THÈSE

en vue de l'obtention du titre de

DOCTEUR DE L'UNIVERSITÉ DE LORRAINE

(arrêté ministériel du 7 Août 2006)

Spécialité Mathématiques appliquées

présentée par

NGUYEN THI MINH TAM

Titre de la thèse :

Approches basées sur DCA pour la programmation mathématique avec des contraintes d'equilibre

—

DCA based Approaches for Mathematical Programs with Equilibrium Constraints

soutenue le 10 septembre 2018

Composition du Jury :

Rapporteurs	Mounir HADDOU	*Professeur, INSA Rennes*
	Pierre MARECHAL	*Professeur, Université Toulouse III - Paul Sabatier*
Examinateurs	Tao PHAM DINH	*Professeur, INSA de Rouen*
	Sophie HENNEQUIN	*MCF, HDR, Université de Lorraine*
Directrice de thèse	Hoai An LE THI	*Professeur, Université de Lorraine*

Thèse préparée au sein du LITA et du département Informatique & Applications, LGIPM, Université de Lorraine, Metz, France

Remerciements

Tout d'abord, je voudrais exprimer ma profonde reconnaissance à Madame Hoai An Le Thi, Professeur à l'Université de Lorraine et Directrice de ma thèse, pour m'avoir accordé l'opportunité de travailler avec elle au sein du LITA et du département Informatique & Applications, LGIPM, Université de Lorraine. Je la remercie très sincèrement de m'avoir donné des précieux conseils, de m'avoir encouragée et soutenue tout au long de ce travail avec patience, enthousiasme et rigueur. Je lui suis également très reconnaissante d'avoir corrigé mes articles et cette thèse. Ses commentaires critiques m'ont permis d'acquérir progressivement une maturité dans le domaine de la recherche scientifique. Sans son encadrement, il m'aurait été impossible de finaliser cette thèse.

Je souhaite ensuite exprimer toute ma gratitude à Monsieur Tao Pham Dinh, Professeur à l'INSA de Rouen, pour les connaissances précieuses et les documents très intéressants qu'il a partagés avec moi. Ses encouragements et ses conseils ont été importants pour cette recherche. Je voudrais le remercier aussi pour l'honneur qu'il me fait en participant au jury de ma thèse.

Je souhaite remercier vivement Monsieur Mounir Haddou, Professeur à l'INSA Rennes, et Monsieur Pierre Maréchal, Professeur à l'Université Toulouse III - Paul Sabatier pour m'avoir fait l'honneur d'accepter d'être rapporteurs de ma thèse et pour leurs temps précieux consacrés.

Je souhaite également remercier Madame Sophie Hennequin, Maître de Conférences à l'Université de Lorraine pour m'avoir fait l'honneur d'accepter d'être membre du jury.

Cette thèse a été réalisée au sein du LITA et du département Informatique & Applications, LGIPM, Université de Lorraine, Metz, France, où j'ai rencontré des personnes sympathiques et très gentilles. Merci à Minh Thuy, Bich Thuy, Tran Thuy, Hoai Minh, Duy Nhat, Vinh Thanh, Xuan Thanh, Tran Bach, Viet Anh, Dinh Chien, Nhu Tuan, Sara, Sarah, ... pour leurs soutiens et leurs encouragements, ainsi que pour les agréables moments passés ensemble lors de mon séjour en France. Je voudrais remercier particulièrement Docteur Vinh Thanh Ho et Docteur Duy Nhat Phan pour leurs aides et leurs grands soutiens tout au long de mes études. Je souhaite remercier Monsieur Van Ngai Huynh, Professeur à l'Université de Quy Nhon (Vietnam) pour ses encouragements et nos discussions scientifiques. Je remercie également Madame Annie Hetet, Secrétaire du département Informatique & Applications, pour sa grande disponibilité et son aide très spontanée.

Je souhaite remercier le Gouvernement Vietnamien qui a financé mes études en France. Je n'oublie pas de remercier toute l'équipe du personnel de l'Université nationale d'agriculture du Vietnam pour son soutien.

Enfin et surtout, je souhaite exprimer ma grande gratitude à tous les membres de ma famille: mes parents, mes beaux parents, mon mari, mes enfants, mes frères, mes belles soeurs pour leur soutien inconditionnel, leurs encouragements et leurs aides tout au long de ce parcours.

NGUYEN Thi Minh Tam

Née le 04 Octobre, 1979 (Viet Nam)

Tél : 07 83 28 80 88

E-mail : thi-minh-tam.nguyen@univ-loraine.fr

Adresse personnelle : D203, CU Technopole 3, 14 impasse des Linières, 57070, Metz, France

Adresse professionnelle : Bureau UM-AN1-34, Département Informatique & Applications, LGIPM, Université de Lorraine, 3 rue Augustin Fresnel, 57073 Metz, France

Situation Actuelle

Doctorante au département Informatique & Applications, LGIPM, Université de Lorraine. Encadrée par Prof. Hoai An Le Thi.

Sujet de thèse : **Approches basées sur DCA pour la programmation mathématique avec des contraintes d'équilibre**

Experience Professionnelle

2002–2014	Enseignante, Université nationale d'agriculture du Vietnam, Vietnam

Diplôme et Formation

2018 au present	Doctorante en Mathématiques appliquées, département Informatique & Applications, LGIPM, Université de Lorraine, Metz, France.
2015–2017	Doctorante en Mathématiques appliquées, Laboratoire d'Informatique Théorique et Appliquée (LITA), Université de Lorraine, Metz, France.
2001–2003	Master en Mathématiques, Ecole normale supérieure de Hanoi - Hanoi Ville, Vietnam.
1997–2001	Diplôme universitaire en Mathématiques, Ecole normale supérieure de Hanoi - Hanoi Ville, Vietnam.

Publications

Refereed international journal papers

[1] Hoai An Le Thi, Thi Minh Tam Nguyen, Tao Pham Dinh. On Solving Difference of Convex Functions Programs with Linear Complementarity Constraints *Submitted.*

[2] Thi Minh Tam Nguyen, Hoai An Le Thi. A DCA Approach for a Maximum Flow Network Interdiction Problem *Submitted.*

[3] Thi Minh Tam Nguyen, Hoai An Le Thi. DCA based Algorithms for Solving a Class of Mathematical Programs with Equilibrium Constraints *Submitted.*

Refereed papers in books / Refereed international conference papers

[1] Thi Minh Tam Nguyen, Hoai An Le Thi: A DC Programming Approach to the Continuous Equilibrium Network Design Problem. In: Nguyen et al. (eds) Advanced Computational Methods for Knowledge Engineering. ICCSAMA 2016. Advances in Intelligent Systems and Computing, Vol 453, pp.3-16, Springer, 2016.

Communications in national / International conferences

[1] Thi Minh Tam Nguyen, Hoai An Le Thi, Tao Pham Dinh. DC Programming and DCA for Solving Quadratic Programs with Linear Complemetarity Constraints. Presentation in the 18th French-German-Italian Conference on Optimization, Paderborn, Germany, September 25-28, 2017.

[2] Thi Minh Tam Nguyen, Hoai An Le Thi. DCA based Algorithms for Solving a Class of Mathematical Programs with Equilibrium Constraints. *Accepted* for presentation in the 29th European Conference on Operational Research, Spain, July 8-11, 2018.

Contents

List of Figures

List of Tables

Abbreviations and Notations

Throughout the dissertation, we use uppercase letters to denote matrices, and lowercase letters for vectors. Vectors are also regarded as matrices with one column. Some of the abbreviations and notations used in the dissertation are summarized as follows.

Abbreviations

CENDP	Continuous Equilibrium Network Design Problem
DC	Difference of Convex functions
DCA	DC Algorithm
ADCA	Accelerated DC Algorithm
DCLCC	DC program/Programming with Linear Complementarity Constraints
EiCP	Eigenvalue Complementarity Problem
KKT	Karush-Kuhn-Tucker
MFIN	Maximum Flow Interdiction Network
MILP	Mixed-Integer Linear Program
MPCC	Mathematical Program with Complementarity Constraints
MPLCC	Mathematical Program/Programming with Linear Complementarity Constraints
MPEC	Mathematical Program/Programming with Equilibrium Constraints
QPLCC	Quadratic Problem with Linear Complementarity Constraints
SBTP	Second-Best Toll Pricing
OD	Origin-Destination

Spaces

$\mathbb{R}$	the set of real numbers
$\overline{\mathbb{R}}$	the set of extended real numbers, $\overline{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$
$\mathbb{R}^n$	the set of real column vectors of size n
$\mathbb{R}^n_+$	the set of nonnegative real column vectors of size n
$\mathbb{R}^{m\times n}$	the set of real matrices of size m-by-n

Vectors

$z^\top$	the transpose of a vector z
$\{z^k\}$	a sequence of vectors $z^1, z^2, z^3, ...$
$x^\top y$ or $\langle x, y\rangle$	the standard inner product of vectors in $\mathbb{R}^n$
$\|x\|$	the Euclidean norm of a vector $x \in \mathbb{R}^n$
$\min(x, y)$	the vector whose i-th component is $\min(x_i, y_i)$
$x \circ y$	the Hadamard product of x and y

Matrices

$A^\top$	the transpose of a matrix A
$\|A\|$	the spectral norm of a matrix A
I	the identity matrix of appropriate order
I_k	the identity matrix of order k
$\det(A)$	the determinant of a matrix A
$\lambda_{\min}(A)$	the smallest eigenvalue of A
$\lambda_{\max}(A)$	the largest eigenvalue of A

Functions

∇f	the gradient of a function $f : \mathbb{R}^n \to \mathbb{R}$
f^*	the conjugate of f
$\partial f(x)$	the subdifferential of f at x
$\partial^\uparrow f(x)$	the Clarke subdifferential of f at x
$f^\uparrow(x, v)$	the Clarke derivative of f at x in the direction v
χ_C	the indicator function of a set C
$\Pi_C(x)$	the projection of a vector x onto a set C
$J_xF(x, y)$	the partial Jacobian matrix of a function $F : \mathbb{R}^{n+m} \to \mathbb{R}^p (p \geq 2)$ with respect to x.

Sets

$\operatorname{conv} C$	the convex hull of a set C
$\operatorname{dom} f$	the effective domain of a function f
$N_C(x)$	the normal cone of a set C at $x \in C$

Résumé

Dans cette thèse, nous étudions des approches basées sur la programmation DC (Difference of Convex functions) et DCA (DC Algorithm) pour la programmation mathématique avec des contraintes d'équilibre, notée MPEC (Mathematical Programming with Equilibrum Constraints en anglais). Etant un sujet classique et difficile de la programmation mathématique et de la recherche opérationnelle, et de par ses diverses applications importantes, MPEC a attiré l'attention de nombreux chercheurs depuis plusieurs années.

La thèse se compose de quatre chapitres principaux. Le chapitre 2 étudie une classe de programmes mathématiques avec des contraintes de complémentarité linéaire. En utilisant quatre fonctions de pénalité, nous reformulons le problème considéré comme des problèmes DC standard, i.e minimisation d'une fonction DC sous les contraintes convexes. Nous développons ensuite des algorithmes appropriés basés sur DCA pour résoudre les problèmes DC résultants. Deux d'entre eux sont reformulés encore sous la forme des problèmes DC généraux (i.e. minimisation d'une fonction DC sous des contraintes DC) pour que les sous-problèmes convexes dans DCA soient plus faciles à résoudre. Après la conception de DCA pour le problème considéré, nous développons ces schémas DCA pour deux cas particuliers: la programmation quadratique avec des contraintes de complémentarité linéaire, et le problème de complémentarité aux valeurs propres.

Le chapitre 3 aborde une classe de programmes mathématiques avec des contraintes d'inégalité variationnelle. Nous utilisons une technique de pénalisation pour reformuler le problème considéré comme un programme DC. Une variante de DCA et sa version accélérée sont proposées pour résoudre ce programme DC. Comme application, nous résolvons le problème de détermination du prix de péages dans un réseau de transport avec des demandes fixes ("the second-best toll pricing problem with fixed demands" en anglais).

Le chapitre 4 se concentre sur une classe de problèmes d'optimisation à deux niveaux avec des variables binaires dans le niveau supérieur. En utilisant une fonction de pénalité exacte, nous reformulons le problème considéré comme un programme DC standard pour lequel nous developpons un algorithme efficace basé sur DCA. Nous appliquons l'algorithme proposé pour résoudre le problème d'interdiction de flot maximum dans un réseau ("maximum flow network interdiction problem" en anglais).

Dans le chapitre 5, nous nous intéressons au problème de conception de réseau d'équilibre continu ("continuous equilibrium network design problem" en anglais). Il est modélisé sous forme d'un programme mathématique avec des contraintes de complémentarité, brièvement nommé MPCC (Mathematical Program with Complementarity Constraints en anglais). Nous reformulons ce problème MPCC comme un programme DC général et proposons un schéma DCA approprié pour le problème résultant.

Abstract

In this dissertation, we investigate approaches based on DC (Difference of Convex functions) programming and DCA (DC Algorithm) for mathematical programs with equilibrium constraints. Being a classical and challenging topic of nonconvex optimization, and because of its many important applications, mathematical programming with equilibrium constraints has attracted the attention of many researchers since many years.

The dissertation consists of four main chapters. Chapter 2 studies a class of mathematical programs with linear complementarity constraints. By using four penalty functions, we reformulate the considered problem as standard DC programs, i.e. minimizing a DC function on a convex set. The appropriate DCA schemes are developed to solve these four DC programs. Two among them are reformulated again as general DC programs (i.e. minimizing a DC function under DC constraints) in order that the convex subproblems in DCA are easier to solve. After designing DCA for the considered problem, we show how to develop these DCA schemes for solving the quadratic problem with linear complementarity constraints and the asymmetric eigenvalue complementarity problem.

Chapter 3 addresses a class of mathematical programs with variational inequality constraints. We use a penalty technique to recast the considered problem as a DC program. A variant of DCA and its accelerated version are proposed to solve this DC program. As an application, we tackle the second-best toll pricing problem with fixed demands.

Chapter 4 focuses on a class of bilevel optimization problems with binary upper level variables. By using an exact penalty function, we express the bilevel problem as a standard DC program for which an efficient DCA scheme is developed. We apply the proposed algorithm to solve a maximum flow network interdiction problem.

In chapter 5, we are interested in the continuous equilibrium network design problem. It was formulated as a Mathematical Program with Complementarity Constraints (MPCC). We reformulate this MPCC problem as a general DC program and then propose a suitable DCA scheme for the resulting problem.

Introduction générale

Cadre général et motivations

La programmation mathématique avec des contraintes d'équilibre, notée MPEC (Mathematical Programming with Equilibrum Constraints en anglais) est une classe de problèmes d'optimisation dont les contraintes contiennent une inégalité variationnelle paramétrique. Plus précisément, le problème est décrit comme suit:

$$\begin{array}{lll} \text{(P)} & \min & f(x,y) \\ & \text{s.t.} & (x,y) \in \Omega, \\ & & x \in C(y),\ (v-x)^T F(x,y) \geq 0, \forall v \in C(y), \qquad \text{(VI)} \end{array}$$

où Ω est un sous-ensemble fermé, non vide de $\mathbb{R}^{n+m}$, $f : \mathbb{R}^{n+m} \to \mathbb{R}, F : \mathbb{R}^{n+m} \to \mathbb{R}^n$ sont des fonctions données, $C : \mathbb{R}^m \to 2^{\mathbb{R}^n}$ est une application multivoque à valeurs convexes fermées, c'est-à-dire, pour tout y dans $\mathbb{R}^m$, $C(y)$ est un sous-ensemble convexe fermé de $\mathbb{R}^n$.

La formulation ci-dessus du MPEC englobe de nombreuses classes de problèmes dont la plus importante est celle où $C(y)$ est l'orthant non négatif de $\mathbb{R}^n$ pour tout y dans Y, la projection de Ω sur $\mathbb{R}^m$. Dans ce cas, les contraintes d'inégalité variationnelle (VI) (Variational Inequality en anglais) du problème (P) sont équivalentes au système de complémentarité suivant

$$x \geq 0, F(x,y) \geq 0, x^T F(x,y) = 0,$$

et le problème (P) devient ainsi *un programme mathématique avec des contraintes de complémentarité*, brièvement nommé MPCC (Mathematical Program with Complementarity Constraints en anglais). De plus, sous des hypothèses appropriées, le problème (P) peut être formulé sous la forme d'un MPCC [66]. Les MPCCs constituent donc une sous-classe primordiale de MPEC, et la plupart des travaux dans la littérature sur la MPEC sont dédiés à MPCCs. Un cas particulier important du MPCC est la programmation mathématique avec des contraintes de complémentarité linéaire (MPLCC) qui consiste à minimiser une fonction continûment différentiable sur un ensemble défini par des contraintes linéaires et des contraintes de complémentarité linéaire. De nombreuses applications peuvent être modélisées sous la forme MPLCC. De plus, certains problèmes d'optimisation NP-difficiles peuvent être reformulés comme MPLCCs.

Par ailleurs, dans le cas où $F(x, y)$ est la dérivée partielle par rapport à la variable x d'une fonction à valeur réelle $\theta(x, y)$ qui est convexe en x, l'ensemble des valeurs de x satisfaisant les contraintes (VI) dans (P) est l'ensemble des solutions optimales du problème suivant en la variable x

$$\min\{\theta(x, y) : \; x \in C(y)\}.$$

Ce cas particulier de MPEC est connu comme *un programme mathématique à deux niveaux.*

MPEC a de nombreuses applications en ingénierie et en économie. Cependant, il est connu que ce problème est très difficile, même lorsque la fonction objectif est linéaire. Les contraintes non convexes d'inégalité variationnelle (resp. de complémentarité) sont considérées comme la cause principale de la difficulté de MPEC (resp. MPCC). De plus, tous les solutions réalisables de MPEC ne vérifient pas la qualification des contraintes de Mangasarian-Fromovitz, par conséquent la majorité des algorithmes standard pour la programmation non linéaire ne peuvent pas être directement appliqués à MPEC.

Etant un sujet classique et difficile de la programmation mathématique et de la recherche opérationnelle, et de par ses diverses applications importantes, MPEC a attiré l'attention de nombreux chercheurs depuis plusieurs années. Un grand nombre de travaux ont été développés pour la résolution de MPEC (dont la plupart concernent MPCC). Pour surmonter la difficulté principale liée aux contraintes d'inégalité variationnelle/de complémentarité, de nombreuses reformulations de MPEC ont été introduites. Citons les approches populaires comme les techniques de relaxation de ces contraintes (ensembliste ou fonctionnelle, voir par exemple [12, 37, 38, 59, 91, 94, 98]), les techniques de lissage (remplacer ces contraintes par une suite de contraintes différentiables, voir par exemple [15, 22, 25, 56]), et les techniques de pénalisation (utiliser une fonction de pénalité pour reformuler ces contraintes puis pénaliser la nouvelle contrainte dans la fonction objectif, voir par exemple [29, 30, 58, 67, 92] et les références citées ci-dessous concernant les méthodes basées sur la programmation DC (Difference of Convex functions) et DCA (DC algorithm)). Plusieurs méthodes pour la programmation non linéaire ont été également adaptées à la résolution de MPEC, par exemple, celles basées sur la programmation quadratique séquentielle (voir par exemple [21, 32, 62, 66, 85]), les méthodes de points intérieurs (voir par exemple [53, 64, 66, 87]). Ces approches ne trouvent qu'un point stationnaire de MPEC. Les algorithmes exactes afin de trouver une solution optimale ont été principalement développés pour MPEC avec des contraintes d'inégalité variationnelle affine et/ou MPLCC dans lesquelles la fonction objectif est convexe. La plupart d'entre eux sont basés sur les méthodes par Séparation et Evaluation ("Branch-and-Bound" en anglais) (voir par exemple [4, 6, 36, 63, 72, 73, 80, 106]). En outre, divers méthodes heuristiques sont développées telles que celles basées sur l'analyse de sensibilité [20] et le recuit simulé [19]. En général, il est très difficile de trouver une solution optimale de MPEC.

Parmi les approches citées plus haut, nous sommes particulièrement intéressés par les techniques de pénalisation basées sur la programmation DC et DCA qui sont reconnus comme des outils puissants d'optimisation non convexe. Durant ces dix dernières années, ces outils ont été exploités pour traiter plusieurs cas particuliers de MPEC (voir par exemple [31, 44, 46, 49, 51, 75, 76]). La programmation DC et DCA constituent l'épine dorsale de l'optimisation non convexe et de l'optimisation globale. Ils ont été introduits par Pham Dinh Tao dans leur forme préliminaire en 1985 et intensivement développés, tant sur les aspects théoriques qu'algorithmiques, depuis 1994 à travers de nombreux travaux conjoints de Le Thi Hoai An et Pham Dinh Tao pour devenir maintenant classiques et de plus en plus populaires. Un problème DC standard est de la forme

$$\alpha = \inf\{f(x) := g(x) - h(x) : \ x \in \mathbb{R}^n\},$$

où g et h sont des fonctions convexes définies sur $\mathbb{R}^n$ et à valeurs dans $\mathbb{R} \cup \{+\infty\}$, semi-continues inférieurement et propres. La fonction f est appelée fonction DC avec les composantes DC g et h, et l'expression $g - h$ est appelée une décomposition DC de f. DCA est basé sur la dualité DC et des conditions d'optimalité locale. La construction de DCA implique les composantes DC g et h et non la fonction DC f elle-même. Chaque fonction DC admet une infinité des décompositions DC qui influencent considérablement la qualité (la rapidité, l'efficacité, la globalité de la solution obtenue, etc.) de DCA. Ainsi, au point de vue algorithmique, la recherche d'une "bonne" décomposition DC est cruciale.

L'utilisation de la programmation DC et DCA dans cette thèse est motivée/justifiée par plusieurs raisons [47, 82]:

- DCA a été appliqué avec succès à de nombreux programmes non convexes de grandes dimensions dans divers domaines des sciences appliquées (voir par exemple la liste de références dans [42, 47]). En particulier, DCA a déjà résolu efficacement plusieurs cas particuliers du MPEC tels que la programmation à deux niveaux [49, 51], des problèmes de complémentarité linéaire [46], des programmes linéaires avec des contraintes de complémentarité linéaire [31] et des problèmes de complémentarité aux valeurs propres [44, 75, 76]. Bien qu'il s'agisse d'une approche d'optimisation locale, DCA fournit souvent une solution globale et s'avère plus robuste et efficace que les méthodes standard.
- DCA est une philosophie plutôt qu'un algorithme. Pour chaque problème, nous pouvons concevoir une famille d'algorithmes basés sur DCA. La flexibilité de DCA sur le choix des décomposition DC peut offrir des schémas DCA plus performants que des méthodes standard.
- L'analyse convexe fournit des outils puissants pour prouver la convergence de DCA dans un cadre général. Donc, tous les algorithmes basés sur DCA bénéficient (au moins) des propriétés de convergence générales du schéma DCA générique qui ont été démontrées.

Il est important de noter qu'avec les techniques de reformulation en programmation DC et les décompositions DC appropriées, la plupart des algorithmes existants en optimisation convexe/non convexe sont retrouvés comme cas particuliers de DCA.

