, leading to the construction of a Montagovian-style type theory with subtyping and coercions from a topos as a categorical model. Finally, a third and last part deals with the choice of base semantic types in order to determine experimentally whether such types can be acquired from empirical data.

Keywords: formal semantics, lexical semantics, type theory, category theory, topos theory, coercion, compositionality, type inference.