Nos contributions

Dans cette thèse, nous étudions les approches basées sur la programmation DC et DCA pour résoudre les trois sous-classes suivantes de MPEC:

- La première classe est MPLCC dans laquelle la fonction objectif est une fonction DC. Cette classe de problèmes est appelée la programmation DC avec des contraintes de complémentarité linéaires, notée brièvement DCLCC (DC programming with Linear Complementarity Constraints en Anglais). Puisque presque toutes les fonctions non convexes sont DC et une fonction convexe peut être également considérée comme une fonction DC, DCLCC constitue la plus large classe des problèmes MPLCC. Et jusqu'à maintenant aucune approche dans la littérature n'aborde cette générale classe DCLCC.
- La seconde classe est MPEC dans laquelle la fonction objectif est une fonction DC, $\Omega = X \times Y$ où X est un polyèdre borné et Y un ensemble convexe compact, F est une fonction continûment différentiable dont les matrices jacobiennes partielles sont Lipschitziennes, et $C(y) = X$. En plus du cas général, nous étudions un cas particulier où la fonction objectif est une fonction continûment différentiable dont le gradient est Lipschitzien. Cette étude est motivée par des nombreuses applications importantes de ce modèle, en particulier le problème de détermination du prix de péages dans un réseau de transport avec des demandes fixes.
- La troisième classe est la programmation mathématique à deux niveaux où la fonction objectif au niveau supérieur est la somme d'une fonction convexe et la fonction valeur d'un programme linéaire, et les variables dans ce niveau sont binaires. Cette étude est suscitée par des applications importantes de type d'allocation de ressource dans plusieur domaines dont dans la sécurité et l'armée dont le problème d'interdiction des flots maximum dans un réseau.

En plus des trois classes de probblèmes ci-dessus, nous étudions également un challenge dans les réseaux de transport - la conception de réseau d'équilibre continu (Continuous equilibrium network design problem) sous la forme d'un MPCC difficile où la fonction objectif est fractionnaire et certaines contraintes sont non convexes (en plus de contraint de complémantarité).

Grâce aux techniques de pénalité, nous reformulons les problèmes considérés comme des programmes DC et développons des algorithmes basés sur DCA pour leur résolution. Nous donnons ci-après une description détaillée des nos contributions.

Concernant la première classe de problèmes (DCLCC), comme cela a été indiqué plus haut, la difficulté principale réside dans les contraintes de complémentarité. Afin de surmonter cette difficulté, nous introduisons quatre fonctions de pénalité (trois parmi elles sont nouvelles parmi les travaux utilisant la pénalité pour MPLCC) pour remplacer les contraintes de complémentarité par des nouvelles contraintes, puis pénalisons ces dernières à la fonction objectif. Par la suite, le DCLCC est reformulé comme des problèmes DC standard, i.e minimisation d'une fonction DC sous les contraintes convexes. Nous prouvons que la pénalisation est exacte (dans le cas où l'ensemble de contraintes linéaires est borné), c.à.d le problème original est équivalent aux problèmes pénalisés. Ces résultats constituent des contributions cruciales sur le plan théorique qui permettent de faire le pont entre MPLCC et la programmation DC. Nous développons ensuite des algorithmes appropriés basés sur DCA pour résoudre les problèmes DC résultants. Deux d'entre eux sont reformulés encore sous la forme des problèmes DC généraux (i.e. minimisation d'une fonction DC sous des contraintes DC) pour que les sous-problèmes convexes dans DCA soient plus faciles à résoudre. La convergence des schémas DCA est soigneuseument étudiée. Nous montrons que, sous certaines hypothèses raisonnables, chaque point d'accumulation de la suite générée par toutes les versions de DCA, est un point stationnaire de DCLCC. De plus, quand la fonction objectif est convexe, certains schémas DCA ont des propriétés de convergence intéressantes telles que la convergence finie, l'optimalité locale, ...

Après la conception de DCA pour DCLCC, nous développons ces schémas DCA pour deux cas particuliers: la programmation quadratique avec des contraintes de complémentarité linéaire et le problème de complémentarité aux valeurs propres. Ce dernier problème est reformulé en DCLCC dans lequel la fonction objectif n'est ni convexe ni quadratique. Les versions DCA correspondantes sont simples, elles consistent à résoudre successivement des problèmes quadratiques convexes à contraintes linéaires. Nous les testons sur nombreux jeux de données, dont plusieurs ont un grand nombre de contraintes de complémentarité (de 512 à 3200). Les résultats numériques montrent l'efficacité de nos algorithmes et leur supériorité par rapport au solveur KNITRO (un solveur avancé pour des problèmes d'optimisation non linéaires, incluant MPECs) et à un autre schéma DCA dans la littérature pour le problème de complémentarité aux valeurs propres.

En résumé, nos contributions significatives dans cette partie portent à la fois sur la théorie et les algorithmes, et ces résultats sont validés par nombreuses expérimentations numériques sur deux problèmes difficiles de DCLCC. Nous offrons quatre schémas DCA génériques résolvant la plus large classe des problèmes MPLCCs.

La seconde classe de problèmes considérée dans cette thèse est également assez large pour couvrir nombreux problèmes rencontrés en pratique, car la fonction objectif est DC. Si la pénalité exacte (avec les fonctions de pénalité proposées) a été prouvée pour DCLCC, il n'en n'est pas de même pour cette seconde classe de problèmes de MPEC. Plus précisément, la pénalité exacte est valide pour certains cas particuliers (par exemple, pour une fonction objectif continûment differentiable, [68]) et non pour le cas général où l'objectif est une fonction DC. De plus, la fonction de pénalité exacte utilisée dans [68] n'est pas favorable à l'utilisation de DCA car il est difficile à mettre en évidence une formulation DC du problème pénalisé. Nous utilisons la même fonction de pénalité inexacte introduite dans [68] pour pénaliser la contrainte d'inégalité variationnelle, cela nous permet de reformuler le problème pénalisé comme un programme DC auquel DCA peut être appliqué. Malgré que la pénalité exacte n'est pas vérifiée pour cette fonction de pénalité, il a été prouvé qu'avec un paramèttre de pénalité suffisament grand, les solutions optimales du problème original se trouvent dans la région où la valeur de la fonction pénalitée est petite. Ceci justifie les techniques de pénalisation pour ce problème. Il est à noter qu'il existe très peu de travaux dans la litérature proposant des méthodes numériques pour MPEC basées sur les techniques de pénalisation, en particulier pour le cas où l'objectif est une fonction DC. La nouveauté et l'originalité de notre travail réside dans le développement des algorithmes avancés basés sur DCA - DCA_ρ et ADCA_ρ (Accelerated DCA_ρ), qui sortent du cadre de DCA standard. En fait, la formulation DC nécessite le calcul d'un paramètre ρ qui peut être obtenu via la constante de Lipschitz du gradient de la fonction de pénalité. En pratique, cette constante de Lipschitz est généralement estimée par une valeur assez élevée qui pourrait rendre le DCA inefficace. Ainsi, nous proposons une variante de DCA (DCA_ρ), dans laquelle une mise à jour de ρ est effectuée à chaque itération et la convexité de H (la second fonction dans la décomposition de l'objectif) n'est pas nécessaire [50]. De plus, étant motivé par le succès des algorithmes accélérés en optimisation convexe / non convexe récemment développés pour améliorer le taux de convergence (voir par exemple [8, 55, 84, 104, 50]), nous offrons également une version accélérée de l'algorithme proposé qui incorpore une étape d'extrapolation. Nous prouvons que la convergence de DCA reste valable pour ces versions avancées, c.à.d. chaque point d'accumulation de la suite générée par nos algorithmes est un point critique du problème pénalisé.

Après l'étude du cas général, nous considérons un cas particulier où la fonction objectif est une fonction continûment différentiable dont le gradient est Lipschitzien. Les versions DCAs correspondantes sont assez simples, elles nécessitent le calcul de projection d'un point sur un polyèdre et/ou sur un ensemble compact. Comme application, nous résolvons le problème de détermination du prix de péages dans un réseau de transport avec des demandes fixes ("the second-best toll pricing problem with fixed demands" en anglais). Les résultats numériques sur plusieurs données indiquent que nos approches sont prometteuses.

Pour la troisième classe de problèmes, la difficulté se trouve non seulement dans la fonction objectif non convexe, mais aussi dans les variables binaires. Bien que la programmation à deux niveaux peut être considérée comme un cas particulier de MPEC, notre technique de résolution est différente, elle ne doit pas passer par MPEC. En fait, la structure particulière du problème, à savoir la fonction valeur d'un programme linéaire et les variables binaires, nous permettent de montrer que sa fonction objectif est une fonction DC, et ce problème à deux niveaux devient un problème DC (à un niveau) avec des variables binaires. Dès lors, en utilisant des nouveaux résultats de la pénalité exacte pour la programmation DC [48], nous pénalisons les contraintes binaires et reformulons ainsi ce dernier problème en un problème DC standard pour lequel nous proposons ensuite un algorithme approprié basé sur DCA. En outre, nous montrons que lorsque la partie convexe dans la fonction objectif est linéaire, nous pouvons choisir le paramètre de pénalité comme un nombre positif arbitraire, et la suite générée par DCA contient toujours des valeurs binaires, Nous appliquons l'algorithme proposé pour résoudre le problème d'interdiction de flot maximum dans un réseau ("maximum flow network interdiction problem" en anglais). Il s'agit d'une application très importante dans le domaine d'allocation de ressources. Le schéma DCA pour ce problème est simple, il consiste à résoudre successivement des programmes linéaires avec contraintes de boîte dont les solutions sont binaires. Afin d'évaluer la qualité de la solution trouvée par DCA, nous reformulons le problème d'application comme un programme linéaire en variables mixtes binaires qui peut être résolu globalement par le logiciel CPLEX. Les résultats numériques montrent que DCA fournit souvent une solution globale en peu de temps.

Le dernier problème étudié dans cette thèse est une application de MPCC. C'est un des problèmes les plus difficiles dans le domaine de transport, à savoir la conception de réseau d'équilibre continu ("continuous equilibrium network design problem" en anglais). Ce problème consiste à déterminer les extensions de capacité des liaisons existantes afin de minimiser le coût total de déplacement plus le coût d'investissement pour ces extensions, lorsque les flux de liaison sont sous contraintes d'équilibre. Il est modélisé sous forme d'un MPCC dans [100]. La double difficulté de ce MPCC vient des contraintes de complémentarité et des fonctions de coût de déplacement non convexes qui sont des fonctions fractionnaires. En introduisant de nouvelles variables et basant sur une technique de pénalisation, nous transformons le problème MPCC en un programme DC général pour lequel nous développons un DCA approprié.

Organisation de la Thèse

La thèse est composée de cinq chapitres.

- Le chapitre 1 présente des outils théoriques et algorithmiques servant des références aux autres. Il s'agit de la Programmation DC et DCA ainsi que des résultats concernant des techniques de pénalisation en optimisation non convexe, et plus particulièrement en programmation DC.
- Le chapitre 2 étudie les méthodes de résolution basées sur la programmation DC et DCA pour DCLCC en général, et pour la programmation quadratique avec contraintes de complémentarité linéaire ainsi que le problème de complémentarité aux valeurs propres asymétriques en particulier.
- Le chapitre 3 concerne la deuxième classe de problème de MPEC, pour le cas général où l'objectif est une fonction DC et pour le cas particulier où la fonction objectif est une fonction continûment différentiable dont le gradient est Lipschitzien ainsi que son application au problème de détermination du prix de péages dans un réseau de transport avec des demandes fixes.
- Le chapitre 4 se rapporte à la programmation à deux niveaux avec des variables binaires dans le niveau supérieur et son application à un problème d'interdiction de flots maximum.
- Enfin, le chapitre 5 étudie le problème de conception de réseau d'équilibre continu via MPCC.

Chapter 1

Preliminary

This chapter presents a brief introduction to DC programming and DCA, and some results concerning the penalty techniques for nonconvex optimization problems.

1.1 DC programming and DCA

In this section, we first recall some basic properties of convex analysis and then present some main points of DC programming and DCA. These contents are extracted from [41, 43, 45, 81, 82, 86, 88].

Throughout this section, X denotes the Euclidean space $\mathbb{R}^n$ and $\overline{\mathbb{R}} = \mathbb{R} \cup \{\pm\infty\}$ is the set of extended real numbers.

1.1.1 Fundamental convex analysis

A subset C of X is said to be *convex* if $(1-\lambda)x + \lambda y \in C$ for any x, $y \in C$ and any $\lambda \in [0, 1]$.

The *convex hull* of a set C, denoted by conv$C$ is the set of all convex combinations of points in C.

Let C be a convex set. A function $f : C \to (-\infty, +\infty]$ is said to be *convex* on C if

$$f((1-\lambda)x + \lambda y) \leq (1-\lambda)f(x) + \lambda f(y), \quad \forall x, y \in C, \ \forall \lambda \in [0, 1].$$

A real-valued function f on a convex set C is said to be *strictly convex* on C if the inequality above holds strictly whenever $x \neq y$ and $0 < \lambda < 1$.

The *effective domain* of a convex function f on C, denoted by dom$f$, is the set

$$\text{dom}f = \{x \in X : f(x) < +\infty\}$$

Clear, dom$f$ is a convex set in X.

A convex function f is called *proper* if dom$f \neq \emptyset$ and $f(x) > -\infty$ for all x.

A function $f : C \to [-\infty, +\infty]$ is said to be *lower semi-continuous* at a point x of C if

$$f(x) \leq \liminf_{y \to x} f(y).$$

Denote by $\Gamma_0(X)$ the set of all proper lower semi-continuous convex functions on X.

Let ρ be a nonnegative number and C be a convex subset of X. A function $\theta : C \to (-\infty, +\infty]$ is said to be ρ–convex if

$$\theta[\lambda x + (1-\lambda)y] \leq \lambda\theta(x) + (1-\lambda)\theta(y) - \frac{\lambda(1-\lambda)}{2}\rho\|x-y\|^2$$

for all $x, y \in C$ and $\lambda \in (0, 1)$.

It is easy to see that θ is ρ–convex if and only if $\theta - (\rho/2)\|\cdot\|^2$ is convex on C.

The modulus of strong convexity of θ on C, denoted by $\rho(\theta, C)$ or $\rho(\theta)$ if $C = X$, is given by

$$\rho(\theta, C) = \sup\{\rho \geq 0 : \theta - (\rho/2)\|\cdot\|^2 \text{ is convex on } C\}.$$

θ is said to be *strongly convex* on C if $\rho(\theta, C) > 0$.

A vector y is said to be a *subgradient* of a convex function f at a point x^0 if

$$f(x) \geq f(x^0) + \langle x - x^0, y\rangle, \quad \forall x \in X.$$

The set of all subgradients of f at x^0 is called the *subdifferential* of f at x^0 and is denoted by $\partial f(x^0)$. If $\partial f(x)$ is not empty, f is said to be *subdifferentiable* at x.
The *effective domain* of ∂f, denoted by dom ∂f is the set

$$\text{dom } \partial f = \{x \in X : \partial f(x) \neq \emptyset\}.$$

For $\varepsilon > 0$, a vector y is said to be an *ε–subgradient* of a convex function f at a point x^0 if

$$f(x) \geq (f(x^0) - \varepsilon) + \langle x - x^0, y\rangle, \quad \forall x \in X.$$

The set of all ε–subgradients of f at x^0 is called the *ε–subdifferential* of f at x^0 and is denoted by $\partial_\varepsilon f(x^0)$.

Proposition 1.1. *Let f be a proper convex function. Then*

1. *$\partial_\varepsilon f(x)$ is a closed convex set, for any $x \in X$ and $\varepsilon \geq 0$.*
2. ri(dom$f$) $\subset$ dom $\partial f \subset$ dom$f$
   where ri(dom$f$) *stands for the* relative interior *of* dom$f$.
3. *If f is differentiable at $x \in$ dom$f$, then $\partial f(x) = \{\nabla f(x)\}$.*
4. $x_0 \in$ argmin$\{f(x) : x \in X\}$ *if and only if* $0 \in \partial f(x_0)$.

Let C be a nonempty convex subset of $\mathbb{R}^n$. The *indicator function* of C, denoted by χ_C, is the function

$$\chi_C(x) = \begin{cases} 0 & \text{if } x \in C \\ +\infty & \text{otherwise} \end{cases} \tag{1.1}$$

The *normal cone* of C at $x \in C$, denoted $N_C(x)$, is given by

$$N_C(x) = \partial\chi_C(x) = \{u \in \mathbb{R}^n : \langle u, y - x\rangle \le 0 \ \forall y \in C\}.$$

A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is said to be *Lipschitz continuous* on C if there exists a real number $\lambda \ge 0$ such that

$$\|f(x_1) - f(x_2)\| \le \lambda\|x_1 - x_2\| \quad \forall x_1, x_2 \in C.$$

Such a number λ is called a Lipschitz constant of f on C.

A function $f : \mathbb{R}^n \to (-\infty, +\infty]$ is said to be *locally Lipschitz* at $x \in \mathbb{R}^n$ if there exists a neighborhood U_x of x such that f is Lipschitz continuous on U_x.

Let $f : \mathbb{R}^n \to (-\infty, +\infty]$ be a locally Lipschitz function at a given $x \in \mathbb{R}^n$. The *Clarke directional derivative* and the *Clarke subdifferential* of f at x is given by the following formulas.

$$\begin{aligned} f^{\uparrow}(x, v) &= \limsup_{(t,y)\to(0^+,x)} \frac{f(y + tv) - f(y)}{t}, \\ \partial^{\uparrow} f(x) &= \left\{x^* \in \mathbb{R}^n : \langle x^*, v\rangle \le f^{\uparrow}(x, v) \ \ \forall v \in \mathbb{R}^n\right\}. \end{aligned}$$

If f is continuously differentiable at x then $\partial^{\uparrow} f(x) = \nabla f(x)$. When f is a convex function, then $\partial^{\uparrow} f(x)$ coincides with the subdifferential $\partial f(x)$.

Conjugates of convex functions

The *conjugate* of a function $f : X \to \overline{\mathbb{R}}$ is the function $f^* : X \to \overline{\mathbb{R}}$, defined by

$$f^*(y) = \sup_{x\in X}\{\langle x, y\rangle - f(x)\}.$$

Proposition 1.2. *Let $f \in \Gamma_0(X)$. Then we have*

1. *$f^* \in \Gamma_0(X)$ and $f^{**} = f$.*
2. *$f(x) + f^*(y) \ge \langle x, y\rangle$, for any $x, y \in X$.*
3. *$f(x) + f^*(y) = \langle x, y\rangle \Leftrightarrow y \in \partial f(x) \Leftrightarrow x \in \partial f^*(y)$.*

Polyhedral Functions

A *polyhedral* set is a closed convex set that has the form

$$C = \{x \in X : \langle b_i, x\rangle \le \beta_i, \ \forall i = 1, \dots, m\}$$

where $b_i \in X$ and $\beta_i \in \mathbb{R}$ for all $i = 1, \ldots, m$.

A function $f \in \Gamma_0(X)$ is said to be *polyhedral* if

$$f(x) = \max\{\langle a_i, x\rangle - \alpha_i : i = 1, \ldots, k\} + \chi_C(x), \quad \forall x \in X \tag{1.2}$$

where $a_i \in X$, $\alpha_i \in \mathbb{R}$ for all $i = 1, \ldots, k$ and C is a nonempty polyhedral set. It is clear that $\operatorname{dom} f = C$.

Proposition 1.3. [86] *Let f be a polyhedral convex function, and $x \in \operatorname{dom} f$. The following statements hold.*

1. *f is subdifferentiable at x, and $\partial f(x)$ is a polyhedral convex set. In particular, if f is defined by (1.2) with $C = X$ then*

$$\partial f(x) = \operatorname{conv}\{a_i : i \in I(x)\}$$

   where $I(x) = \{i \in \{1, \ldots, k\} : \langle a_i, x\rangle - \alpha_i = f(x)\}$.

2. *The conjugate f^* is a polyhedral convex function. Moreover, if $C = X$ then*

$$\operatorname{dom} f^* = \operatorname{conv}\{a_i : i = 1, \ldots, k\},$$
$$f^*(y) = \inf\left\{\sum_{i=1}^k \lambda_i \alpha_i : \sum_{i=1}^k \lambda_i a_i = y, \sum_{i=1}^k \lambda_i = 1, \lambda_i \geq 0, \forall i = 1, \ldots, k\right\}.$$

   In particular,

$$f^*(a_i) = \alpha_i, \quad \forall i = 1, \ldots, k.$$

DC functions

A function f is called *DC function* on X if it is of the form

$$f(x) = g(x) - h(x), \quad x \in X$$

where $g, h \in \Gamma_0(X)$. One says that $g - h$ is a *DC decomposition* of f and the functions g, h are its *DC components*. If, in addition, g and h are finite at all points of X then f is said to be a finite DC function on X. The set of DC functions (resp. finite DC functions) on X is denoted by $\mathcal{DC}(X)$ (resp. $\mathcal{DC}_f(X)$).

Remark 1.1. *If f is a DC function with DC decomposition $f = g - h$ then for every $\theta \in \Gamma_0(X)$ finite on X, $f = (g + \theta) - (h + \theta)$ is another DC decomposition of f. Thus, a DC function has infinitely many DC decompositions.*

1.1.2 Standard DC optimization

Standard DC program

In the sequel, we use the convention $+\infty - (+\infty) = +\infty$.

A so-called *standard DC program* takes the form

$$(P) \qquad \alpha = \inf\{f(x) := g(x) - h(x) : \ x \in X\},$$

where $g, h \in \Gamma_0(X)$.

Remark 1.2. *The constrained DC program whose feasible set C is closed convex always can be converted into the unconstrained DC program by adding the indicator function χ_C of C to the first DC component, i.e.*

$$\inf\{f(x) := g(x) - h(x) : x \in C\} = \inf\{g(x) + \chi_C(x) - h(x) : x \in X\}.$$

The dual program of (P) is also a DC program with the same optimal value and defined by

$$(D) \qquad \alpha = \inf\{h^*(y) - g^*(y) : \ y \in X\}.$$

It is noted that there is a perfect symmetry between primal and dual programs (P) and (D): the dual program of (D) is exactly (P).

We will always keep the following assumption that is deduced from the finiteness of α

$$\operatorname{dom} g \subset \operatorname{dom} h \ \text{ and } \ \operatorname{dom} h^* \subset \operatorname{dom} g^*. \tag{1.3}$$

Polyhedral DC program

In the problem (P), if one of the DC components g and h is polyhedral, we call (P) a *polyhedral DC program.*

Optimality conditions for DC standard programs

A point x^* is said to be a *local minimizer* of $g - h$ if $x^* \in \operatorname{dom} g \cap \operatorname{dom} h$ and there is a neighborhood U of x^* such that

$$g(x) - h(x) \geq g(x^*) - h(x^*), \qquad \forall x \in U. \tag{1.4}$$

A point x^* is said to be a *critical point* of $g - h$ if

$$\partial g(x^*) \cap \partial h(x^*) \neq \emptyset. \tag{1.5}$$

Optimality conditions for DC standard programs are shown in the following theorem (see [81]).

Theorem 1.1. *i) Global optimality condition:*
x^* is an optimal solution of the problem (P) if and only if

$$\partial_\varepsilon h(x^*) \subset \partial_\varepsilon g(x^*), \ \forall \varepsilon > 0.$$

ii) Necessary condition for local optimality :
if x^* is a local minimizer of $g - h$, then

$$\partial h(x^*) \subset \partial g(x^*).$$

iii) Sufficient condition for local optimality:
Let x^* be a critical point of $g - h$ and $y^* \in \partial g(x^*) \cap \partial h(x^*)$. Let U be a neighborhood of x^* such that $(U \cap \operatorname{dom} g) \subset \operatorname{dom} \partial h$. If for any $x \in U \cap \operatorname{dom} g$, there is $y \in \partial h(x)$ such that $h^*(y) - g^*(y) \geq h^*(y^*) - g^*(y^*)$, then x^* is a local minimizer of $g - h$. More precisely,

$$g(x) - h(x) \geq g(x^*) - h(x^*), \qquad \forall x \in U \cap \operatorname{dom} g.$$

Remark 1.3. *a) By the symmetry of the DC duality, Theorem* 1.1 *has its corresponding dual part.*

b) The necessary local optimality condition $\partial h(x^*) \subset \partial g(x^*)$ is also sufficient for many important classes of programs, for example, if h is polyhedral convex, or when f is locally convex at x^*, i.e. there exists a convex neighborhood U of x^* such that f is finite and convex on U. We know that a polyhedral convex function is differentiable everywhere except on a set of measure zero. Thus, if h is a polyhedral convex function, then a critical point of $g - h$ is almost always a local solution to (P).

c) If $f = g - h$ is actually convex on X, we call (P) a "false" DC program. *Furthermore, if* $\operatorname{ri}(\operatorname{dom} g) \cap \operatorname{ri}(\operatorname{dom} h) \neq \emptyset$ *and $x^* \in \operatorname{dom} g$ such that g is continuous at x^*, then $0 \in \partial f(x^*) \Leftrightarrow \partial h(x^*) \subset \partial g(x^*)$. Thus, in this case, the local optimality is also sufficient for the global optimality. If, in addition, h is differentiable, a critical point is also a global solution.*

Standard DC algorithm

The idea of DCA for solving the problem (P) is that each iteration k of DCA approximates the concave part $-h$ by its affine majorization (that corresponds to taking $y^k \in \partial h(x^k)$) and minimizes the resulting convex function. This algorithm can be summarized as follows.

Algorithm 1.1

```
Initialization. Choose an initial point $x^0 \in X$, set $k := 0$.
Repeat
    1. Compute $y^k \in \partial h(x^k)$.
    2. Compute $x^{k+1} \in \arg\min\{g(x) - h(x^k) - \langle x - x^k, y^k \rangle : x \in X\}$.
    3. Set $k := k + 1$.
Until convergence of $\{x^k\}$.
```

The convergence properties of DCA was completely investigated in [81]. The following theorem indicates some important results.

Theorem 1.2. *Let $\{x^k\}$ and $\{y^k\}$ be the sequences generated by Algorithm* 1.1. *Then the following statements hold.*

- **i)** *The sequences $\{g(x^k) - h(x^k)\}$ and $\{h^*(y^k) - g^*(y^k)\}$ are decreasing.*
- **ii)** *If $g(x^{k+1}) - h(x^{k+1}) = g(x^k) - h(x^k)$ then x^k, x^{k+1} are the critical points of $g - h$. In this case, Algorithm* 1.1. *terminates after a finite number of iterations.*
- **iii)** *If $\rho(g) + \rho(h) > 0$ (resp. $\rho(h^*) + \rho(g^*) > 0$), then the sequence $\{\|x^{k+1} - x^k\|^2\}$ (resp. $\{\|y^{k+1} - y^k\|^2\}$) converges.*
- **iv)** *If the optimal value α is finite and the sequences $\{x^k\}$ and $\{y^k\}$ are bounded, then every limit point x^* (resp. y^*) of the sequence $\{x^k\}$ (resp. $\{y^k\}$) is a critical point of $g - h$ (resp. $h^* - g^*$).*
- **v)** *For polyhedral DC programs, the sequences $\{x^k\}$ and $\{y^k\}$ contain finitely many elements and DCA has a finite convergence. Especially, if h is differentiable at x^*, then x^* is a local minimizer of the problem (P).*

Remark 1.4. *a) When h is a polyhedral function, the calculation of the subdifferential $\partial h(x^k)$ is explicit by Proposition* 1.3. *With a fixed choice of subgradients of h, the sequence $\{y^k\}$ has only finitely many different elements. This leads to finite convergence of DCA.*

b) DCA's distinctive feature relies upon the fact that DCA deals with the convex DC components g and h but not with the DC function f itself. Moreover, a DC function f has infinitely many DC decompositions which have crucial implications for the qualities (e.g. convergence speed, robustness, efficiency, globality of computed solutions) of DCA. For a given DC program, the choice of optimal DC decompositions is still open. Of course, this depends strongly on the very specific structure of the problem being considered.

1.1.3 General DC optimization

A general DC program is of the form

$$\begin{aligned} \min_x \quad & f_0(x) \\ \text{s.t} \quad & x \in C, \\ & f_i(x) \leq 0, \; i = 1, ..., m, \end{aligned} \tag{1.6}$$

where C is a nonempty closed convex set in $\mathbb{R}^n$; $f_i : \mathbb{R}^n \to \mathbb{R}(i = 0, 1, ..., m)$ are DC functions.

This class of nonconvex programs is the most general in DC programming and more difficult than standard DC programs because of the nonconvexity of the constraints. Two approaches for the problem (1.6) were proposed in [43]. The first one employs a penalty technique in DC programming to reformulate the problem (1.6) as a standard DC program. The second one linearizes concave parts in DC constraints to build convex inner approximations of the feasible set. Before presenting these two approaches, we recall some definitions.

Let F be the feasible set of (1.6). A point $x^* \in F$ is a Karush-Kuhn-Tucker (KKT) point for the problem (1.6) if there exist nonnegative scalars $\lambda_i, i = 1, ..., m$ such that

$$\begin{cases} 0 \in \partial^{\uparrow} f_0(x^*) + \sum_{i=1}^m \lambda_i \partial^{\uparrow} f_i(x^*) + N_C(x^*), \\ \lambda_i f_i(x^*) = 0, \quad i = 1, \dots, m. \end{cases} \tag{1.7}$$

Denote

$$p(x) = \max\{f_1(x), f_2(x), ..., f_m(x)\},$$
$$I(x) = \{i \in \{1, ..., m\} : f_i(x) = p(x)\}; p^+(x) = \max\{p(x), 0\}.$$

One says that the *extended Mangasarian-Fromowitz constraint qualification (EMFCQ)* is satisfied at $x^* \in F$ with $I(x^*) \neq \emptyset$ if there is a vector $d \in \text{cone}(C - \{x^*\})$ (the cone hull of $C - \{x^*\}$) such that

$$f_i^{\uparrow}(x^*, d) < 0 \quad \forall i \in I(x^*).$$

When $f_i's$ are continuously differentiable, then $f_i^{\uparrow}(x^*, d) = \langle \nabla f_i(x^*), d \rangle$. Therefore, the EMFCQ becomes the well-known Mangasarian-Fromowitz constraint qualification.

General DCA using l_∞-penalty function with updated penalty parameter

Consider the following penalized problems

$$\begin{aligned} \min_x \quad & \phi_k(x) = f_0(x) + \beta_k p^+(x) \\ \text{s.t} \quad & x \in C, \end{aligned} \tag{1.8}$$

where β_k are penalty parameters. Since $f_i(x), i = 1, \dots, m$ are DC functions, so is p^+. Suppose that f_0 and p^+ are decomposed into the difference of two convex functions as follows

$$f_0(x) = g_0(x) - h_0(x), p^+(x) = p_1(x) - p_2(x)$$

where g_0, h_0, p_1, p_2 are convex functions defined on the whole space. Then a DC decomposition of ϕ_k can be chosen to be

$$\phi_k(x) = g_k(x) - h_k(x)$$

where

$$g_k(x) = g_0(x) + \beta_k p_1(x), \quad h_k(x) = h_0(x) + \beta_k p_2(x).$$

DCA with updated penalty parameter is described in the following algorithm.

Algorithm 1.2

Initialization: Take an initial point $x^1 \in C, \delta > 0$, an inital penalty parameter $\beta_1 > 0$ and set $k := 1$.

1. Compute $y^k \in \partial h_k(x^k)$.

2. Compute x^{k+1} by solving the convex program
$$\min\left\{g_k(x) - \langle x, y^k\rangle : x \in C\right\}.$$
3. Stopping test.
   Stop if $x^{k+1} = x^k$ and $p(x^k) \leq 0$.
4. Penalty parameter update.
   Compute $r_k = \min\left\{p(x^k), p(x^{k+1})\right\}$ and set
$$\beta_{k+1} = \begin{cases} \beta_k & \text{if either } \beta_k \geq \|x^{k+1} - x^k\|^{-1} \text{ or } r_k \leq 0, \\ \beta_k + \delta & \text{if } \beta_k < \|x^{k+1} - x^k\|^{-1} \text{ and } r_k > 0 \end{cases}$$
5. Set $k := k + 1$ and go to Step 1.

In the global convergence theorem, the authors use the following assumptions.

Assumption 1.1. *$f_i (i = 0, ..., m)$ are locally Lipschitz functions at every point of C.*

Assumption 1.2. *Either g_k or h_k is differentiable on C, and $\rho(g_0) + \rho(h_0) + \rho(p_1) + \rho(p_2) > 0$.*

Assumption 1.3. *The EMFCQ is satisfied at any $x \in \mathbb{R}^n$ with $p(x) \geq 0$.*

Theorem 1.3. *Suppose that C is a nonempty closed convex set in $\mathbb{R}^n$ and $f_i, i = 0, 1, \ldots, m$ are DC functions on C. Suppose further that Assumptions* 1.1-1.3 *are verified. Let $\delta > 0, \beta_1 > 0$ be given and $\{x^k\}$ be a sequence generated by Algorithm* 1.2. *Then Algorithm* 1.2 *either stops, after finitely many iterations, at a KKT point x^k for the problem* (1.6) *or generates an infinite sequence $\{x^k\}$ of iterates such that $\lim_{k\to\infty} \|x^{k+1} - x^k\| = 0$ and every limit point x^∞ of the sequence $\{x^k\}$ is a KKT point of the problem* (1.6).

This theorem is proved in detail in [43].

General DCA using slack variables with updated parameter

Since $f_i (i = 0, ..., m)$ are DC functions, they can be decomposed into the difference of two convex functions as follows $f_i(x) = g_i(x) - h_i(x),\ x \in \mathbb{R}^n, i = 0, ..., m.$

In this second approach, at each iteration, one solves the following convex subproblem, which is obtained by replacing the concave parts of the DC structure with their affine majorization.

$$\begin{aligned} \min \quad & g_0(x) - \langle y_0^k, x\rangle \\ \text{s.t.} \quad & x \in C, \\ & g_i(x) - h_i(x^k) - \langle y_i^k, x - x^k\rangle \leq 0, i = 1, ..., m, \end{aligned} \tag{1.9}$$

where $x^k \in \mathbb{R}^n$ is the current iterate, $y_i^k \in \partial h_i(x^k)$ for $i = 0, ..., m$. However, the inner convex approximation of the feasible set of the problem (1.6) is quite often poor and can lead to infeasibility of the convex subproblem (1.9). To deal with the infeasibility of subproblems, a relaxation technique was proposed. Instead of (1.9), the authors consider the subproblem

$$\begin{aligned} \min \quad & g_0(x) - \langle y_0^k, x\rangle + t_k s && (1.10)\\ \text{s.t.} \quad & x \in C, \\ & g_i(x) - h_i(x^k) - \langle y_i^k, x - x^k\rangle \le s, i = 1, ..., m, && (1.11)\\ & s \ge 0, \end{aligned}$$

where $t_k > 0$ is a penalty parameter. Clearly, (1.10) is a convex problem that is always feasible. Moreover, the Slater constraint qualification is satisfied for the constraints of (1.10), thus the Karush-Kuhn-Tucker (KKT) optimality condition holds for some solution (x^{k+1}, s^{k+1}) of (1.10). The algorithm is summarized as follows.

Algorithm 1.3

```
Initialization. Choose an initial point $x^1 \in C$; $\delta_1, \delta_2 > 0$, an initial penalty parameter $t_1 > 0$. Set $k := 1$.
  1. Compute $y_i^k \in \partial h_i(x^k),\ i = 0, ..., m$.
  2. Compute $(x^{k+1}, s^{k+1})$ as a solution to the convex problem (1.10) and the Lagrange multipliers $\lambda_i^{k+1}$ associated with the constraints (1.11).
  3. Stopping test
     If $x^{k+1} = x^k$ and $s^{k+1} = 0$, then stop, otherwise go to Step 4.
  4. Penalty parameter update.
     Compute
        $r_k = \min\left(\|x^{k+1} - x^k\|^{-1}, \sum_{i=1}^{m} |\lambda_i^{k+1}| + \delta_1\right)$
     and set
        $t_{k+1} = \begin{cases} t_k & \text{if } t_k \ge r_k, \\ t_k + \delta_2 & \text{if } t_k < r_k. \end{cases}$
  5. Set $k := k + 1$ and go to Step 1.
```

The global convergence of the above algorithm is shown in the theorem below.

Theorem 1.4. *Suppose that C is a nonempty closed convex set in $\mathbb{R}^n$ and $f_i, i = 0, 1, \ldots, m$ are DC functions on C such that Assumptions 1.1 and 1.3 are verified. Suppose further that for each $i = 0, ..., m$ either g_i or h_i is differentiable on C and that*

$$\rho = \rho(g_0) + \rho(h_0) + \min\{\rho(g_i) : i = 1, \ldots, m\} > 0.$$

Let $\delta_1, \delta_2 > 0, t_1 > 0$ be given and $\{x^k\}$ be a sequence generated by Algorithm 1.3. Then Algorithm 1.3 either stops, after finitely many iterations, at a KKT point x^k for the problem (1.6) or generates an infinite sequence $\{x^k\}$ of iterates such that $\lim_{k\to\infty} \|x^{k+1} - x^k\| = 0$ and every limit point x^∞ of the sequence $\{x^k\}$ is a KKT point of the problem (1.6).

The proof of this theorem is presented in [43].

1.2 Penalty Techniques

In nonconvex optimization, we often face problems in which one or several constraints are complex and difficult to handle directly. The penalty approach is used to transform such a problem into a simpler problem or into a sequence of simpler problems by penalizing difficult constraints. We now present some results concerning this approach. The following content is drawn from [7, 48].

Consider the problem

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g_i(x) \leq 0, \ i = 1, ..., m, \\ & h_j(x) = 0, \ j = 1, ..., l, \\ & x \in C, \end{aligned} \tag{1.12}$$

where $f, g_i, h_j : \mathbb{R}^n \to \mathbb{R}$ are continuous functions and C is a nonempty set in $\mathbb{R}^n$.

Let $\mathcal{F}$ be the feasible set of (1.12). We assume throughout this section that $\mathcal{F}$ is nonempty. Consider the following penalized problem

$$\min\{f(x) + tp(x) : x \in C\} \tag{1.13}$$

where t is a positive number and $p : \mathbb{R}^n \to \mathbb{R}$ is a function satisfying the properties below:

i) p is continuous on $\mathbb{R}^n$,

ii) $p(x) \geq 0, \forall x \in C$,

iii) $p(x) = 0, x \in C \Leftrightarrow x \in \mathcal{F}$.

Such a number t is called a *penalty parameter* and p is called a *penalty function.*

We say that the exact penalty holds if there exists a number $t_0 \geq 0$ such that for all $t > t_0$ the problems (1.12), (1.13) are equivalent in the sense that they have the same optimal value and the same optimal solution set.

The relationship between the problems (1.12) and (1.13) is shown in the next theorem.

Theorem 1.5. [7] *Suppose that $f, g_i, h_j : \mathbb{R}^n \to \mathbb{R}, i = 1, ..., m; j = 1, ..., l$ are continuous functions, C is a nonempty set in $\mathbb{R}^n$ and p is a function satisfying the properties* i)-iii). *Furthermore, suppose that for each t there exists an optimal solution x_t to the penalized problem* (1.13). *If $\{x_t\}$ is contained in a compact set of X, then*

1. *$\theta = \lim\limits_{t\to\infty} \theta(t)$, where $\theta, \theta(t)$ are the optimal value of* (1.12) *and* (1.13) *respectively.*
2. *Every limit point of $\{x_t\}$ is an optimal solution to the problem* (1.12) *and $\lim\limits_{t\to\infty} tp(x_t) = 0$.*

Error bounds for concave inequality systems over polyhedral convex sets

Theorem 1.6. [48] *Let C be a nonempty bounded polyhedral convex set in $\mathbb{R}^n$ and let h be a concave function on K defined by*

$$h(x) = \sum_{j=1}^{m} \min\{h_{ij}(x) : i \in J_j\},$$

where $J_1, ..., J_m$ are finite index sets and h_{ij} are differentiable concave functions on K. If $S = \{x \in C : h(x) \leq 0\}$ is nonempty then there exists $\tau > 0$ such that

$$d(x, S) \leq \tau h^+(x) \ \forall x \in C,$$

where $d(x, S) = \inf\{\|x - z\| : z \in S\}, h^+(x) = \max(h(x), 0)$.

Exact penalty via error bounds

Consider the problem

$$\alpha = \min\{f(x) : x \in C, g(x) \leq 0\}, \tag{1.14}$$

and the penalized problem

$$\alpha(t) = \min\{f(x) + tg(x) : x \in C\}, \tag{1.15}$$

where C is a subset of $\mathbb{R}^n$, f is a real-valued function defined on C and $g : \mathbb{R}^n \to \mathbb{R}$ is a nonnegative function on C. Let $\mathcal{P}$ and $\mathcal{P}_t$ be the optimal solution set of the problems (1.14) and (1.15) respectively.

Proposition 1.4. [48] *Suppose that f is a Lipschitz continuous function on C with Lipschitz constant L and that g is a nonnegative finite function on C. If $S = \{x \in C : g(x) \leq 0\}$ is nonempty and there exists $\ell > 0$ such that*

$$d(x, S) \leq \ell g(x) \ \forall x \in C,$$

then one has:
i) $\alpha(t) = \alpha$ and $\mathcal{P} \subset \mathcal{P}_t$ for all $t \geq L\ell$,
ii) $\mathcal{P}_t = \mathcal{P}$ for all $t > L\ell$.

Exact penalty for concave inequalities constraints

Theorem 1.7. [48] *Suppose that C be a nonempty bounded polyhedral convex set in $\mathbb{R}^n$ and that h be a concave function on C defined as in Theorem 1.6. If $S = \{x \in C : h(x) \leq 0\}$ is nonempty and f is a Lipschitz continuous function on C, then there exists $t_0 > 0$ such that for all $t \geq t_0$ the following problems are equivalent.*

$$\min\{f(x) : x \in C, h(x) \leq 0\}, \tag{1.16}$$

$$\min\{f(x) + th^+(x) : x \in C\}. \tag{1.17}$$

Chapter 2

DC Programs with Linear Complementarity Constraints

Abstract: *We address a large class of Mathematical Programs with Linear Complementarity Constraints which minimizes a continuously differentiable DC function on a set defined by linear constraints and linear complementarity constraints, named DC programs with Linear Complementarity Constraints. Using exact penalty techniques, we reformulate the considered problem, via four penalty functions, as standard DC programs. The DCA schemes are then developed to solve the resulting problems. Two particular cases are considered: quadratic problems with linear complementarity constraints and asymmetric eigenvalue complementarity problems. Numerical experiments are performed on several benchmark data illustrate the effectiveness of proposed DCA schemes.*

2.1 Introduction

The Mathematical Program with Linear Complementarity Constraints (MPLCC) is an important subclass of the Mathematical Program with Equilibrium Constraints (MPEC). It consists of minimizing a continuously differentiable function on a set defined by linear constraints and pairs of complementary variables. The problem has a wide range of applications in engineering, economics, and mathematics itself (see, e.g., [10, 27, 28, 34, 39, 79]). Several NP-hard problems such as bilevel programs, the asymmetric eigenvalue complementarity problem, zero-norm optimization problems can be efficiently solved by reformulating them as MPLCCs. However, solving the MPLCC is known to be very difficult, even when the objective function is linear, due to the non-convexity of linear complementarity constraints. The NP-hardness of the linear bilevel programming problem, a special case of the MPLCC, was proved in [5].

Related works

1. The material of this chapter is based on the following work:
Hoai An Le Thi, Thi Minh Tam Nguyen, Tao Pham Dinh. On Solving Difference of Convex Functions Programs with Linear Complementarity Constraints. *Submitted*

To date, many approaches have been proposed to deal with the MPEC in general and the MPLCC in particular. Under appropriate assumptions, an MPEC can be expressed in the form of a Mathematical Program with Complementarity Constraints (MPCC) by replacing the variational inequality constraints with its Karush-Kuhn-Tucker representation. Therefore, MPCCs constitute a crucial subclass of MPECs and most of the literature on MPECs have been concerned with this subclass. Complementarity constraints are considered as the main cause rendering MPCCs difficult to solve. To overcome this matter, various reformulations of MPCCs have been proposed. The popular approaches include relaxation techniques (see, e.g., [12, 37, 38, 59, 91, 94, 98]), smoothing techniques (see, e.g., [15, 22, 25, 56]) and penalization techniques (see, e.g., [29, 30, 58, 67, 92]). The idea of relaxation techniques is to replace the complementarity constraints $\{G(z) \geq 0, H(z) \geq 0, G(z)^\top H(z) = 0\}$ by extended constraints with favorable properties, for example,

- $\{G_i(z) \geq 0, H_i(z) \geq 0, G_i(z)H_i(z) \leq t\}$ [91],
- $\{G_i(z) \geq 0, H_i(z) \geq 0, G(z)^T H(z) \leq t\}$ [91],
- $\{G_i(z) \geq -t, H_i(z) \geq -t, (G_i(z) - t)(H_i(z) - t) \leq 0\}$ [37],
- $\{G_i(z)H_i(z) - t^2 \leq 0, (G_i(z) + t)(H_i(z) + t) - t^2 \geq 0\}$ [59],
- $\{G_i(z) \geq 0, H_i(z) \geq 0, \varphi(G_i(z) - t, H_i(z) - t) \leq 0\}$ [38], where $\varphi : \mathbb{R}^2 \to \mathbb{R}$ is defined by

$$\varphi(a,b) = \begin{cases} ab & \text{if } a + b \geq 0, \\ -(a^2 + b^2)/2 & \text{if } a + b < 0, \end{cases}$$

where $t > 0$ is a relaxation parameter.

Smoothing methods are often based on a reformulation of the complementarity constraints as a system of equations $\phi(G_i(z), H_i(z)) = 0$, where $\phi : \mathbb{R}^2 \to \mathbb{R}$ is a so-called NCP function. Since NCP functions are not differentiable, they are replaced by smooth functions, for example,

- $\phi_t^{\min}(a,b) = a + b - \sqrt{(a-b)^2 + 4t^2}$ [15],
- $\phi_t^{\mathrm{FB}}(a,b) = a + b - \sqrt{a^2 + b^2 + t}$ [22],
- $\phi_t(a,b) = -t \ln\left[\exp(-a/t) + \exp(-b/t)\right]$ [56],

where $t > 0$ is a smoothing parameter. Besides, the smoothing method in [25] replaces the constraints $G_i(z)H_i(z) = 0$ by $\theta_t(G_i(z)) + \theta_t(H_i(z)) \leq 1$, where $\theta_t : \mathbb{R}_+ \to [0,1]$ is a smooth function and satisfies some appropriate conditions.

Most penalty techniques applied to MPCCs transform the considered problem into a problem with simpler constraints or into a sequence of such problems by penalizing the constraint $G(z)^\top H(z) = 0$. For example, Hu and Ralph [29] gave a class of smooth penalty functions including the function $G(z)^\top H(z)$. Mangasarian and Pang [67] (resp. Jara-Moroni et al. [31]) used the penalty function $p_1(y,w) = \sum_{i=1}^m \min(y_i, w_i)$ to penalize the constraint $y^\top w$ in the MPLCC (resp. in the linear program with linear complementarity constrains, abbreviated as LPLCC). In [67], the authors proved that the exact penalty holds in the case the objective function is concave and bounded below on the set defined by the linear constraints. Some works used a penalty function to penalize all constraints of the MPCC (see, e.g. [30, 92]). In [92], the authors proved that the exact penalty holds, this result requires checking the Mangasarian-Fromovitz type conditions at a feasible point, which is not easy. Besides, an exact penalization technique which penalizes all the constraints except the linear complementarity constraints of the form $\{y \geq 0, w \geq 0, y^\top w\}$ can be found in [33, 61].

Several methods for solving nonlinear programs are also adapted for solution of MPCCs, for example, the methods based on sequential quadratic programming (see, e.g., [21, 32, 62, 66, 85]), interior point methods (see, e.g., [53, 64, 66, 87]). These approaches only find a stationary point of MPCCs. The algorithms for finding a globally optimal solution have been mainly suggested for MPLCCs with convex objectives. Most of which are based on the branch-and-bound methods (see, e.g., [4, 6, 36, 63, 106]). A survey of algorithms for solving MPLCCs can be found in [34]. In general, finding a globally optimal solution to MPLCCs is very hard when the number of complementarity constraints is large.

Recently, several methods based on DC programming and DCA have been given to handle some special cases of the MPLCC. Le Thi and Pham Dinh [46] proposed four equivalent optimization formulations of a LPLCC, and transformed them into standard DC programs. Suitable DCA versions were developed to solve the resulting problems. Following the work in [46], Jara-Moroni et al. [31] studied the DC penalty formulations and the corresponding DCA schemes for computing stationary points of the general LPLCC. DCA was also extensively developed for symmetric and asymmetric eigenvalue complementarity problems in [44] and [76] respectively, and for quadratic eigenvalue complementarity problems in [75].

Our contributions

The purpose of this chapter is to investigate new approaches based on DC programming and DCA for solving the DC programs with linear complementarity constraints (DCLCC), a large class of MPLCCs in which the objective function is a continuously differentiable DC function. To the best of our knowledge, this is the first work in the literature which addresses the general DCLCC. Let us consider the DCLCC of the form

$$\begin{aligned} \min \quad & f(x,y) = g(x,y) - h(x,y) \\ \text{s.t.} \quad & Ax + By + a \leq 0, \\ & Nx + My + q = w, \\ & y \geq 0, w \geq 0, y^\top w = 0, \end{aligned} \tag{2.1}$$

where the functions g, $h : \mathbb{R}^{n+m} \to \mathbb{R}$ are convex and continuously differentiable, $a \in \mathbb{R}^p, q \in \mathbb{R}^m, A \in \mathbb{R}^{p\times n}, B \in \mathbb{R}^{p\times m}, N \in \mathbb{R}^{m\times n}, M \in \mathbb{R}^{m\times m}$.

Our main idea is to introduce a penalty function $p(y,w)$ to replace the constraint $y^\top w = 0$ in (2.1) by $p(y,w) \leq 0$, and then penalize this constraint to the objective function. In this way, the DCLCC (2.1) is reformulated as a DC program for which DCA can be investigated. A natural penalty function has been used in previous works is $p_1(y,w) = \sum_{i=1}^m \min(y_i, w_i)$ (see, e.g., [67] for MPLCCs, [49] for bilevel programs and [31] for LPLCCs). However, the relationship between the critical points of the penalized problem and the stationary points of the MPCC has been only considered for LPLCCs. In this chapter, besides p_1, we consider three new penalty functions and prove that exact penalty holds for all the four penalty functions when the linear constraint set is bounded. By the penalty technique we get four DC formulations of (2.1) which are standard DC programs. We develop suitable DCA schemes to handle these four DC programs. Two among them are reformulated again as general DC programs in order that the convex subproblems in DCA is easier to solve. We prove that, under some reasonable assumptions, every limit point of the sequences generated by each DCA version is a stationary point of the DCLCC. Especially, for some penalty functions, the corresponding DCA schemes enjoy interesting convergence properties and local optimality (they converge after a finite number of iterations to a stationary point of the DCLCC, and in most cases, such a stationary point is a local solution).

After designing DCA for the general DCLCC, we show how to develop these DCA schemes for solving the quadratic problem with linear complementarity constraints and the asymmetric eigenvalue complementarity problem. The corresponding DCA versions are simple, they consist of successively solving convex quadratic programs with linear constraints. We test them on several instances of two above problems, 18 out of 42 test problems have a large number of complementarity constraints (from 512 to 3200).

The rest of the chapter is organized as follows. Section 2.2 presents some stationarity concepts for the DCLCC. Section 2.3 discusses the solution methods for the DCLCC. Section 2.4 shows how to apply the proposed methods to solve the quadratic problem with linear complementarity constraints and the asymmetric eigenvalue complementarity problem. The numerical results are reported in Section 2.5 and some conclusions are given in Section 2.6.

2.2 Stationarity concepts

For reader's convenience, we present some stationarity concepts for the DCLCC. These concepts are based on that studied in [66, 90] for MPCCs. Let $\mathcal{F}$ denote the feasible set of the problem (2.1). For each $(x^*, y^*, w^*) \in \mathcal{F}$, we define the index sets:

$$\begin{aligned} &I_y(y^*, w^*) = \{i : 0 = y_i^* < w_i^*\},\ I_w(y^*, w^*) = \{i : y_i^* > w_i^* = 0\},\\ &I_0(y^*, w^*) = \{i : y_i^* = w_i^* = 0\}. \end{aligned}$$

Définition 2.2.1. *A point $z^* = (x^*, y^*, w^*) \in \mathcal{F}$ is said to be:*

a) weakly stationary, iff there exist multipliers $\beta \in \mathbb{R}^p$ and $\nu^y, \nu^w \in \mathbb{R}^m$ such that

$$\begin{aligned} &\nabla_x f(x^*, y^*) + A^\top \beta - N^\top \nu^w = 0,\\ &\nabla_y f(x^*, y^*) + B^\top \beta - M^\top \nu^w - \nu^y = 0,\\ &\beta \geq 0,\ \beta^\top (Ax^* + By^* + a) = 0,\\ &\nu_i^w = 0,\ i \in I_y(y^*, w^*),\\ &\nu_i^y = 0,\ i \in I_w(y^*, w^*), \end{aligned}$$

b) strongly stationary, iff it is weakly stationary and

$$\nu_i^y \geq 0,\ \nu_i^w \geq 0\ \forall i \in I_0(y^*, w^*),$$

c) B-stationary, iff

$$\nabla_x f(x^*, y^*)^\top dx + \nabla_y f(x^*, y^*)^\top dy \geq 0, \ \forall (dx, dy, dw) \in \mathcal{T}(z^*, \mathcal{F}),$$

where $\mathcal{T}(z^*, \mathcal{F})$ is the tangent cone of $\mathcal{F}$ at z^*.

$\square$

Remark 2.1. i) *If (x^*, y^*, w^*) is a strongly stationary point of the problem* (2.1)*, then it is also a B-stationary point of* (2.1) [90].
ii) *If $f(x, y)$ is convex and (x^*, y^*, w^*) is a B-stationary point of (2.1), then (x^*, y^*, w^*) is a local solution to* (2.1) [66].

2.3 Solution methods based on DC programming and DCA

2.3.1 Reformulations of the DCLCC via penalty functions

Let $\mathcal{C}$ be the set defined by linear constraints of (2.1), i.e.

$$\mathcal{C} = \left\{(x, y, w) \in \mathbb{R}^{n+2m} : Ax + By + a \leq 0, Nx + My + q = w, y \geq 0, w \geq 0\right\}.$$

Obviously, $\mathcal{C}$ is a polyhedron in $\mathbb{R}^{n+2m}$. We assume throughout this chapter that $\mathcal{C}$ is nonempty.

Define the function $\psi : \mathbb{R}^2 \to \mathbb{R}$ by $\psi \in \left\{\psi^{\min}, \psi^{\text{FB}}\right\}$, with

$$\begin{aligned} \psi^{\min}(a, b) &= \min(a, b) \text{ (the min function)}, \\ \psi^{\text{FB}}(a, b) &= a + b - \sqrt{a^2 + b^2} \text{ (the Fischer-Burmeister function)}. \end{aligned}$$

It is straightforward to prove that ψ satisfies the following properties:
i) ψ is concave and continuous on $\mathbb{R}^2_+$;
ii) $\psi(a, b) \geq 0, \forall a, b \geq 0$;
iii) $\psi(a, b) = 0 \Leftrightarrow ab = 0, a \geq 0, \ b \geq 0$.
Hence, the constraint $y^\top w = 0$ in (2.1) can be replaced by

$$\text{either } \sum_{i=1}^{m} \psi(y_i, w_i) = 0 \text{ or } \max_{i=1,\ldots,m} \psi(y_i, w_i) = 0.$$

Now let p denote one of the four penalty functions $p_i : \mathbb{R}^{n+m} \to \mathbb{R}$, $i = 1, \ldots, 4$, defined below

$$p_1(y,w) := \sum_{i=1}^{m} \psi^{\min}(y_i,w_i) = \sum_{i=1}^{m} \min(y_i,w_i), \tag{2.2}$$

$$p_2(y,w) := \sum_{i=1}^{m} \psi^{FB}(y_i,w_i) = \sum_{i=1}^{m}(y_i + w_i - \sqrt{(y_i)^2 + (w_i)^2}), \tag{2.3}$$

$$p_3(y,w) := \max_{i=1,\ldots,m} \psi^{\min}(y_i,w_i) = \max_{i=1,\ldots,m} \min(y_i,w_i), \tag{2.4}$$

$$p_4(y,w) := \max_{i=1,\ldots,m} \psi^{FB}(y_i,w_i) = \max_{i=1,\ldots,m} (y_i + w_i - \sqrt{(y_i)^2 + (w_i)^2}). \tag{2.5}$$

Then the feasible set $\mathcal{F}$ of (2.1) can be expressed as

$$\mathcal{F} = \{(x,y,w) \in \mathcal{C} : p(y,w) \leq 0\},$$

and the problem (2.1) is rewritten in the form

$$\min \{f(x,y) : (x,y,w) \in \mathcal{C},\ p(y,w) \leq 0\}. \tag{2.6}$$

Consider now the penalized problem:

$$\min\{f(x,y) + tp(y,w) : (x,y,w) \in \mathcal{C}\}. \tag{2.7}$$

The following theorem shows that when $\mathcal{C}$ is bounded, the exact penalty for the problem (2.6) holds, that is, (2.6) is equivalent to (2.7) with a sufficiently large t.

Theorem 2.1. *Suppose that $\mathcal{C}$ is bounded and the feasible set $\mathcal{F}$ of* (2.1) *is nonempty. Then, there exists a positive number t_0 such that for all $t \geq t_0$, the problems* (2.6) *and* (2.7) *are equivalent in the sense that they have the same optimal value and the same optimal solution set.*

Proof We first prove that there exists $\tau > 0$ such that

$$d((x,y,w),\mathcal{F}) \leq \tau p(y,w),\ \forall (x,y,w) \in \mathcal{C}.$$

According to Theorem 1.6 in Chapter 1, there exists $\tau_1 > 0$ such that

$$d((x,y,w),\mathcal{F}) \leq \tau_1 p_1(y,w),\ \forall (x,y,w) \in \mathcal{C}. \tag{2.8}$$

On the other hand, one has

$$p_1(y,w) \leq \frac{2+\sqrt{2}}{2} p_2(y,w),\ \forall y,w \geq 0. \tag{2.9}$$

Indeed, if $0 < a \leq b$, then

$$a + b - \sqrt{a^2+b^2} = \frac{2ab}{a+b+\sqrt{a^2+b^2}} = \frac{2a}{\dfrac{a}{b} + 1 + \sqrt{\left(\dfrac{a}{b}\right)^2 + 1}} \geq \frac{2a}{2+\sqrt{2}}.$$

Similarly, if $0 < b \leq a$, then

$$a + b - \sqrt{a^2 + b^2} \geq \frac{2b}{2 + \sqrt{2}}.$$

Consequently,

$$a + b - \sqrt{a^2 + b^2} \geq \frac{2}{2 + \sqrt{2}} \min(a, b),\ \forall a, b \geq 0.$$

Therefore, the inequality (2.9) holds.
Moreover, it is easy to see that

$$p_1(y, w) \leq m p_3(y, w),\ p_2(y, w) \leq m p_4(y, w),\ \forall y, w \geq 0. \tag{2.10}$$

It follows from (2.8)-(2.10) that

$$d((x, y, w), \mathcal{F}) \leq \tau p(y, w),\ \forall (x, y, w) \in \mathcal{C} \text{ with } \tau = \frac{2 + \sqrt{2}}{2} m \tau_1.$$

Obviously, f is a Lipschitz continuous function on $\mathcal{C}$ (since f is continuously differentiable and $\mathcal{C}$ is compact) and if $(x, y, w) \in \mathcal{C}$, then $p(y, w) \geq 0$. As a consequence of Proposition 1.4 in Chapter 1, there exists a positive number t_0 such that for all $t \geq t_0$, the problems (2.6) and (2.7) are equivalent. □

A similar result to Theorem 2.1 for the penalty function p_1 was presented in ([66], Theorem 2.4.3) for mathematical programs with affine equilibrium constraints.

In the next two subsections, we will present the DCA schemes for the problem (2.7) corresponding to the penalty functions defined by (2.2)-(2.5).

2.3.2 Standard DCA schemes for solving the penalized problem when $p \in \{p_1, p_2\}$

When $p(y, w) = p_1(y, w)$, the problem (2.7) is written as the standard DC program:

$$\min\{F_t^1(x, y, w) := G^1(x, y, w) - H_t^1(x, y, w) : (x, y, w) \in \mathbb{R}^{n+2m}\}, \tag{2.11}$$

where

$$G^1(x, y, w) = g(x, y) + \chi_{\mathcal{C}}(x, y, w),\ H_t^1(x, y, w) = h(x, y) - t \sum_{i=1}^{m} \psi^{\min}(y_i, w_i).$$

When $p(y, w) = p_2(y, w)$, the problem (2.7) can also be expressed as the standard DC program:

$$\min\{F_t^2(x, y, w) := G^1(x, y, w) - H_t^2(x, y, w) : (x, y, w) \in \mathbb{R}^{n+2m}\}, \tag{2.12}$$

where

$$H_t^2(x, y, w) = h(x, y) - t \sum_{i=1}^{m} \psi^{\mathrm{FB}}(y_i, w_i).$$

According to the generic DCA scheme for solving standard DC programs, each iteration of DCA applied to (2.11) (resp. (2.12)) consists of first computing a subgradient

$$(\overline{x}^k, \overline{y}^k, \overline{w}^k) \in \partial H_t^1(x^k, y^k, w^k) \text{ (resp. } (\overline{x}^k, \overline{y}^k, \overline{w}^k) \in \partial H_t^2(x^k, y^k, w^k)),$$

and then solving the convex problem

$$\min\{g(x,y) - (\overline{x}^k)^\top x - (\overline{y}^k)^\top y - (\overline{w}^k)^\top w : (x,y,w) \in \mathcal{C}\}. \tag{2.13}$$

Hence, the two corresponding DCAs to solve (2.11) and (2.12) differ from one another only on the first step.
A subgradient $(\overline{x}, \overline{y}, \overline{w}) \in \partial H_t^1(x,y,w)$ can be calculated as follows:

$$\overline{x} = \nabla_x h(x,y),\ \overline{y} = \nabla_y h(x,y) + t\widehat{y},\ \overline{w} = t\widehat{w}, \tag{2.14}$$

with $(\widehat{y}_i, \widehat{w}_i) \in \partial(-\psi^{\min})(y_i, w_i)$.
Since

$$-\psi^{\min}(y_i, w_i) = \max(-y_i, -w_i),$$

a subgradient $(\widehat{y}_i, \widehat{w}_i) \in \partial(-\psi^{\min})(y_i, w_i)$ can be selected to be

$$\widehat{y}_i = \begin{cases} -1 & \text{if } y_i < w_i, \\ 0 & \text{if } y_i \geq w_i, \end{cases} \quad \widehat{w}_i = \begin{cases} 0 & \text{if } y_i < w_i, \\ -1 & \text{if } y_i \geq w_i. \end{cases} \tag{2.15}$$

A subgradient $(\overline{x}, \overline{y}, \overline{w}) \in \partial H_t^2(x,y,w)$ can also be calculated by (2.14) with $(\widehat{y}_i, \widehat{w}_i) \in \partial(-\psi^{\mathrm{FB}})(y_i, w_i)$.
By the definition of the function ψ^{FB}, we have

$$-\psi^{\mathrm{FB}}(y_i, w_i) = \sqrt{(y_i)^2 + (w_i)^2} - y_i - w_i,$$

hence a subgradient $(\widehat{y}_i, \widehat{w}_i) \in \partial(-\psi^{\mathrm{FB}})(y_i, w_i)$ can be chosen as follows:

$$\widehat{y}_i = \begin{cases} \dfrac{y_i}{\sqrt{(y_i)^2 + (w_i)^2}} - 1 & \text{if } (y_i, w_i) \neq (0,0), \\ -1 & \text{if } y_i = w_i = 0, \end{cases} \tag{2.16}$$

$$\widehat{w}_i = \begin{cases} \dfrac{w_i}{\sqrt{(y_i)^2 + (w_i)^2}} - 1 & \text{if } (y_i, w_i) \neq (0,0), \\ -1 & \text{if } y_i = w_i = 0. \end{cases} \tag{2.17}$$

The DCA schemes applied to (2.11) and (2.12) are described in the following algorithms.

Algorithm 2.1 Standard DCA for solving (2.11) (DCA1)

```
Initialization: Choose a point $z^1 = (x^1, y^1, w^1) \in \mathbb{R}^{n+2m}$ and a sufficiently small positive number $\varepsilon$. Set $k := 1$.
    1. Compute $(\overline{x}^k, \overline{y}^k, \overline{w}^k) \in \partial H_t^1(x^k, y^k, w^k)$ using (2.14), (2.15).
    2. Compute $z^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1})$ as an optimal solution to (2.13).
```

```
3. If
   $$\|z^{k+1} - z^k\| \le \varepsilon(\|z^k\| + 1) \text{ or } |F_t^1(z^{k+1}) - F_t^1(z^k)| \le \varepsilon(|F_t^1(z^k)| + 1)$$
   then stop, otherwise set $k := k + 1$ and go to Step 1.
```

Algorithm 2.2 Standard DCA for solving (2.12) (DCA2)

```
Initialization: Choose a point $z^1 = (x^1, y^1, w^1) \in \mathbb{R}^{n+2m}$ and a sufficiently small positive number $\varepsilon$. Set $k := 1$.
1. Compute $(\overline{x}^k, \overline{y}^k, \overline{w}^k) \in \partial H_t^2(x^k, y^k, w^k)$ using (2.14), (2.16), and (2.17).
2. Compute $z^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1})$ as an optimal solution to (2.13).
3. If
   $$\|z^{k+1} - z^k\| \le \varepsilon(\|z^k\| + 1) \text{ or } |F_t^2(z^{k+1}) - F_t^2(z^k)| \le \varepsilon(|F_t^2(z^k)| + 1)$$
   then stop, otherwise set $k := k + 1$ and go to Step 1.
```

Remark 2.2. *The choice of a good penalty parameter is difficult, hence in practice, we update penalty parameters as follows.*

DCA1 with updated penalty parameter

```
Initialization: Choose a point $(x^1, y^1, w^1) \in \mathbb{R}^{n+2m}$, a penalty parameter $t_1 > 0$, a parameter $\delta > 1$, an upper bound $t_{\max}$ and sufficiently small positive numbers $\varepsilon_1, \varepsilon_2$. Set $k := 1$.
1. Compute $(\overline{x}^k, \overline{y}^k, \overline{w}^k) \in \partial H_t^1(x^k, y^k, w^k)$ via (2.14), (2.15).
2. Compute $(x^{k+1}, y^{k+1}, w^{k+1})$ as a solution to the convex program (2.13).
3. Compute $v_k = \max_{i=1,\dots,m} \min(y_i^{k+1}, w_i^{k+1})$. If
   $$\|z^{k+1} - z^k\| \le \varepsilon_1(\|z^k\| + 1) \text{ or } |F_{t_k}^1(z^{k+1}) - F_{t_k}^1(z^k)| \le \varepsilon_1(|F_{t_k}^1(z^k)| + 1)$$
   and $v_k \le \varepsilon_2$, then stop, otherwise go to Step 4.
4. Update penalty parameter
   Set $t_{k+1} = \begin{cases} t_k & \text{if } v_k \le \varepsilon_2, \\ \min(\delta t_k, t_{\max}) & \text{if } v_k > \varepsilon_2. \end{cases}$
5. Set $k := k + 1$ and go to Step 1.
```

The stopping condition in this algorithm ensures that the feasibility error for the complementarity constraints does not exceed ε_2.
The strategy for updating penalty parameters in DCA2 is similar to the one in DCA1.

The following theorem indicates the convergence properties of DCA1 and DCA2.

Theorem 2.2. *The following statements hold:*

i) *DCA1 (resp. DCA2) generates the sequence $\{z^k = (x^k, y^k, w^k)\}$ in $\mathcal{C}$ such that $\{F_t^1(z^k)\}$ (resp. $\{F_t^2(z^k)\}$ is decreasing.*

ii) *If the optimal value of the problem* (2.11) *(resp.* (2.12)*) is finite and the sequence $\{z^k\}$ is bounded, then every limit point $z^* = (x^*, y^*, w^*)$ of $\{z^k\}$ is a critical point of the problem* (2.11) *(resp.* (2.12)*).*

iii) *Furthermore, if* $\min(y^*, w^*) = 0$ *then* z^* *is a weakly stationary point of the DCLCC* (2.1). *If, in addition,* $y_i^* \neq w_i^*$ *for all* $i \in \{1, 2, \dots, m\}$, *then* (x^*, y^*, w^*) *is a strongly stationary point of* (2.1).

Proof i) and ii) are derived from the convergence properties of DCA (see Theorem 1.2 in Chapter 1). The techniques for proving iii) for DCA1 and DCA2 are similar, thus we only present the proof of iii) for DCA1.
Clearly, $(x^*, y^*, w^*) \in \mathcal{C}$. Since $\min(y^*, w^*) = 0, (x^*, y^*, w^*)$ is feasible for the DCLCC (2.1).
According to ii), (x^*, y^*, w^*) is a critical point of (2.11), therefore there exists $(\overline{x}, \overline{y}, \overline{w}) \in \partial H_t^1(x^*, y^*, w^*)$ such that

$$(\overline{x}, \overline{y}, \overline{w}) \in \partial G^1(x^*, y^*, w^*).$$

It follows that

$$(\nabla_x h(x^*, y^*), \nabla_y h(x^*, y^*) + t\widehat{y}, t\widehat{w}) \in (\nabla_x g(x^*, y^*), \nabla_y g(x^*, y^*), 0) + N_{\mathcal{C}}(z^*),$$

where $(\widehat{y}_i, \widehat{w}_i) \in \partial(-\psi^{\min})(y_i^*, w_i^*), N_{\mathcal{C}}(z^*)$ is the normal cone of $\mathcal{C}$ at z^*. Thus

$$0 \in (\nabla_x g(x^*, y^*), \nabla_y g(x^*, y^*), 0) - (\nabla_x h(x^*, y^*), \nabla_y h(x^*, y^*) + t\widehat{y}, t\widehat{w}) + N_{\mathcal{C}}(z^*).$$

This implies that (x^*, y^*, w^*) is an optimal solution to the problem

$$\begin{array}{ll} \min & g(x, y) - \nabla_x h(x^*, y^*)^\top x - (\nabla_y h(x^*, y^*) + t\widehat{y})^\top y - t\widehat{w}^\top w \\ \text{s.t.} & Ax + By + a \leq 0, \\ & Nx + My + q = w, \\ & y \geq 0, w \geq 0. \end{array}$$

Accordingly, there exist multipliers $\beta, \overline{\nu}^y, \overline{\nu}^w$ such that

$$\nabla_x f(x^*, y^*) + A^\top \beta - N^\top (t\widehat{w} + \overline{\nu}^w) = 0, \tag{2.18}$$
$$\nabla_y f(x^*, y^*) + B^\top \beta - M^\top (t\widehat{w} + \overline{\nu}^w) - t\widehat{y} - \overline{\nu}^y = 0, \tag{2.19}$$
$$\beta \geq 0, \ \beta^\top (Ax^* + By^* + a) = 0, \tag{2.20}$$
$$\overline{\nu}_i^y \geq 0, \ \overline{\nu}_i^y y_i^* = 0, \ i = 1, \dots, m, \tag{2.21}$$
$$\overline{\nu}_i^w \geq 0, \ \overline{\nu}_i^w w_i^* = 0, \ i = 1, \dots, m. \tag{2.22}$$

Set $\nu^y = \overline{\nu}^y + t\widehat{y}, \nu^w = \overline{\nu}^w + t\widehat{w}$.
For $i \in I_w(y^*, w^*)$, we have $y_i^* > w_i^* = 0$. Hence, $\widehat{y}_i = 0$ and $\overline{\nu}_i^y = 0$.
Similarly, $\widehat{w}_i = 0$ and $\overline{\nu}_i^w = 0$ for $i \in I_y(y^*, w^*)$.
Therefore, one has

$$\nabla_x f(x^*, y^*) + A^\top \beta - N^\top \nu^w = 0, \tag{2.23}$$
$$\nabla_y f(x^*, y^*) + B^\top \beta - M^\top \nu^w - \nu^y = 0, \tag{2.24}$$
$$\nu_i^y = 0, i \in I_w(y^*, w^*) \tag{2.25}$$
$$\nu_i^w = 0, i \in I_y(y^*, w^*). \tag{2.26}$$

By (2.20) and (2.23)-(2.26), it follows that (x^*, y^*, w^*) is a weakly stationary point of the DCLCC (2.1).
If $y_i^* \neq w_i^*$ for all $i \in \{1, 2, \dots, m\}$, then $I_0(y^*, w^*) = \emptyset$. Consequently, (x^*, y^*, w^*) is also a strongly stationary point of (2.1). $\square$

When f is convex, take $g = f$, $h = 0$, then H_t^1 is convex polyhedral. Thus, (2.11) is a polyhedral DC program. Because of the finite convergence of DCA for polyhedral DC programs and Theorem 2.2, we have the following corollary:

Corollary 2.1. *Suppose that f is a continuously differentiable convex function and the optimal value of the problem* (2.11) *is finite. The following statements hold.*

i) *DCA1 generates the sequence $\left\{z^k = (x^k, y^k, w^k)\right\}$ containing finitely many different elements in $\mathcal{C}$ such that the sequence $\{F_t^1(z^k)\}$ is decreasing. Thus, DCA1 terminates after a finite number of iterations.*

ii) *Assume that DCA1 terminates at $z^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1})$. Then, z^{k+1} is a critical point of* (2.11).

iii) *Moreover, if $\min(y^{k+1}, w^{k+1}) = 0$, then $(x^{k+1}, y^{k+1}, w^{k+1})$ is a weakly stationary point of the DCLCC* (2.1). *If, in addition, $y_i^{k+1} \neq w_i^{k+1}, \forall i \in \{1, \dots, m\}$, then $(x^{k+1}, y^{k+1}, w^{k+1})$ is a local solution to* (2.1).

2.3.3 General DCA schemes for solving the penalized problem when $p \in \{p_3, p_4\}$

When $p(y, w) = p_3(y, w)$, the problem (2.7) can also be expressed as a standard DC program. A DC decomposition of $f(x, y) + tp_3(y, w)$ can be chosen as (see, e.g., [81])

$$f(x, y) + tp_3(y, w) = G_t(x, y, w) - H_t(x, y, w),$$

where

$$G_t(x, y, w) = g(x, y) + t \max_i \{-\sum_{j \neq i} \psi^{\min}(y_i, w_i)\},$$
$$H_t(x, y, w) = h(x, y) - t \sum_{i=1}^{m} \psi^{\min}(y_i, w_i).$$

However, this decomposition leads to hard subproblems because their objective functions contain the maximum of m nonsmooth functions. For overcoming this difficulty, we consider the following equivalent formulation of the problem (2.7)

$$\min\{f(x, y) + ts : (x, y, w) \in \mathcal{C}, \psi^{\min}(y_i, w_i) \leq s, i = 1, \dots, m;\ s \geq 0\}$$

which can be reformulated as the next general DC program:

$$\begin{aligned} \min\{F_t^3(x, y, w, s) := G_t^3(x, y, w, s) - H^3(x, y, w, s) : (x, y, w) \in \mathcal{C}, \\ s \geq 0, -s - [-\psi^{\min}(y_i, w_i)] \leq 0,\ i = 1, \dots, m\}, \end{aligned} \tag{2.27}$$

where $G_t^3(x, y, w, s) = g(x, y) + ts,\ H^3(x, y, w, s) = h(x, y)$.
Adapting the second DCA scheme presented in Chapter 1 for general DC programs,

we propose a DCA for solving the problem (2.27) as follows. At each iteration k, we compute

$$(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\min})(y_i^k, w_i^k),\ i = 1, \ldots, m,$$

and then solve the convex subproblem

$$\begin{aligned}\min\{g(x,y) + ts - \nabla_x h(x^k, y^k)^\top x - \nabla_y h(x^k, y^k)^\top y : (x,y,w) \in \mathcal{C},\ s \geq 0,\\ -s + \psi^{\min}(y_i^k, w_i^k) - \overline{y}_i^k(y_i - y_i^k) - \overline{w}_i^k(w_i - w_i^k) \leq 0,\ i = 1, \ldots, m\}.\end{aligned} \quad (2.28)$$

The general DCA applied to (2.27) can be described as follows.

Algorithm 2.3 General DCA for solving (2.27) (DCA3)

```
Initialization: Choose a point $X^1 = (x^1, y^1, w^1, s^1) \in \mathbb{R}^{n+2m+1}$, and a sufficiently small positive number $\varepsilon$. Set $k := 1$.
    1. Compute $(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\min})(y_i^k, w_i^k),\ i = 1, ..., m$ using (2.15).
    2. Solve the convex problem (2.28) to obtain $X^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$.
    3. If
           $\|X^{k+1} - X^k\| \leq \varepsilon(\|X^k\| + 1)$ or $|F_t^3(X^{k+1}) - F_t^3(X^k)| \leq \varepsilon(|F_t^3(X^k)| + 1)$
       then stop, otherwise set $k := k + 1$ and go to Step 1.
```

Remark 2.3. *In the implementation of DCA3, we update penalty parameters as follows.*

DCA3 with updated penalty parameter

```
Initialization: Choose a point $X^1 = (x^1, y^1, w^1, s^1) \in \mathbb{R}^{n+2m+1}$, an penalty parameter $t_1 > 0$, parameters $\delta > 1, \delta_1 > 0$, an upper bound $t_{\max}$ and sufficiently small positive numbers $\varepsilon_1, \varepsilon_2$. Set $k := 1$.
    1. Compute $(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\min})(y_i^k, w_i^k),\ i = 1, ..., m$ using (2.15).
    2. Compute $X^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$ as an optimal solution to (2.28) and the Lagrange multipliers $\lambda_i^{k+1}$ associated with the last $m$ constrains.
    3. If
           $\|X^{k+1} - X^k\| \leq \varepsilon_1(\|X^k\| + 1)$ or $|F_{t_k}^3(X^{k+1}) - F_{t_k}^3(X^k)| \leq \varepsilon_1(|F_{t_k}^3(X^k)| + 1)$
       and $s^{k+1} \leq \varepsilon_2$ then stop, otherwise go to Step 4.
    4. Update penalty parameter
       Compute
           $r_k = \min\left(\|X^{k+1} - X^k\|^{-1}, \sum_{i=1}^{m} |\lambda_i^{k+1}| + \delta_1\right)$
       and set
           $t_{k+1} = \begin{cases} t_k & \text{if } t_k \geq r_k, \\ \min(\delta t_k, t_{\max}) & \text{if } t_k < r_k. \end{cases}$
    5. Set $k := k + 1$ and go to Step 1.
```

Similarly, when $p(y,w) = p_4(y,w)$, the problem (2.7) can be reformulated as the following general DC program

$$\begin{aligned}\min\{F_t^3(x,y,w,s) = G_t^3(x,y,w,s) - H^3(x,y,w,s) :\\ (x,y,w) \in \mathcal{C}; s \geq 0; -s - [-\psi^{\mathrm{FB}}(y_i,w_i)] \leq 0,\ i = 1,\dots,m\}.\end{aligned} \tag{2.29}$$

Hence, the general DCA applied on (2.29) is summarized in Algorithm 2.4 below.

Algorithm 2.4 General DCA for solving (2.29) (DCA4)

```
Initialization: Choose a point $X^1 = (x^1, y^1, w^1, s^1) \in \mathbb{R}^{n+2m+1}$, and a sufficiently small positive number $\varepsilon$. Set $k := 1$.
1. Calculate $(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\mathrm{FB}})(y_i^k, w_i^k),\ i = 1, ..., m$ using (2.16) and (2.17).
2. Calculate $X^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$, an optimal solution to the next convex problem
       $\min\{g(x,y) + ts - \nabla_x h(x^k,y^k)^\top x - \nabla_y h(x^k,y^k)^\top y : (x,y,w) \in \mathcal{C}, s \geq 0,$
           $-s + \psi^{\mathrm{FB}}(y_i^k,w_i^k) - \overline{y}_i^k(y_i - y_i^k) - \overline{w}_i^k(w_i - w_i^k) \leq 0, i = 1,\dots,m\}.$
3. If
       $\|X^{k+1} - X^k\| \leq \varepsilon(\|X^k\| + 1)$ or $|F_t^3(X^{k+1}) - F_t^3(X^k)| \leq \varepsilon(|F_t^3(X^k)| + 1)$
   then stop, otherwise set $k := k + 1$ and go to Step 1.
```

Remark 2.4. *In practice, we also update penalty parameters in DCA4 by a similar way in DCA3.*

The convergence properties of DCA3 and DCA4 are shown in the following theorem.

Theorem 2.3. *Suppose that either g or h are strongly convex and the optimal value of the problem* (2.27) *(resp.* (2.29)*) is finite. Let $\{X^k = (x^k, y^k, w^k, s^k)\}$ be the sequence generated by DCA3 (resp. DCA4). The following statements hold:*

i) *$\{F_t^3(X^k)\}$ is decreasing.*

ii) *If the sequence $\{X^k\}$ is bounded and $X^* = (x^*, y^*, w^*, s^*)$ is a limit point of $\{X^k\}$ satisfying $s^* = 0$, then (x^*, y^*, w^*) is a weakly stationary point of the DCLCC* (2.1)*. Moreover, if $y_i^* \neq w_i^*$ for all $i \in \{1, 2, \dots, m\}$, then (x^*, y^*, w^*) is a strongly stationary point of* (2.1)*.*

Since the techniques for proving Theorem 2.3 for DCA3 and DCA4 are similar, we only present the proof of this theorem for DCA3. We first prove the following lemma based on the proof of Lemma 1 in [43].

Lemma 2.1. *Let $\varphi(x,y,w) = f(x,y) + tp_3(y,w)$. Then, DCA3 generates the sequence $\{X^k = (x^k, y^k, w^k, s^k)\}$ satisfying*

$$\varphi(x^k,y^k,w^k) - \varphi(x^{k+1},y^{k+1},w^{k+1}) \geq \frac{\rho(g)+\rho(h)}{2}\left(\|x^k - x^{k+1}\|^2 + \|y^k - y^{k+1}\|^2\right),$$

for all $k = 2, 3, \ldots$, *where*

$$\rho(g) = \sup \left\{ \rho \geq 0 \ : \ g - \frac{\rho}{2}\| \cdot \|^2 \textit{ is convex on } \mathbb{R}^{n+m} \right\}.$$

Proof Since $X^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$ is a solution to the problem (2.28) and the constraints of this problem satisfy the Slater constraint qualification, for each k, there exist multipliers $\lambda_i^k \in \mathbb{R}$ $(i = 1, \ldots, m)$ and $\mu^k \in \mathbb{R}$ such that

$$\begin{aligned}
&0 \in (\nabla_x g(x^{k+1}, y^{k+1}), \nabla_y g(x^{k+1}, y^{k+1}), 0) - (\nabla_x h(x^k, y^k), \nabla_y h(x^k, y^k), 0) \\
&\quad + N_{\mathcal{C}}(x^{k+1}, y^{k+1}, w^{k+1}) - (0, \lambda^k \circ \overline{y}^k, \lambda^k \circ \overline{w}^k), && (2.30) \\
&t = \sum_{i=1}^m \lambda_i^k + \mu^k, && (2.31) \\
&(x^{k+1}, y^{k+1}, w^{k+1}) \in \mathcal{C}, && (2.32) \\
&\lambda_i^k \geq 0, && (2.33) \\
&- s^{k+1} + \psi^{\min}(y_i^k, w_i^k) - \overline{y}_i^k(y_i^{k+1} - y_i^k) - \overline{w}_i^k(w_i^{k+1} - w_i^k) \leq 0, \ \forall i, && (2.34) \\
&\lambda_i^k[\psi^{\min}(y_i^k, w_i^k) - \overline{y}_i^k(y_i^{k+1} - y_i^k) - \overline{w}_i^k(w_i^{k+1} - w_i^k) - s^{k+1}] = 0, \ \forall i, && (2.35) \\
&s^{k+1} \geq 0, \ \mu^k \geq 0, \ s^{k+1}\mu^k = 0. && (2.36)
\end{aligned}$$

Here, $x \circ y$ stands for the Hadamard product between x and y, $(x \circ y)_i = x_i y_i$.
By (2.30) and (2.32), one has

$$\begin{aligned}
&\left[\nabla_x g(x^{k+1}, y^{k+1}) - \nabla_x h(x^k, y^k)\right]^\top (x^k - x^{k+1}) \\
&+ \left[\nabla_y g(x^{k+1}, y^{k+1}) - \nabla_y h(x^k, y^k)\right]^\top (y^k - y^{k+1}) \\
&- \sum_{i=1}^m \lambda_i^k \overline{y}_i^k (y_i^k - y_i^{k+1}) - \sum_{i=1}^m \lambda_i^k \overline{w}_i^k (w_i^k - w_i^{k+1}) \geq 0, \ k = 2, 3, \ldots. && (2.37)
\end{aligned}$$

By the convexity and the differentiability of the functions g and h, one has

$$\begin{aligned}
g(x^k, y^k) &\geq g(x^{k+1}, y^{k+1}) + \nabla_x g(x^{k+1}, y^{k+1})^\top (x^k - x^{k+1}) \\
&\quad + \nabla_y g(x^{k+1}, y^{k+1})^\top (y^k - y^{k+1}) + \frac{\rho(g)}{2}\left(\|x^k - x^{k+1}\|^2 + \|y^k - y^{k+1}\|^2\right), && (2.38) \\
h(x^{k+1}, y^{k+1}) &\geq h(x^k, y^k) + \nabla_x h(x^k, y^k)^\top (x^{k+1} - x^k) + \nabla_y h(x^k, y^k)^\top (y^{k+1} - y^k) \\
&\quad + \frac{\rho(h)}{2}\left(\|x^k - x^{k+1}\|^2 + \|y^k - y^{k+1}\|^2\right). && (2.39)
\end{aligned}$$

From (2.37)-(2.39) it follows that

$$\begin{aligned}
f(x^k, y^k) - f(x^{k+1}, y^{k+1}) &\geq \sum_{i=1}^m \lambda_i^k \overline{y}_i^k (y_i^k - y_i^{k+1}) + \sum_{i=1}^m \lambda_i^k \overline{w}_i^k (w_i^k - w_i^{k+1}) \\
&+ \ \frac{\rho(g) + \rho(h)}{2}\left(\|x^k - x^{k+1}\|^2 + \|y^k - y^{k+1}\|^2\right), k = 2, 3, \ldots. && (2.40)
\end{aligned}$$

Since $(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\min})(y_i^k, w_i^k)$, we have

$$-\psi^{\min}(y_i^{k+1}, w_i^{k+1}) \geq -\psi^{\min}(y_i^k, w_i^k) + \overline{y}_i^k(y_i^{k+1} - y_i^k) + \overline{w}_i^k(w_i^{k+1} - w_i^k).$$

This inequality together with (2.34) implies

$$\psi^{\min}(y_i^{k+1}, w_i^{k+1}) \le s^{k+1}, i = 1, \ldots, m \text{ or } p_3(y^{k+1}, w^{k+1}) \le s^{k+1}.$$

By (2.31), (2.35) and (2.36), for $k = 2, 3, \ldots$, one has

$$\begin{aligned}
&\sum_{i=1}^m \lambda_i^k \overline{y}_i^k (y_i^k - y_i^{k+1}) + \sum_{i=1}^m \lambda_i^k \overline{w}_i^k (w_i^k - w_i^{k+1}) \\
&= -\sum_{i=1}^m \lambda_i^k \psi^{\min}(y_i{}^k, w_i^k) + \sum_{i=1}^m \lambda_i^k s^{k+1} = -\sum_{i=1}^m \lambda_i^k \psi^{\min}(y_i^k, w_i^k) + (t - \mu^k) s^{k+1} \\
&\ge -(t - \mu^k) p_3(y^k, w^k) + t s^{k+1} \ge -t p_3(y^k, w^k) + t p_3(y^{k+1}, w^{k+1}).
\end{aligned}$$

Therefore, by (2.40), we have

$$\varphi(x^k, y^k, w^k) - \varphi(x^{k+1}, y^{k+1}, w^{k+1}) \ge \frac{\rho(g) + \rho(h)}{2} \left(\|x^k - x^{k+1}\|^2 + \|y^k - y^{k+1}\|^2 \right),$$

for all $k = 2, 3, \ldots$. □

We now present the proof of Theorem 2.3 for DCA3.
Proof i) can be proved similarly to the proof in [60]. We only need to prove ii).
Since X^* is a limit point of $\{X^k\}$, there exists a subsequence $\left\{X^{k_j+1}\right\}$ of $\{X^k\}$ such that $\lim_{j\to\infty} X^{k_j+1} = X^*$.
Because of Lemma 2.1, the sequence $\{\varphi(x^k, y^k, w^k)\}$ is decreasing. Furthermore, this sequence is bounded below by the optimal value of (2.27), therefore, it converges. This implies that

$$\lim_{k\to\infty} \|x^k - x^{k+1}\| = 0, \ \lim_{k\to\infty} \|y^k - y^{k+1}\| = 0,$$

and consequently

$$\begin{aligned}
&\lim_{j\to\infty} x^{k_j} = \lim_{j\to\infty} x^{k_j+1} = x^*, \lim_{j\to\infty} y^{k_j} = \lim_{j\to\infty} y^{k_j+1} = y^*, \\
&\lim_{j\to\infty} w^{k_j} = \lim_{j\to\infty} w^{k_j+1} = N x^* + M y^* + q = w^*.
\end{aligned}$$

By (2.31), (2.33), and (2.36), the sequences $\left\{\lambda^k\right\}$ and $\left\{\mu^k\right\}$ are bounded. Moreover the sequence $\{(\overline{y}^k, \overline{w}^k)\}$ is bounded. Without loss of generality, assume that

$$\lim_{j\to\infty} \lambda^{k_j} = \lambda^*, \lim_{j\to\infty} \mu^{k_j} = \mu^*, \lim_{j\to\infty} \overline{y}^{k_j} = \overline{y}^*, \lim_{j\to\infty} \overline{w}^{k_j} = \overline{w}^*.$$

By (2.30), (2.32), (2.34), and let $j \to \infty$ one has

$$\begin{aligned}
0 \in\ & (\nabla_x g(x^*, y^*), \nabla_y g(x^*, y^*), 0) \\
& - (\nabla_x h(x^*, y^*), \nabla_y h(x^*, y^*) + \lambda^* \circ \overline{y}^*, \lambda^* \circ \overline{w}^*) + N_{\mathcal{C}}(x^*, y^*, w^*), && (2.41) \\
& (x^*, y^*, w^*) \in \mathcal{C}, \ \psi^{\min}(y_i^*, w_i^*) \le s^*. && (2.42)
\end{aligned}$$

It follows from (2.42) and $s^* = 0$ that (x^*, y^*, w^*) is a feasible point of the DCLCC (2.1).
Since the function $-\psi^{\min}$ is continuous and $(\overline{y}_i^{k_j}, \overline{w}_i^{k_j}) \in \partial(-\psi^{\min})(y_i^{k_j}, w_i^{k_j})$, one gets

$$(\overline{y}_i^*, \overline{w}_i^*) \in \partial(-\psi^{\min})(y_i^*, w_i^*), i = 1, \ldots, m. \qquad (2.43)$$

From (2.41) and (2.43), using the similar reasoning as in the proof of Theorem 2.2, (x^*, y^*, w^*) is a weakly stationary point of (2.1).
If $y_i^* \neq w_i^*$ for all $i \in \{1, 2, \dots, m\}$, then $I_0(y^*, w^*) = \emptyset$. Consequently, (x^*, y^*, w^*) is also a strongly stationary point of (2.1). □

When f is convex, the finite convergence of DCA3 is indicated in the following theorem.

Theorem 2.4. *Suppose that f is a continuously differentiable convex function and the optimal value of the problem* (2.27) *is finite. The following statements hold:*

i) *DCA3 generates the sequence $\left\{X^k = (x^k, y^k, w^k, s^k)\right\}$ containing finitely many different elements such that the sequence $\{F_t^3(X^k)\}$ is decreasing. Thus, DCA3 stops after a finite number of iterations.*

ii) *Assume that DCA3 terminates at $X^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$ satisfying $s^{k+1} = 0$. Then, $(x^{k+1}, y^{k+1}, w^{k+1})$ is a weakly stationary point of the DCLCC* (2.1). *Furthermore, if $y_i^{k+1} \neq w_i^{k+1}$ for all $i \in \{1, \dots, m\}$, then $(x^{k+1}, y^{k+1}, w^{k+1})$ is a local solution to* (2.1).

Proof i) Since f is convex, take $g = f$, $h = 0$, the subproblem (2.28) becomes:

$$\begin{aligned} \min \quad & \{f(x,y) + ts : (x,y,w) \in \mathcal{C},\ s \geq 0, \\ & -s + \psi^{\min}(y_i^k, w_i^k) - \overline{y}_i^k (y_i - y_i^k) - \overline{w}_i^k (w_i - w_i^k) \leq 0,\ i = 1, \dots, m\}. \end{aligned} \tag{2.44}$$

Set $I_k = \{i \in \{1, \dots, m\} : y_i^k < w_i^k\}$ and note that,

$$\psi^{\min}(y_i^k, w_i^k) + \overline{y}_i^k y_i^k + \overline{w}_i^k w_i^k = 0,$$

the problem (2.44) can be rewritten as

$$\min\{f(x,y) + ts : (x,y,w) \in \mathcal{C}; y_i \leq s, i \in I_k;\ w_i \leq s, i \notin I_k; s \geq 0\}. \tag{2.45}$$

Clearly, $\{F_t^3(X^k)\}$ is decreasing because X^{k+1} is an optimal solution to (2.44) while X^k is a feasible point of (2.44).
For each I_k, X^{k+1} is a solution to (2.45). Since the number of subsets I_k of $\{1, \dots, m\}$ is finite, the sequence $\{X^k\}$ has only finitely many different elements. This leads to $F_t^3(X^k) = F_t^3(X^{k+1})$ after a finite number of iterations, that is, i) is proved.
ii) Assume that DCA3 terminates at $X^{k+1} = (x^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$ and $s^{k+1} = 0$. Since X^{k+1} is a solution to (2.44), there exist multipliers $\beta, \overline{\nu}^y, \overline{\nu}^w, \lambda_i^k$, μ^k such that X^{k+1} and these multipliers satisfy the KKT conditions including the following:

$$\nabla_x f(x^{k+1}, y^{k+1}) + A^\top \beta - N^\top (\lambda^k \circ \overline{w}^k + \overline{\nu}^w) = 0, \tag{2.46}$$
$$\nabla_y f(x^{k+1}, y^{k+1}) + B^\top \beta - M^\top (\lambda^k \circ \overline{w}^k + \overline{\nu}^w) - \lambda^k \circ \overline{y}^k - \overline{\nu}^y = 0, \tag{2.47}$$
$$(x^{k+1}, y^{k+1}, w^{k+1}) \in \mathcal{C}, \tag{2.48}$$
$$\beta \geq 0,\ \beta^\top (Ax^{k+1} + By^{k+1} + a) = 0, \tag{2.49}$$
$$\overline{\nu}_i^y \geq 0,\ \overline{\nu}_i^y y_i^{k+1} = 0,\ i = 1, \dots, m, \tag{2.50}$$
$$\overline{\nu}_i^w \geq 0,\ \overline{\nu}_i^w w_i^{k+1} = 0,\ i = 1, \dots, m, \tag{2.51}$$
$$\psi^{\min}(y_i^k, w_i^k) - \overline{y}_i^k (y_i^{k+1} - y_i^k) - \overline{w}_i^k (w_i^{k+1} - w_i^k) \leq s^{k+1},\ i = 1, \dots, m. \tag{2.52}$$

By (2.52), one has

$$\psi^{\min}(y_i^{k+1}, w_i^{k+1}) \le s^{k+1}, \forall i \text{ or } \min(y_i^{k+1}, w_i^{k+1}) \le s^{k+1},\ i = 1, \dots, m,$$

and hence $(x^{k+1}, y^{k+1}, w^{k+1})$ is feasible for (2.1).
Set $\nu^y = \overline{\nu}^y + \lambda^k \circ \overline{y}^k, \quad \nu^w = \overline{\nu}^w + \lambda^k \circ \overline{w}^k$, by (2.46) and (2.47) one has

$$\nabla_x f(x^{k+1}, y^{k+1}) + A^\top \beta - N^\top \nu^w = 0, \tag{2.53}$$

$$\nabla_y f(x^{k+1}, y^{k+1}) + B^\top \beta - M^\top \nu^w - \nu^y = 0, \tag{2.54}$$

X^{k+1} is a solution to the problem (2.44), therefore it is also a solution to (2.45). It follows that $y_i^{k+1} = 0$ for $i \in I_k, w_i^{k+1} = 0$ for $i \notin I_k$.
For $i \in I_w(y^{k+1}, w^{k+1})$, we have $y_i^{k+1} > w_i^{k+1} = 0$, thus $i \notin I_k$, that is, $y_i^k \ge w_i^k$ which leads to $\overline{y}_i^k = 0$. Consequently,

$$\nu_i^y = \overline{\nu}_i^y = 0, i \in I_w(y^{k+1}, w^{k+1}). \tag{2.55}$$

Similarly,

$$\nu_i^w = \overline{\nu}_i^w = 0, i \in I_y(y^{k+1}, w^{k+1}). \tag{2.56}$$

By (2.49) and (2.53)-(2.56), $(x^{k+1}, y^{k+1}, w^{k+1})$ is a weakly stationary point of the DCLCC (2.1).
If $y_i^{k+1} \ne w_i^{k+1}$ for all $i \in \{1, \dots, m\}$ then $(x^{k+1}, y^{k+1}, w^{k+1})$ is a strongly stationary point, hence it is also a B-stationary point. Since the objective function of (2.1) is convex, $(x^{k+1}, y^{k+1}, w^{k+1})$ is a local minimum of this problem (according to Remark 2.1). $\square$

2.3.4 Performance analysis on DCA based algorithms

The standard DC formulations of the problem (2.7) are similar, they have the same first DC component while their second DC component are distinguished by $\psi^{\min}$ or ψ^{FB}. The general DC formulations of (2.7) are also similar, they have the same DC decomposition of objective function and the same first DC component in the DC constraints. Therefore, in terms of running time, DCA1 (resp. DCA3) should be faster than DCA2 (resp. DCA4). The reason is that computing a subgradient of $-\psi^{\min}$ requires less time than that of $-\psi^{\text{FB}}$. Furthermore, DCA1 (resp. DCA2) is expected to be faster than DCA3 (resp. DCA4). It is because each subproblem in the general DCA schemes contains more $m+1$ constraints than each subproblem in the standard DCA schemes, while their objective functions have the same form (being the sum of $g(x, y)$ and a linear function). These observations are confirmed by computational results in our experiments.

2.4 Applications

In this section, we show how to apply the proposed algorithms to solve the Quadratic Problem with Linear Complementarity Constraints (QPLCC) and the asymmetric Eigenvalue Complementarity Problem (EiCP).

2.4.1 Quadratic problems with linear complementarity constraints

We address a special case of the DCLCC (2.1) where f is a quadratic function of the form

$$f(x,y)=\frac{1}{2}\begin{bmatrix}x\\y\end{bmatrix}^\top P\begin{bmatrix}x\\y\end{bmatrix}+[c^\top\ d^\top]\begin{bmatrix}x\\y\end{bmatrix}$$

Here, the matrix $P\in\mathbb{R}^{(n+m)\times(n+m)}$ is symmetric, $c\in\mathbb{R}^n, d\in\mathbb{R}^m$. Let us consider the DC decomposition $f(x,y)=g(x,y)-h(x,y)$, where

$$g(x,y)=\frac{1}{2}\begin{bmatrix}x\\y\end{bmatrix}^\top (P+\rho I)\begin{bmatrix}x\\y\end{bmatrix}+[c^\top\ d^\top]\begin{bmatrix}x\\y\end{bmatrix},\ h(x,y)=\frac{1}{2}\rho(\|x\|^2+\|y\|^2),$$

with $\rho=0$ if P is positive semidefinite, $-\lambda_{\min}(P)+0.001$ otherwise ($\lambda_{\min}(P)$ denotes the smallest eigenvalue of P). The DC formulations of the penalized problem (2.7) corresponding to the four penalty functions are summarized in Table 2.1.

Table 2.1 – DC formulations for the QPLCC

Penalty function	DC formulations function
$p_1(y,w)$	$\min\{F_t^1(x,y,w)=G^1(x,y,w)-H_t^1(x,y,w):\ (x,y,w)\in\mathbb{R}^{n+2m}\}$, $G^1(x,y,w)=\frac{1}{2}\begin{bmatrix}x\\y\end{bmatrix}^\top (P+\rho I)\begin{bmatrix}x\\y\end{bmatrix}+[c^\top\ d^\top]\begin{bmatrix}x\\y\end{bmatrix}+\chi_{\mathcal{C}}(x,y,w)$, $H_t^1(x,y,w)=\frac{1}{2}\rho(\|x\|^2+\|y\|^2)-t\sum_{i=1}^m\psi^{\min}(y_i,w_i)$.
$p_2(y,w)$	$\min\{F_t^2(x,y,w)=G^1(x,y,w)-H_t^2(x,y,w):\ (x,y,w)\in\mathbb{R}^{n+2m}\}$, $H_t^2(x,y,w)=\frac{1}{2}\rho(\|x\|^2+\|y\|^2)-t\sum_{i=1}^m\psi^{\mathrm{FB}}(y_i,w_i)$.
$p_3(y,w)$	$\min\ \{F_t^3(x,y,w,s)=G_t^3(x,y,w,s)-H^3(x,y,w,s):$ $(x,y,w)\in\mathcal{C};\ -s-[-\psi^{\min}(y_i,w_i)]\le 0,\ i=1,\dots,m;\ s\ge 0\}$, $G_t^3(x,y,w,s)=\frac{1}{2}\begin{bmatrix}x\\y\end{bmatrix}^\top (P+\rho I)\begin{bmatrix}x\\y\end{bmatrix}+[c^\top\ d^\top]\begin{bmatrix}x\\y\end{bmatrix}+ts$, $H^3(x,y,w,s)=\frac{1}{2}\rho(\|x\|^2+\|y\|^2)$.
$p_4(y,w)$	$\min\ \{F_t^3(x,y,w,s)=G_t^3(x,y,w,s)-H^3(x,y,w,s):$ $(x,y,w)\in\mathcal{C};\ -s-[-\psi^{\mathrm{FB}}(y_i,w_i)]\le 0,\ i=1,\dots,m;\ s\ge 0\}$.

Then, the two steps in each iteration k of corresponding DCA schemes are described as follow.

QP-DCA1.

- Compute $(\overline{x}^k,\overline{y}^k,\overline{w}^k)\in\partial H_t^1(x^k,y^k,w^k)$ by the following formulas:

$$\overline{x}^k=\rho x^k,\ \overline{y}^k=\rho y^k+t\widehat{y^k},\ \overline{w}^k=t\widehat{w^k}, \tag{2.57}$$

where $\widehat{y^k}_i,\widehat{w^k}_i$ are calculated using (2.15).

- Compute $(x^{k+1}, y^{k+1}, w^{k+1})$ as a solution to the quadratic program with linear constraints:

$$\begin{array}{ll} \min & \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^\top (P+\rho I)\begin{bmatrix} x \\ y \end{bmatrix} + [(c-\overline{x}^k)^\top \ (d-\overline{y}^k)^\top]\begin{bmatrix} x \\ y \end{bmatrix} \\ & -(\overline{w}^k)^T w \qquad\qquad (2.58) \\ \text{s.t.} & (x,y,w)\in\mathcal{C}. \end{array}$$

QP-DCA2.

- Compute $(\overline{x}^k, \overline{y}^k, \overline{w}^k) \in \partial H_t^2(x^k, y^k, w^k)$ by (2.57), where $\widehat{y^k}_i, \widehat{w^k}_i$ are calculated using (2.16) and (2.17).
- Compute $(x^{k+1}, y^{k+1}, w^{k+1})$ by solving (2.58).

QP-DCA3.

- Compute $(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\min})(y_i^k, w_i^k),\ i = 1,\dots,m$ using (2.15).
- Compute $(x^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$ by solving the quadratic program with linear constraints:

$$\begin{array}{ll} \min & \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^\top (P+\rho I)\begin{bmatrix} x \\ y \end{bmatrix} + [(c-\rho x^k)^\top \ (d-\rho y^k)^\top]\begin{bmatrix} x \\ y \end{bmatrix} + ts \quad (2.59) \\ \text{s.t.} & (x,y,w)\in\mathcal{C},\ s\geq 0, \\ & \psi^{\min}(y_i^k, w_i^k) - \overline{y}_i^k(y_i - y_i^k) - \overline{w}_i^k(w_i - w_i^k) - s \leq 0,\ i=1,\dots,m. \end{array}$$

QP-DCA4.

- Compute $(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\mathrm{FB}})(y_i^k, w_i^k),\ i = 1,\dots,m$ using (2.16) and (2.17).
- Compute $(x^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$ as a solution to the quadratic program with linear constraints:

$$\begin{array}{ll} \min & \frac{1}{2}\begin{bmatrix} x \\ y \end{bmatrix}^\top (P+\rho I)\begin{bmatrix} x \\ y \end{bmatrix} + [(c-\rho x^k)^\top \ (d-\rho y^k)^\top]\begin{bmatrix} x \\ y \end{bmatrix} + ts \quad (2.60) \\ \text{s.t.} & (x,y,w)\in\mathcal{C},\ s\geq 0 \\ & \psi^{\mathrm{FB}}(y_i^k, w_i^k) - \overline{y}_i^k(y_i - y_i^k) - \overline{w}_i^k(w_i - w_i^k) - s \leq 0,\ i=1,\dots,m. \end{array}$$

2.4.2 Asymmetric eigenvalue complementarity problems

Let A be a square matrix of order n and B be an $n \times n$ symmetric positive definite matrix. The eigenvalue complementarity problem is to find a real number $\lambda > 0$ and a vector $y \in \mathbb{R}^n \setminus \{0\}$ such that

$$w = (\lambda B - A)y,\ w \geq 0,\ y \geq 0,\ y^T w = 0.$$

For any solution (λ, y) to the EiCP, λ is said to be a complementary eigenvalue of (A, B) and y is called a corresponding complementary eigenvector. As mentioned in [35], the class of the matrix A plays a very important role in the solution of the EiCP. When A is symmetric, the EiCP is equivalent to the problem of finding a stationary point of the Rayleigh function on the simplex. However, this is no longer true when A is asymmetric. In this application, we consider the case where A is asymmetric. Let

$$v = 1/\lambda,\ u = vy,$$

the EiCP can be transformed into the following MPLCC [35, 76]

$$\min_{u,v,y,w} \{f(u,v,y) := \|u - vy\|^2 : \; w = By - Au, \; e^T y = 1, \\ l_b \le v \le u_b, u \ge 0; \; y \ge 0, w \ge 0, y^T w = 0\}. \tag{2.61}$$

where e denotes the all-ones vector, $l_b = \dfrac{\lambda_{\min}(B)}{\lambda_{\max}(D)}, D = \dfrac{1}{2}(A + A^\top), u_b$ is the optimal value of the linear program below

$$\max_{u,v,y}\{v : By - Au \ge 0, e^T y = 1, e^T u = v, u \ge 0, y \ge 0, v \ge 0\}.$$

If this problem is unbounded, we can take u_b to be a sufficiently large positive number. Note that, (λ, y) is a solution to the EiCP iff $(y/\lambda, 1/\lambda, y, w)$ is a solution to the MPLCC problem (2.61) with zero objective value.

Let $\rho_1 := \max\{4u_b, 2u_b + 2\}, \rho_2 = \max\{2u_b^2 + 4u_b, 4u_b + 2\}$. Using the same reasoning as in [76], we chose the following DC decomposition of f:

$$f(u,v,y) = g(u,v,y) - h(u,v,y),$$

where

$$\begin{aligned} g(u,v,y) &= \left(\frac{\rho_1}{2} + 1\right)\|u\|^2 + \frac{\rho_1 + \rho_2}{2} v^2 + \frac{\rho_1 + \rho_2}{2}\|y\|^2, \\ h(u,v,y) &= g(u,v,y) - f(u,v,y). \end{aligned}$$

Therefore, the problem (2.61) takes the form of (2.1) in which $x = (u,v)$ and the set of linear constraints is

$$\mathcal{C} = \left\{ (u,v,y,w) \in \mathbb{R}^{3n+1} : \begin{array}{l} w = By - Au, e^T y = 1, l_b \le v \le u_b, \\ u \ge 0, y \ge 0, w \ge 0 \end{array} \right\}.$$

The DC formulations of the penalized problem (2.7) corresponding to the four penalty functions are summarized in Table 2.2.
The two steps in each iteration k of corresponding DCA schemes can be described as follow.

Ei-DCA1.

- Compute $(\overline{u}^k, \overline{v}^k, \overline{y}^k, \overline{w}^k) \in \partial H_t^1(u^k, v^k, y^k, w^k)$ by the formulas below:

$$\overline{u}^k = \nabla_u h(u^k, v^k, y^k), \; \overline{v}^k = \nabla_v h(u^k, v^k, y^k), \tag{2.62}$$

$$\overline{y}^k = \nabla_y h(u^k, v^k, y^k) + t\widehat{y^k}, \; \overline{w}^k = t\widehat{w^k}, \tag{2.63}$$

where

$$\nabla_u h(u,v,y) = \rho_1 u + 2vy, \tag{2.64}$$

$$\nabla_v h(u,v,y) = (\rho_1 + \rho_2)v + 2y^T u - 2v\|y\|^2 \tag{2.65}$$

$$\nabla_y h(u,v,y) = (\rho_1 + \rho_2)y + 2vu - 2v^2 y. \tag{2.66}$$

and $\widehat{y^k}_i, \widehat{w^k}_i$ are computed using (2.15).

Table 2.2 – DC formulations for the EiCP

penalty function	DC formulations
$p_1(y,w)$	$\min\{F_t^1(u,v,y,w) = G^1(u,v,y,w) - H_t^1(u,v,y,w) : \ (u,v,y,w) \in \mathbb{R}^{3n+1}\}$, $G^1(u,v,y,w) = g(u,v,y) + \chi_{\mathcal{C}}(u,v,y,w)$, $H_t^1(u,v,y,w) = g(u,v,y) - f(u,v,y) - t\sum_{i=1}^m \psi^{\min}(y_i,w_i)$.
$p_2(y,w)$	$\min\{F_t^2(u,v,y,w) = G^1(u,v,y,w) - H_t^2(u,v,y,w) : \ (u,v,y,w) \in \mathbb{R}^{3n+1}\}$, $H_t^2(u,v,y,w) = g(u,v,y) - f(u,v,y) - t\sum_{i=1}^m \psi^{\mathrm{FB}}(y_i,w_i)$.
$p_3(y,w)$	$\min\ \{F_t^3(u,v,y,w,s) = G_t^3(u,v,y,w,s) - H^3(u,v,y,w,s) :$ $(u,v,y,w) \in \mathcal{C};\ -s - [-\psi^{\min}(y_i,w_i)] \le 0,\ i=1,\dots,m;\ s \ge 0\}$, $G_t^3(u,v,y,w,s) = g(u,v,y) + ts,\ H^3(u,v,y,w,s) = g(u,v,y) - f(u,v,y)$.
$p_4(y,w)$	$\min\ \{F_t^3(u,v,y,w,s) = G_t^3(u,v,y,w,s) - H^3(u,v,y,w,s) :$ $(u,v,y,w) \in \mathcal{C};\ -s - [-\psi^{\mathrm{FB}}(y_i,w_i)] \le 0,\ i=1,\dots,m;\ s \ge 0\}$,

- Compute $(u^{k+1}, v^{k+1}, y^{k+1}, w^{k+1})$ as the optimal solution to the convex quadratic program:

$$\begin{array}{ll} \min & g(u,v,y) - (\overline{u}^k)^T u - (\overline{v}^k)^T v - (\overline{y}^k)^T y - (\overline{w}^k)^T w \\ \text{s.t.} & (u,v,y,w) \in \mathcal{C}. \end{array} \tag{2.67}$$

Ei-DCA2.
- Compute $(\overline{u}^k, \overline{v}^k, \overline{y}^k, \overline{w}^k) \in \partial H_t^2(u^k, v^k, y^k, w^k)$ by (2.62)-(2.66), where $\widehat{y^k}_i$ and $\widehat{w^k}_i$ are calculated using (2.16) and (2.17).
- Compute $(u^{k+1}, v^{k+1}, y^{k+1}, w^{k+1})$ by solving (2.67).

Ei-DCA3.
- Compute $(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\min})(y_i^k, w_i^k),\ i = 1,\dots,m$ using (2.15).
- Compute $(u^{k+1}, v^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$ as the optimal solution to the convex quadratic program:

$$\begin{array}{ll} \min & g(u,v,y) + ts - \nabla_u h(u^k,v^k,y^k)^T u \\ & -\nabla_v h(u^k,v^k,y^k)^T v - \nabla_y h(u^k,v^k,y^k)^T y \\ \text{s.t.} & (u,v,y,w) \in \mathcal{C},\ s \ge 0 \\ & \psi^{\min}(y_i^k,w_i^k) - \overline{y}_i^k(y_i - y_i^k) - \overline{w}_i^k(w_i - w_i^k) - s \le 0,\ i = 1,\dots,m. \end{array}$$

Ei-DCA4.
- Compute $(\overline{y}_i^k, \overline{w}_i^k) \in \partial(-\psi^{\mathrm{FB}})(y_i^k, w_i^k),\ i = 1,\dots,m$ using (2.16) and (2.17).
- Compute $(u^{k+1}, v^{k+1}, y^{k+1}, w^{k+1}, s^{k+1})$ as the optimal solution to the convex quadratic program:

$$\begin{array}{ll} \min & g(u,v,y) + ts - \nabla_u h(u^k,v^k,y^k)^T u \\ & -\nabla_v h(u^k,v^k,y^k)^T v - \nabla_y h(u^k,v^k,y^k)^T y \\ \text{s.t.} & (u,v,y,w) \in \mathcal{C},\ s \ge 0 \\ & \psi^{\mathrm{FB}}(y_i^k,w_i^k) - \overline{y}_i^k(y_i - y_i^k) - \overline{w}_i^k(w_i - w_i^k) - s \le 0,\ i = 1,\dots,m. \end{array} \tag{2.68}$$

2.5 Numerical experiments

The proposed algorithms are tested on two sets of test problems:

- Quadratic Problems with Linear Complementarity Constraints in MacMPEC [52], a collection of MPEC test problems in AMPL.
- Asymmetric Eigenvalue Complementarity Problems with data taken from Matrix Market [2], a visual repository of test data.

To evaluate the performance of the DCA schemes, we compare them with KNITRO solver [3], an advanced solver for nonlinear optimization problems including MPCCs, via NEO server [4]. For the second set of test problems, we also make a comparison of DCA3 with a DCA based algorithm to solve the EiCP developed in [76] (we call DCA-NLP). The five versions of DCA are coded in Matlab R2013b and run on an HP computer Intel(R) Core(TM) i5-3470 CPU, 3.2GHz, 8 Go of RAM. The software CPLEX 12.6 is used to solve convex quadratic programs. We take $\varepsilon_1 = \varepsilon_2 = 10^{-6}$, $t_{\max} = 10^6$ for all DCA based algorithms.

2.5.1 Numerical results on QPLCCs

We tested the four DCA schemes on a collection of 24 QPLCCs in MacMPEC. These algorithms require choosing an initial penalty parameter t_1 and a parameter δ that can affect the quality of solutions. Finding good theoretical choices of such parameters is difficult and beyond the scope of this chapter. Observing from several numerical experiments on different values of these parameters that these values could not be large, we take $t_1 = 10$ and $\delta = 2$ for the test problems from 1 to 15 and 24. For the other test problems, t_1 and δ are selected in the sets $\{1, 2, 5, 10\}$ and $\{1.2, 1.5, 2, 4, 4.5, 5, 5.5, 6.5, 7, 7.5, 9, 9.5\}$ respectively such that the corresponding value of the objective function is the smallest one. In the experiments, we compare the value of objective function obtained by the DCA schemes and KNITRO as well as the computing time (in seconds) of the DCA schemes. We also report the CPU time of KNITRO, but as it is executed in NEO server - another computing environment, it is not fair to compare with DCAs. The computational results are summarized in Table 2.3. From these results, we observe that:

- In terms of the objective function value: Overall, the general DCA versions (QP-DCA3 and QP-DCA4) are better than KNITRO while the standard DCA schemes (QP-DCA1 and QP-DCA2) are comparable with KNITRO. More precisely, all algorithms yield the same objective values in 4/24 test problems (numbers 1, 2, 14, 15). QP-DCA4 outperforms KNITRO on 11/24 problems and comparable with KNITRO on the problems 9-13 (the objective function values obtained by QP-DCA4 and KNITRO are in the interval [1.4e-5,1.6e-5]

2. `https://math.nist.gov/MatrixMarket/matrices.html`
3. `https://www.artelys.com/en/optimization-tools/knitro`
4. `https://neos-server.org/neos/solvers`

for the problems 9, 10 and in the interval [2.1e-6,6.6e-6] for the problems 11-13). For the eight remaining problems, QP-DCA4 and KNITRO give the same objective values. QP-DCA3 is also superior to KNITRO on 11/24 problems but it furnishes a slightly worse solution than KNITRO in the problems 11, 20 and 22. QP-DCA1 (resp. QP-DCA2) is better than KNITRO in 8/24 (resp. 9/24) problems and worse than KNITRO in 7/24 problems. It is noted that QP-DCA3 and QP-DCA4 find a very good solution to the problem 24 (the value of the objective function is -683.0330 compared to -230.0386 found by KNITRO) whereas QP-DCA1 and QP-DCA2 do not find a feasible solution to this problem after 3600 seconds. Among the proposed algorithms, the general DCA schemes are comparable while for the standard DCA schemes, QP-DCA2 provides better solutions in the test problems with a medium number of complementarity constraints (the problems from 17 to 23).

- In terms of computing time, QP-DCA1 is the most efficient among the DCA based algorithms, it is faster than all the other DCA schemes on 16/24 problems. On average, it is 1.33 times faster than QP-DCA2 and around 2.2 times faster than QP-DCA3 and QP-DCA4. We also notice that QP-DCA1 (resp. QP-DCA3) is faster than QP-DCA2 (resp. QP-DCA4) on 17/24 problems, and QP-DCA1 (resp. QP-DCA2) is faster than QP-DCA3 (resp. QP-DCA4) on 21/24 problems (resp. 18/24 problems). Moreover, we observe that the computing time of DCA versions is quite steady, it ranges between 0.01 seconds and 15.03 seconds whereas the computing time of KNITRO solver via NEOS server ranges from 0.005 seconds to 102.148 seconds.

Table 2.3 – Comparative results of the DCA schemes and KNITRO on QPLCCs. Best results are written in bold.

Problem					Value of Objective Function					computing time (seconds)				
No.	Name	n	m	p	QP-DCA1	QP-DCA2	QP-DCA3	QP-DCA4	KNITRO	QP-DCA1	QP-DCA2	QP-DCA3	QP-DCA4	KNITRO
1	bard1	2	3	1	**17**	**17**	**17**	**17**	**17**	**0.0150**	0.0286	0.0171	0.0512	0.0055
2	bard2	8	4	5	**-6598**	**-6598**	**-6598**	**-6598**	**-6598**	**0.0141**	0.0240	0.0172	0.0308	0.0055
3	bilevel2	8	12	5	**-6600**	-6598	**-6600**	**-6600**	**-6600**	0.0273	**0.0238**	0.0379	0.0493	0.0115
4	flp4-1	50	30	30	**1.2382e-11**	**1.2382e-11**	2.6276e-10	2.6276e-10	8.7744e-7	**0.0215**	0.0261	0.0278	0.0321	0.0407
5	flp4-2	50	60	50	**4.2481e-12**	**4.2481e-12**	2.5837e-10	2.5837e-10	1.3337e-7	0.0424	**0.0416**	0.0451	0.0434	0.0767
6	flp4-3	70	70	100	1.1257e-10	1.1257e-10	**3.9103e-11**	**3.9103e-11**	1.4551e-7	0.0913	**0.0887**	0.0923	0.0907	0.1174
7	flp4-4	100	100	150	**1.2111e-11**	**1.2111e-11**	4.9565e-11	4.9565e-11	1.0818e-7	0.3876	0.2926	**0.2117**	0.2691	0.2012
8	nash1	4	2	2	2.7572e-19	2.7572e-19	**1.8788e-19**	**1.8788e-19**	2.5506e-17	**0.0121**	0.0158	0.0225	0.0153	0.0051
9	portfl-i-1	75	12	13	**1.5025e-5**	1.5028e-5	**1.5025e-5**	1.5030e-5	1.5188e-5	**0.0155**	0.0392	0.0261	0.0433	0.0094
10	portfl-i-2	75	12	13	1.4577e-5	1.4578e-5	**1.4573e-5**	1.4579e-5	1.4574e-5	**0.0149**	0.0243	0.0188	0.0359	0.0099
11	portfl-i-3	75	12	13	1.3739e-4	1.3739e-4	1.3739e-4	**6.2666e-6**	6.5513e-6	**0.0150**	0.0393	0.0188	0.0443	0.0755
12	portfl-i-4	75	12	13	2.1811e-6	2.1801e-6	**2.1791e-6**	2.1813e-6	3.4806e-6	**0.0154**	0.0257	0.0183	0.0507	0.0120
13	portfl-i-6	75	12	13	2.3620e-6	2.3648e-6	**2.3618e-6**	2.3640e-6	6.5325e-6	**0.0158**	0.0232	0.0189	0.0352	0.0078
14	qpec-1	10	20	0	**80**	**80**	**80**	**80**	**80**	0.1032	**0.0249**	0.0342	0.0308	0.0051
15	qpec-2	10	20	0	**45**	**45**	**45**	**45**	**45**	**0.0158**	0.0206	0.0555	0.0934	0.0062
16	qpec-100-1	5	100	2	0.1533	0.2066	**0.0990**	**0.0990**	0.1092	**0.4607**	2.3859	1.9873	2.8956	0.1041
17	qpec-100-2	10	100	2	-5.5510	-6.2910	**-6.5907**	**-6.5907**	-6.2605	3.2278	4.0573	4.3072	**3.2104**	1.5854
18	qpec-100-3	10	100	4	-5.2950	-5.3511	**-5.4806**	**-5.4806**	**-5.4806**	**1.1423**	1.6828	4.2007	3.0418	0.0933
19	qpec-100-4	20	100	4	-3.4068	-3.6042	**-4.0920**	-4.0648	-1.3680	**1.2022**	2.7990	1.5081	2.0983	0.1841
20	qpec-200-1	10	200	4	-1.9090	-1.9336	-1.9346	**-1.9348**	**-1.9348**	4.2381	**2.2344**	7.3003	10.0591	0.7155
21	qpec-200-2	20	200	4	-21.8682	-23.5043	**-24.0410**	**-24.0410**	-21.7179	**3.9500**	5.4249	15.0262	12.4366	102.148
22	qpec-200-3	20	200	8	-1.8059	-1.8521	-1.9483	**-1.9534**	**-1.9534**	**3.2055**	5.1374	4.0850	4.7170	7.7648
23	qpec-200-4	40	200	8	-5.5872	-5.9927	-6.1672	**-6.2014**	-4.9308	**2.5588**	3.1809	5,3053	5.8601	63.5179
24	ralphmod	4	100	0	–	–	**-683.0330**	**-683.0330**	-230.0386	–	–	**4.0380**	4.5280	1.0070
	Average									**0.9040**	1.2018	2.0175	2.0734	7.4046

2.5.2 Numerical results on EiCPs

In this numerical experiments, B is the identity matrix of order n and A is an asymmetric square matrix in Matrix Market. The names of matrix A are listed in the second column of Table 2.4. The 18 test problems in this case have a large number of complementarity constraints (from 512 to 3200). For the proposed DCA schemes, the parameters t_1 and δ are chosen as follows: $t_1 = 10, \delta = 2$.

2.5.2.1 DCA based algorithms and KNITRO solver

As before, we are interested in the value of objective function obtained by the DCA versions and KNITRO as well as the computing time (in seconds) of the DCA schemes. The numerical results are reported in Table 2.4. We observe from these results that:

- In terms of the objective function value, Ei-DCA3 is the best. Ei-DCA3 outperforms KNITRO in all cases and yields a global optimal solution to (2.61) with the accuracy less than 10^{-6} (resp. 10^{-5}) in 15/18 (resp. 17/18) problems. Ei-DCA4 comes after, it is better than KNITRO in 17/18 problems (for which it gives a global optimal solution to the EiCP with the accuracy less than 10^{-5}). Both Ei-DCA1 and Ei-DCA2 outperform KNITRO on 16/18 problems. Ei-DCA1 (resp. Ei-DCA2) provides a global optimal solution to the EiCP with the accuracy less than 10^{-5} in 15/18 (resp. 16/18) problems and fails (i.e. does not find a feasible solution after 3600 seconds) in the problems 4 and 6 (resp. the problem 4). KNITRO yields solutions to the EiCP with the worst accuracy (greater than 10^{-4}) in 7/18 test problems and finds an infeasible solution in the problems 4 and 6. We also notice that the general DCA schemes provide a very good solution in the problem 4 (the values of objective function less than 1.1^{-10}) while the other algorithms fail in this problem. Among standard (resp. general) DCA versions, Ei-DCA1 is better than Ei-DCA2 in 10/18 test problems and only worse than Ei-DCA2 in the problems 6 and 12 (resp. Ei-DCA3 outperforms Ei-DCA4 in 14/18 problems and is comparable to Ei-DCA4 in the remaining problems).
- In terms of computing time, although the test problems have the large dimension, the DCA based algorithms still run fairly fast (less than 12 seconds). On average, Ei-DCA1 is fastest and it is also faster than all the other DCA schemes on 8/18 problems.

Table 2.4 – Comparative results of the DCA schemes and KNITRO on EiCPs. Best results are written in bold.

Problem			Value of Objective Function					computing time (seconds)				
No.	Name	n	Ei-DCA1	Ei-DCA2	Ei-DCA3	Ei-DCA4	KNITRO	Ei-DCA1	Ei-DCA2	Ei-DCA3	Ei-DCA4	KNITRO
1	dwa512	512	5.6656e-6	7.9248e-5	5.5293e-6	**1.7379e-6**	8.3814e-4	**0.8686**	1.1377	1.2514	1.2868	2.2281
2	dwb512	512	**3.0900e-9**	4.3922e-6	3.4041e-9	7.2344e-7	1.1767e-6	0.1046	0.3674	**0.1040**	0.2066	0.4616
3	west0655	655	1.8638e-6	2.6080e-6	**1.4025e-6**	1.6185e-6	9.1148e-4	0.4723	0.7319	**0.3884**	0.7724	18.0686
4	qh768	768	–	–	**2.0198e-11**	1.0484e-10	–	–	–	**0.1549**	0.1909	–
5	rdb800l	800	**2.0102e-8**	4.5556e-6	2.1420e-8	1.8555e-6	4.7724e-6	**0.7571**	1.6451	1.1036	1.5707	0.8154
6	qh882	882	–	2.6568e-7	**3.7900e-8**	1.9447e-7	–	–	0.2081	**0.1894**	0.4158	–
7	rdb968	968	2.0829e-7	**1.9912e-7**	2.0750e-7	4.4420e-6	3.9908e-6	0.2590	**0.0692**	0.2973	1.8652	0.5274
8	west0989	989	9.1056e-7	**1.4321e-7**	9.2937e-7	1.1738e-6	4.5971e-4	0.5286	0.9077	0.7057	**0.5233**	65.1368
9	tub1000	1000	5.1065e-8	9.5377e-7	**5.0393e-8**	1.4111e-7	7.5573e-5	0.5872	**0.0814**	1.1054	0.7667	6.4924
10	rdb1250	1250	**1.3308e-7**	7.1673e-6	1.3330e-7	5.0640e-6	5.2706e-5	**3.4227**	9.3372	5.2022	4.4787	2.9550
11	rdb1250l	1250	1.7334e-8	2.9591e-6	**1.5632e-8**	4.7407e-6	3.0654e-5	**1.4015**	4.9601	2.4995	4.0606	1.0916
12	mahindas	1258	9.1371e-5	**9.0167e-10**	8.4940e-5	7.5615e-5	1.2418e-2	2.0424	**0.3481**	2.9327	3.4825	14.8657
13	west1505	1505	5.6225e-7	**3.2082e-7**	5.7380e-7	1.3148e-6	9.6668e-4	1.1789	1.0767	1.2412	**0.7941**	208.224
14	west2021	2021	9.9915e-7	**2.4175e-7**	9.6340e-7	9.6909e-7	5.4344e-4	1.7936	1.7572	**0.5231**	1.1817	458.994
15	dw2048	2048	**2.8063e-9**	6.7402e-6	3.2703e-9	1.3635e-6	1.2467e-4	**0.4949**	0.6153	0.8453	1.0875	1.3047
16	rdb2048	2048	**1.9521e-7**	3.6344e-6	1.9949e-7	4.3934e-6	1.7117e-5	**4.1138**	8.3240	6.9954	6.1358	5.4121
17	rdb2048l	2048	1.1858e-8	1.4351e-6	**1.1147e-8**	5.6413e-6	2.4557e-6	**3.3448**	4.4478	6.8582	3.9519	2.8792
18	rdb3200l	3200	9.2192e-9	1.0654e-6	**8.4147e-9**	3.9040e-6	1.7606e-5	**5.3747**	6.3130	11.2058	6.9516	7.8334
	Average							**1.6715**	2.4899	2.4224	2.2068	49.8306

2.5.2.2 Ei-DCA3 and DCA-NLP

We compare Ei-DCA3 (the best among our DCA versions) with the DCA for NLP1 in [76], which solves the problem

$$\min\{F(u,v,y,w) = \|u - vy\|^2 + y^T w : (u,v,y,w) \in \mathcal{C}\}. \tag{2.69}$$

Two DC decompositions of F have been introduced in [76]. We have tested the corresponding DCA schemes on the 18 above problems and observed that the DCA corresponding to the following DC decomposition is better

$$F(u,v,y,w) = G(u,v,y,w) - H(u,v,y,w),$$

where (ρ_1, ρ_2 are given above)

$$\begin{aligned} G(u,v,y,w) &= \left(\frac{\rho_1}{2} + 1\right)\|u\|^2 + \frac{\rho_1 + \rho_2}{2} v^2 + \frac{\rho_1 + \rho_2}{2}\|y\|^2 + \frac{\|y + w\|^2}{4}, \\ H(u,v,y,w) &= G(u,v,y,w) - F(u,v,y,w). \end{aligned}$$

Therefore, we compare Ei-DCA3 with the DCA corresponding to this decomposition.

We are interested in the value of OBJ $= \|u - vy\|^2 + y^T w$ obtained by the two comparative DCA schemes and their running time. Note that, (λ, y) is a solution to the EiCP iff $(y/\lambda, 1/\lambda, y, w)$ is a solution to the problem (2.69) with zero objective value. Thus, OBJ $= 0$ means $(1/v, y)$ is an optimal solution to the EiCP. For DCA-NLP, we choose all tolerances to be 10^{-6} in order to get the better values of OBJ. The stopping condition of Ei-DCA3 is the same as that of DCA-NLP. The computational results are presented in the Table 2.5. It can be observed from these results that:

- In terms of the value of OBJ, Ei-DCA3 slightly outperforms DCA-NLP in most of the test problems, namely 12/18 problems. Ei-DCA3 (resp. DCA-NLP) provides a solution to the EiCP with the accuracy less than 10^{-6} in 15/18 problems (resp. 12/18).
- In terms of computing time, Ei-DCA3 is faster than DCA-NLP in 12/18 problems. On average, Ei-DCA3 is 1.62 times faster than DCA-NLP.

2.6 Conclusions

We have proposed four DCA schemes to handle a large class of MPLCC, namely DCLCC. Using four penalty functions based on the min/Fischer-Burmeister function to penalize the complementarity constraints, we reformulated the DCLCC as standard DC programs. Two of which were reformulated again as general DC programs (to get more suitable DCA versions). The DCA schemes were developed to solve the resulting DC programs. Numerical experiments on several QPLCCs in MacMPEC and EiCPs with data taken from Matrix Market prove the efficiency of the proposed algorithms

Table 2.5 – Comparative results of Ei-DCA3 and DCA-NLP on EiCPs. Better results are written in bold.

Problem			OBJ		computing time (sec.)	
No.	name	n	Ei-DCA3	DCA-NLP	Ei-DCA3	DCA-NLP
1	dwa512	512	**5.5293e-6**	2.1564e-5	**1.2307**	2.1237
2	dwb512	512	4.1885e-9	**4.1418e-9**	**0.0787**	0.1310
3	west0655	655	**1.4025e-6**	1.9909e-5	**0.3956**	1.5826
4	qh768	768	**1.0460e-11**	9.4662e-7	**0.0692**	0.3435
5	rdb800l	800	**2.5274e-8**	5.2809e-8	0.1691	**0.1662**
6	qh882	882	**8.6202e-8**	8.2785e-7	**0.1403**	0.3921
7	rdb968	968	**6.2662e-7**	7.1265e-7	0.2465	**0.2103**
8	west0989	989	**9.2947e-7**	6.5392e-6	**0.7206**	1.0402
9	tub1000	1000	5.3887e-8	**9.2770e-9**	**0.0818**	0.2995
10	rdb1250	1250	1.7966e-7	**5.2805e-8**	0.3572	**0.3548**
11	rdb1250l	1250	**1.7213e-8**	3.8323e-8	**0.3079**	0.3472
12	mahindas	1258	8.4940e-5	**7.0250e-5**	**2.9528**	4.6294
13	west1505	1505	**5.7388e-7**	5.0368e-6	**1.3004**	1.7622
14	west2021	2021	**9.6340e-7**	4.1319e-6	**0.5227**	2.5443
15	dw2048	2048	4.3089e-9	**3.8237e-9**	**0.4497**	0.5972
16	rdb2048	2048	2.6495e-7	**3.3079e-8**	0.7130	**0.6755**
17	rdb2048l	2048	**1.1761e-8**	2.3748e-8	0.6927	**0.5891**
18	rdb3200l	3200	**1.1036e-8**	1.7553e-8	1.2316	**1.1173**
	Average				**0.6478**	1.0503

(especially general DCAs) and their superiority in comparing with the KNITRO solver and a previous DCA scheme for the asymmetric eigenvalue complementarity problems. Our approaches may also be applied to DCLCC problems with a large number of complementarity constraints. The promising numerical results suggested us to investigate, in future works, these DCA schemes for other applications of DCLCC.

Chapter 3

DC Programs with variational inequality constraints

Abstract: *We address a class of Mathematical Programs with Equilibrium Constraints (MPECs) in which the objective function is a DC function, the constraint set is defined by a variational inequality and the Cartesian product of a polytope and a compact convex set. Furthermore, we investigate a particular case where the objective function is continuously differentiable with Lipschitz continuous gradient. By using a penalty technique, we reformulate the considered problem as a DC program. A variant of DCA and its accelerated version are proposed to solve this DC program. Numerical experiments on the second-best toll pricing problem with fixed demands illustrate the efficiency of the proposed algorithms.*

3.1 Introduction

We consider a class of MPECs of the form:

$$\begin{aligned} \min \quad & f(x,y) = g(x,y) - h(x,y) && (3.1)\\ \text{s.t.} \quad & x \in X, y \in Y, \\ & F(x,y)^\top (v - x) \geq 0 \ \forall v \in X, && (3.2) \end{aligned}$$

where g, $h : \mathbb{R}^{n+m} \to \mathbb{R}$ are convex functions, $F : \mathbb{R}^{n+m} \to \mathbb{R}^n$ is a continuously differentiable function, X is a polytope and Y is a compact convex set. We further assume that the partial Jacobian matrices J_xF, J_yF of F are Lipschitz continuous on Ω with $\Omega = X \times Y$.

This type of problems appears in several applications in transportation and economics such as the second-best toll pricing problem [40], the continuous network design problem [54], the Nash-Cournot oligopolistic equilibrium market model [73]. In general,

1. The material of this chapter is developed from the following work:
Thi Minh Tam Nguyen, Hoai An Le Thi. DCA based Algorithms for Solving a Class of Mathematical Programs with Equilibrium Constraints *Submitted.*

the MPEC problem (3.1) is hard to solve due to the variational inequality constraint (3.2), which is nonconvex and is not explicitly given as a constraint of a standard optimization problem.

Related works

An approach to deal with the problem (3.1) is to reformulate it as a Mathematical Program with Complementarity Constraints (MPCC). The resulting problem was solved by smoothing methods [15, 25], the penalty interior-point algorithm, piecewise sequential quadratic programming method [66], the penalty techniques [65], etc. The drawback of this approach is that it increases the number of variables of the original problem and when $F(x, y)$ is not affine, apart from complementarity constraints, the remaining constraint set of the resulting MPCC is nonconvex. Another approach is to use penalty techniques [26, 68, 105]. This approach replaces the variational inequality constraint with a smooth/nonsmooth equation and penalize it. The penalized problems in general are nonconvex, and hence they are still difficult. In this direction, only a few works in the literature offer algorithms to compute their solutions. In [68], the authors studied the properties of a class of penalty functions, both exact and inexact, and proved that the exact penalty holds in the case where the objective function is continuously differentiable. The MPEC problem (3.1) was also solved by replacing v in (3.2) with the extreme points of X and employing the cutting constraint algorithm as in [40]. Nevertheless, at each iteration of this algorithm, we have to deal with a master problem and a subproblem, where the former is nonconvex. When the solution set of the variational inequality (3.2) is singleton, the MPEC problem (3.1) can be restated as an optimization problem in the variable y. This optimization problem was solved by numerical methods based on nonsmooth optimization techniques [77, 78], gradient-based methods [11]. If, in addition, Y is a box, then the problem in the variable y was solved by sensitivity analysis based heuristic algorithms [20]. Some algorithms based on branch-and-bound have been proposed to find a global soltion to the MPEC problem (3.1) in the case where $F(x, y)$ is affine and $f(x, y)$ is convex (see, e.g., [72, 73, 80]).

Our contributions

In this chapter, we propose two algorithms based on DC programming and DCA to solve the MPEC problem (3.1). We use the inexact penalty function in [68] to penalize the variational inequality constraint. The reason for this choice is that this penaty function is continuously differentiable with Lipschitz continuous gradient on Ω, and therefore it is a DC function on Ω. This property of the penalty function permits us to reformulate the penalized problem as a DC program. The DCA scheme applied to this DC program requires computing a parameter ρ which can be obtained via the Lipschitz constant of the gradient of the penalty function. In practice, this Lipschitz constant is usually estimated by a quite large value that could make the DCA inefficient. Therefore, we propose a variant of DCA, named DCA_ρ, in which ρ is updated at each iteration. This algorithm is inspired from DCA-like scheme proposed in [50]. Moreover, recently, acceleration techniques have been developed for improving the convergence rate of proximal gradient algorithms (see, e.g., [8, 55, 104]), DCA and DCA-like [84, 50]. Motivated by the success of the accelerated algorithms for solving convex/nonconvex optimization problems, we also offer an accelerated version of DCA_ρ, called ADCA_ρ, that incorporates an extrapolation step into DCA_ρ. We prove that the convergence of DCA is still valid for the proposed algorithms, i.e. every limit point of the sequence generated by DCA_ρ/ADCA_ρ is a critical point of the penalized problem. Afterward, we consider a special case where the objective function has Lipschitz continuous gradient. The variants of DCA in this case lead to the simple subproblems. They can be decomposed into two problems whose solutions are the projection of a point on the polytope X or on the compact convex set Y. As an application, we solve the second-best toll pricing problem with fixed demands. For this problem, Y is a box and consequently the projection of a point on this set can be explicitly computed. The numerical results on some data show that our algorithms are quite promising.

The rest of the chapter is organized as follows. Section 3.2 presents the solution methods for the considered MPEC problem. Section 3.3 considers a special case where the objective function has Lipschitz continuous gradient. Section 3.4 shows how to apply the proposed methods to solve the second-best toll pricing problem with fixed demands. The numerical results are reported in Section 3.5 and the conclusions are given in the last section.

3.2 Solution methods

3.2.1 Reformulation of the MPEC (3.1)

Consider the function:

$$\theta_\gamma(x,y) = \max_{v\in X} \{\psi_\gamma(x,y,v)\},$$

where

$$\psi_\gamma(x,y,v) = F(x,y)^\top(x-v) - \frac{\gamma}{2}\|x-v\|^2 \text{ and } \gamma > 0.$$

By Theorem 10.2.3 in [16], this function has the following properties.

1) $\forall (x,y) \in \mathbb{R}^{n+m}$,

$$\begin{aligned} M_\gamma(x,y) &:= \{v \in X | \theta_\gamma(x,y) = \psi_\gamma(x,y,v)\} \\ &= \left\{\Pi_X\left(x - \frac{1}{\gamma}F(x,y)\right)\right\}, \end{aligned}$$

where $\Pi_X(v)$ is the projection of v on the set X.

2) $\theta_\gamma(x,y) \geq 0, \forall (x,y) \in X \times \mathbb{R}^m$.

3) $[\theta_\gamma(x,y) = 0, x \in X] \Leftrightarrow x \in X$ and (x,y) satisfies the constraint (3.2).

4) $\theta_\gamma(x,y)$ is continuously differentiable on $\mathbb{R}^{n+m}$ and

$$\nabla\theta_\gamma(x,y) = \nabla_{(x,y)}\psi_\gamma\left(x,y,\Pi_X\left(x - \frac{1}{\gamma}F(x,y)\right)\right).$$

More specifically, the partial gradients of θ_γ with respect to x and y are computed as follows:

$$\nabla_x \theta_\gamma(x,y) = [J_x F(x,y)]^\top (x - v(x,y)) + F(x,y) - \gamma(x - v(x,y)), \tag{3.3}$$
$$\nabla_y \theta_\gamma(x,y) = [J_y F(x,y)]^\top (x - v(x,y)). \tag{3.4}$$

with $v(x,y) = \Pi_X \left(x - \frac{1}{\gamma} F(x,y) \right)$.
Furthermore, the Lipschitz continuity of $\nabla\theta_\gamma(x,y)$ is shown in the following proposition.

Proposition 3.1. *Suppose that F is continuously differentiable on $\mathbb{R}^{n+m}$ and the partial Jacobian matrices J_xF, J_yF of F are Lipschitz continuous on Ω with the Lipschitz constants being L_1, L_2, respectively. Then the gradient of $\theta_\gamma(x,y)$ is Lipschitz continuous on Ω.*

Proof. For all $(x,y), (x',y') \in \Omega$ we have

$$\begin{aligned}\|\nabla\theta_\gamma(x,y) - \nabla\theta_\gamma(x',y')\| &\le \|\nabla_x\theta_\gamma(x,y) - \nabla_x\theta_\gamma(x',y')\| + \|\nabla_y\theta_\gamma(x,y) - \nabla_y\theta_\gamma(x',y')\| \\ &\le D1 + D2 + \|\nabla_y\theta_\gamma(x,y) - \nabla_y\theta_\gamma(x',y')\|, \qquad (3.5)\end{aligned}$$

with

$$D_1 = \|[J_xF(x,y)]^\top(x - v(x,y)) - [J_xF(x',y')]^\top(x' - v(x',y'))\|, \tag{3.6}$$
$$D_2 = \|F(x,y) - F(x',y')\| + \gamma[\|x - x'\| + \|v(x,y) - v(x',y')\|]. \tag{3.7}$$

Because of the nonexpansiveness of Π_X, one gets

$$\begin{aligned}\|v(x,y) - v(x',y')\| &= \|\Pi_X\left(x - \frac{1}{\gamma}F(x,y)\right) - \Pi_X\left(x' - \frac{1}{\gamma}F(x',y')\right)\| \\ &\le \|x - x'\| + \frac{1}{\gamma}\|F(x,y) - F(x',y')\|, \qquad (3.8)\end{aligned}$$

and therefore

$$D_2 \le 2\gamma\|x - x'\| + 2\|F(x,y) - F(x',y')\|. \tag{3.9}$$

Moreover, we have

$$\begin{aligned}D_1 \le&\|[J_xF(x,y)]^\top(x - v(x,y)) - [J_xF(x',y')]^\top(x - v(x,y))\| + \\ &\|[J_xF(x',y')]^\top(x - v(x,y)) - [J_xF(x',y')]^\top(x' - v(x',y'))\| \\ \le&\|J_xF(x,y) - J_xF(x',y')\|\|x - v(x,y)\| + \\ &\|J_xF(x',y')\|[\|x - x'\| + \|v(x,y) - v(x',y')\|] \\ \le&\|J_xF(x,y) - J_xF(x',y')\|\|\frac{1}{\gamma}F(x,y)\| + \\ &\|J_xF(x',y')\|\left[2\|x - x'\| + \frac{1}{\gamma}\|F(x,y) - F(x',y')\|\right]. \qquad (3.10)\end{aligned}$$

Similarly,

$$\begin{aligned}
&\|\nabla_y \theta_\gamma(x,y) - \nabla_y \theta_\gamma(x',y')\| \\
=&\|[J_yF(x,y)]^\top (x - v(x,y)) - [J_yF(x',y')]^\top (x' - v(x',y'))\| \\
\le&\|J_yF(x,y) - J_yF(x',y')\| \|\frac{1}{\gamma}F(x,y)\| + \\
&\|J_yF(x',y')\| \left[2\|x - x'\| + \frac{1}{\gamma}\|F(x,y) - F(x',y')\|\right]. \qquad (3.11)
\end{aligned}$$

It follows from (3.5) and (3.9)-(3.11) that

$$\begin{aligned}
&\|\nabla \theta_\gamma(x,y) - \nabla \theta_\gamma(x',y')\| \\
\le&\frac{1}{\gamma}\|F(x,y)\| \left(\|J_xF(x,y) - J_xF(x',y')\| + \|J_yF(x,y) - J_yF(x',y')\|\right) + \\
&2\|x - x'\| \left(\|J_xF(x',y')\| + \|J_yF(x',y')\| + \gamma\right) + \\
&\|F(x,y) - F(x',y')\| \left(\frac{\|J_xF(x',y')\|}{\gamma} + \frac{\|J_yF(x',y')\|}{\gamma} + 2\right). \qquad (3.12)
\end{aligned}$$

Since F is continuously differentiable, F is locally Lipschitz continuous on $\mathbb{R}^{n+m}$. Because of the compactness of Ω, F is Lipschitz continuous on Ω. Assume that the Lipschitz constant of F is L_3. The continuous differentiability of F also results in the boundedness of the functions F, J_xF, J_yF on Ω. Thus, there exist the constants M, M_1, M_2 such that

$$\|F(x,y)\| \le M, \ \|J_xF(x,y)\| \le M_1, \|J_yF(x,y)\| \le M_2 \ \forall (x,y) \in \Omega,$$

and consequently for all $(x,y), (x',y') \in \Omega$, we have

$$\begin{aligned}
&\|\nabla \theta_\gamma(x,y) - \nabla \theta_\gamma(x',y')\| \\
\le&\frac{1}{\gamma}M\left(L_1 + L_2\right)\|(x,y) - (x',y')\| + 2\|x - x'\| \left(M_1 + M_2 + \gamma\right) + \\
&L_3\left(\frac{M_1 + M_2}{\gamma} + 2\right)\|(x,y) - (x',y')\|. \qquad (3.13)
\end{aligned}$$

It follows that

$$\|\nabla \theta_\gamma(x,y) - \nabla \theta_\gamma(x',y')\| \le L\|(x,y) - (x',y')\|$$

with $L = \frac{1}{\gamma}M\left(L_1 + L_2\right) + 2\left(M_1 + M_2 + \gamma\right) + L_3\left(\frac{M_1 + M_2}{\gamma} + 2\right)$.
This implies that $\nabla \theta_\gamma(x,y)$ is Lipschitz continuous on Ω. □

Because of the third property of the function θ_γ, the problem (3.1) is equivalent to the problem:

$$\begin{aligned}
\min \quad & f(x,y) \qquad (3.14)\\
\text{s.t.} \quad & x \in X, y \in Y \\
& \theta_\gamma(x,y) = 0.
\end{aligned}$$

We approximate the problem (3.14) by the following penalized problem (P_t)

$$\min\{F_t(x,y) = f(x,y) + t\theta_\gamma(x,y) : x \in X, y \in Y\}, \tag{3.15}$$

where $t > 0$ is a penalty parameter.
This technique was used in [68]. However, the authors only proved the theoretical results and did not give a solution method for the penalized problem. It follows from the results in the above paper that for a large number t, the optimal solutions of the problem (3.15) will be in a region where $\theta_\gamma(x,y)$ are relatively small. Therefore, we consider the penalized problem (P_t) with sufficiently large value of t.
For simplicity, we denote $z = (x,y)$.

Definition 3.1. *A point z^* is called a critical point of the problem* (3.15) *if*

$$(\partial g(z^*) + t\nabla\theta_\gamma(z^*) + N_\Omega(z^*)) \cap \partial h(z^*) \neq \emptyset.$$

For each positive number ρ, the function $F_t(z)$ can be written in the form:

$$F_t(z) = G_\rho(z) - H_\rho(z),$$

where

$$G_\rho(z) = g(z) + \frac{\rho}{2}\|z\|^2, H_\rho(z) = h(z) + \frac{\rho}{2}\|z\|^2 - t\theta_\gamma(z).$$

Obviously, $G_\rho(z)$ is strongly convex. It is easy to see that if $\rho \geq tL$ with L being the Lipschitz constant of $\nabla\theta_\gamma(z)$, then $H_\rho(z)$ is convex on Ω. Thus, if we choose $\rho \geq tL$, then the problem (3.15) can be reformulated as the DC program below:

$$\min\{F_t(z) = G_\rho(z) - H_\rho(z) : z \in \Omega\}. \tag{3.16}$$

3.2.2 DCA based algorithms

Based on the generic scheme of standard DCA presented in Chapter 1, each iteration of DCA applied to the problem (3.16) consists of first computing a subgradient $\overline{z}^k \in \partial H_\rho(z^k)$ and then solving the following convex subproblem:

$$\min\left\{g(z) + \frac{\rho}{2}\|z\|^2 - \langle \overline{z}^k, z\rangle : z \in \Omega\right\}. \tag{3.17}$$

Note that

$$\partial H_\rho(z) = \partial h(z) + \rho z - t\nabla\theta_\gamma(z), \tag{3.18}$$

where the partial gradients of $\theta_\gamma(z)$ is computed by (3.3), (3.4).
DCA applied to (3.16) is described as follows.

Algorithm 3.1 DCA_MPEC

```
Initialization: Choose an initial point $z^0 \in \Omega$, a number $\rho \geq tL$ and a sufficiently small positive number $\varepsilon$. Set $k := 0$.
repeat
```

```
    1. Compute $\overline{z}^k \in \partial H_\rho(z^k)$ using (3.18).
    2. Solve the convex subproblem (3.17) to obtain $z^{k+1}$.
    3. $k := k+1$.
until $\|z^k - z^{k-1}\| \le \varepsilon \max(\|z^{k-1}\|, 1)$.
```

The convergence properties of Algorithm 3.1 is shown in the following theorem.

Theorem 3.1. *Suppose that* $\{z^k\}$ *is the sequence generated by Algorithm 3.1. Then the following statements hold:*
i) *The sequence* $\{F_t(z^k)\}$ *is decreasing.*
ii) *Every limit point of the sequence* $\{z^k\}$ *is a critical point of the problem* (3.15).

It is noticeable that the statements in this theorem can not be directly deduced from Theorem 3 in [81], which requires the convexity of H_ρ on $\mathbb{R}^n$, because in our problem, H_ρ is only convex on Ω. However, thanks to the strong convexity of G_ρ, we can prove Theorem 3.1 as follows.

Proof. i) Clearly, $\{z^k\} \subset \Omega$. For each k, z^{k+1} is the solution to (3.17), therefore

$$\overline{z}^k \in \partial G_\rho(z^{k+1}) + N_\Omega(z^{k+1}). \tag{3.19}$$

Since $G_\rho + \chi_\Omega$ is $\rho-$convex, we have:

$$G_\rho(z^k) \ge G_\rho(z^{k+1}) + \langle \overline{z}^k, z^k - z^{k+1}\rangle + \frac{\rho}{2}\|z^k - z^{k+1}\|^2, \tag{3.20}$$

On the other hand, $\overline{z}^k \in \partial H_\rho(z^k)$, therefore $\overline{z}^k = u^k + \rho z^k - t\nabla\theta_\gamma(z^k)$, where $u^k \in \partial h(z^k)$. Because of the convexity of h, one has

$$h(z^{k+1}) \ge h(z^k) + \langle u^k, z^{k+1} - z^k\rangle. \tag{3.21}$$

Moreover, $\frac{\rho}{2}\|z\|^2 - t\theta_\gamma(z)$ is differentiable on $\mathbb{R}^{n+m}$ and convex on Ω, hence

$$\frac{\rho}{2}\|z^{k+1}\|^2 - t\theta_\gamma(z^{k+1}) \ge \frac{\rho}{2}\|z^k\|^2 - t\theta_\gamma(z^k) + \langle \rho z^k - t\nabla\theta_\gamma(z^k), z^{k+1} - z^k\rangle. \tag{3.22}$$

From (3.21) and (3.22) we get

$$H_\rho(z^{k+1}) \ge H_\rho(z^k) + \langle \overline{z}^k, z^{k+1} - z^k\rangle. \tag{3.23}$$

This combined with (3.20) leads to the following inequality:

$$F_t(z^k) - F_t(z^{k+1}) \ge \frac{\rho}{2}\|z^k - z^{k+1}\|^2. \tag{3.24}$$

which implies that the sequence $\{F_t(z^k)\}$ is decreasing.
ii) Since $\{F_t(z^k)\}$ is decreasing and is bounded below by the optimal value of the problem (3.16), it converges. It follows that

$$\lim_{k\to\infty}(z^k - z^{k+1}) = 0.$$

Assume that z^* is a limit point of $\{z^k\}$, then there exists a subsequence $\{z^{k_j}\}$ of $\{z^k\}$ such that

$$\lim_{j\to\infty} z^{k_j} = z^*.$$

We will prove that the sequence $\{u^k\}$ is bounded. Indeed, for each k, let $\lambda^k = u^k/\|u^k\|$, then $\{\lambda^k\}$ is bounded. As $u^k \in \partial h(z^k)$, the following inequality holds

$$h(z^k + \lambda^k) \geq h(z^k) + \langle u^k, \lambda^k\rangle \ \forall k.$$

This implies that

$$\|u^k\| \leq h(z^k + \lambda^k) - h(z^k) \ \forall k.$$

Since $h : \mathbb{R}^n \to \mathbb{R}$ is convex, it is continuous on $\mathbb{R}^n$. Owing to the boundedness of the sequences $\{z^k + \lambda^k\}, \{z^k\}$ and the continuity of h, the sequence $\{u^k\}$ is bounded. Without loss of generality, assume that $\lim\limits_{j\to\infty} u^{k_j} = u^*$. Then $u^* \in \partial h(z^*)$.
By (3.19), we have

$$u^{k_j} + \rho z^{k_j} - t\nabla\theta_\gamma(z^{k_j}) \in \partial g(z^{k_j+1}) + \rho z^{k_j+1} + N_\Omega(z^{k_j+1}),$$

which is equivalent to

$$u^{k_j} + \rho(z^{k_j} - z^{k_j+1}) - t\nabla\theta_\gamma(z^{k_j}) \in \partial g(z^{k_j+1}) + N_\Omega(z^{k_j+1}).$$

Let j approach infinity we get

$$u^* - t\nabla\theta_\gamma(z^*) \in \partial g(z^*) + N_\Omega(z^*),$$

which implies that z^* is a critical point of the problem (3.15). □

Nevertheless, in practice, the Lipschitz constant of the function $\nabla\theta_\gamma(z)$ may be not computable and is usually estimated by a quite large value that could make DCA_MPEC inefficient. Therefore, we update the parameter ρ at each iteration. In DCA_MPEC, since $\rho \geq tL$, we have

$$H_\rho(z) \geq H_\rho(z^k) + \langle \overline{z}^k, z - z^k\rangle \ \forall z \in \Omega, \ \forall k. \tag{3.25}$$

When updating ρ, at each iteration k, we only need to find a number ρ^{k+1} such that for $\rho = \rho^{k+1}$, the inequality (3.25) holds at $z = z^{k+1}$. The strategy for updating the parameter ρ is described in the following algorithm.

Algorithm 3.2 DCA_ρ

```
Initialization: Choose an initial point $z^0 \in \Omega$, an initial parameter $\rho^0 > 0$, parameters
$\eta_1, \eta_2 > 1$, and a sufficiently small positive number $\varepsilon$. Set $k := 0$.
repeat
    1. Let $\rho = \max(\rho^0, \rho^k/\eta_1)$.
    2. Compute $\overline{z}^k \in \partial H_\rho(z^k)$ using (3.18).
    3. Compute $z^{k+1}$ as follows
        3.1 Compute $\tilde{z}$ as the solution to the convex subproblem (3.17).
        3.2 **While** $H_\rho(\tilde z) < H_\rho(z^k) + \langle \overline{z}^k, \tilde z - z^k\rangle$ **do**
            3.2.1. Update $\rho := \eta_2\rho$.
            3.2.2. Compute $\overline{z}^k$ as in the step 2.
            3.2.3. Compute $\tilde z$ as in the step 3.1.
            **End While**
        3.3 Update $z^{k+1} = \tilde z, \rho^{k+1} = \rho$.
    4. $k := k+1$.
**until** $\|z^k - z^{k-1}\| \le \varepsilon \max(\|z^{k-1}\|, 1)$.
```

Remark 3.1.

- *Updating the parameter ρ in step 3 terminates after a finite number of iterations because for all $\rho \ge tL$ we have*

$$H_\rho(\tilde z) \ge H_\rho(z^k) + \langle \overline{z}^k, \tilde z - z^k\rangle, \forall \tilde z \in \Omega.$$

- *The sequence $\{\rho^k\}$ is bounded, more specifically, $\rho^0 \le \rho^k \le \eta_2 tL$, $\forall k$.*

- *It is worth noting that updating ρ does not guarantee that H_ρ is convex, however, the convergence properties of DCA_MPEC is still valid for DCA_ρ. This is shown in the following theorem.*

Theorem 3.2. *Suppose that $\{z^k\}$ is the sequence generated by the DCA_ρ algorithm. Then the following statements hold:*

i) $F_t(z^k) - F_t(z^{k+1}) \ge \frac{\rho^{k+1}}{2}\|z^k - z^{k+1}\|^2, \forall k$. *This results in the sequence $\{F_t(z^k)\}$ being decreasing.*

ii) *Every limit point of the sequence $\{z^k\}$ is a critical point of the problem* (3.15).

Proof. i) Use the same reasoning as in the proof of Theorem 3.1 i) but ρ is replaced by ρ^{k+1} and the inequality (3.23) is deduced from the update rule.
ii) Use the same reasoning as in the proof of Theorem 3.1 ii) but ρ is replaced with ρ^{k_j+1}. □

The accelerated version of DCA_ρ is described in the algorithm below.

Algorithm 3.3 ADCA_ρ

```
**Initialization:** Choose an initial point $z^0 \in \Omega$, parameters $t_0 = 1, \rho^0 > 0; \eta_1, \eta_2 > 1$, and a sufficiently small positive number $\varepsilon$. Set $k = 0, z^{-1} = z^0$.
**repeat**
    1. Compute $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.
    2. Compute $\hat z^k = z^k + \tau^k(z^k - z^{k-1})$, with $\tau^k = \frac{t_k - 1}{t_{k+1}}$.
    3. Let $\Gamma^k = \max\{F_t(z^l) : l = \max(0, k-q), ..., k\}$.
       If $\hat z^k \in \Omega$ and $F_t(\hat z^k) \le \Gamma^k$ then set $w^k = \hat z^k$, else set $w^k = z^k$.
    4. Let $\rho = \max(\rho^0, \rho^k/\eta_1)$.
    5. Compute $\overline{z}^k \in \partial H_\rho(w^k)$ using (3.18).
    6. Compute $z^{k+1}$ as follows
       6.1 Compute $\tilde{z}$ as the solution to the convex subproblem (3.17).
       6.2 **While** $H_\rho(\tilde{z}) < H_\rho(w^k) + \langle \overline{z}^k, \tilde{z} - w^k \rangle$ **do**
           6.2.1. Update $\rho := \eta_2 \rho$.
           6.2.2. Compute $\overline{z}^k$ as in the step 5.
           6.2.3. Compute $\tilde{z}$ as in the step 6.1.
           **End While**
       6.3 Update $z^{k+1} = \tilde{z}, \rho^{k+1} = \rho$.
    7. $k := k + 1$.
**until** $\|z^k - z^{k-1}\| \le \varepsilon \max(\|z^{k-1}\|, 1)$.
```

Remark 3.2. *The acceleration technique in this algorithm is same as that in the ADCA* [84] *and the niAPG algorithm* [104]. *However, in the ADCA (resp. the niAPG algorithm) this acceleration technique is incorporated into the DCA (resp. the proximal gradient algorithm) while our algorithm incorporates this acceleration technique into DCA_ρ, which is not a DCA since updating ρ does not guarantee the successive decompositions of $F_t(z)$ are DC decomposition.*

The next theorem indicates the convergence of ADCA_ρ.

Theorem 3.3. *Suppose that $\{z^k\}$ and $\{w^k\}$ are the sequences generated by ADCA_ρ. The following statements hold.*

i) $\Gamma^k - \Gamma^{k+q+1} \ge \dfrac{\rho^0}{2}\|z^{\phi(k)} - w^{\phi(k)-1}\|^2$, *where*

$$\phi(k) = argmin\left\{\|z^l - w^{l-1}\|^2 : l = k+1, ..., k+q+1\right\}.$$

ii) *Every limit point of the sequence $\{z^{\phi(k)}\}$ is a critical point of the problem* (3.15).

Proof. i) According to the way of constructing the sequence $\{w^k\}, w^k \in \Omega$ for all k. Using the same reasoning as in the proof of Theorem 3.1 i) in which z^k, ρ are replaced by w^k, ρ^{k+1} respectively, we obtain

$$\overline{z}^k \in \partial G_{\rho^{k+1}}(z^{k+1}) + N_\Omega(z^{k+1}), \tag{3.26}$$

$$F_t(w^k) - F_t(z^{k+1}) \ge \frac{\rho^{k+1}}{2}\|z^{k+1} - w^k\|^2, \forall k = 0, 1, ... \tag{3.27}$$

The inequality (3.27) combined with $F_t(w^k) \le \Gamma^k$ and $\rho^{k+1} \ge \rho^0$ implies that

$$F_t(z^{k+1}) \le \Gamma^k - \frac{\rho^0}{2}\|z^{k+1} - w^k\|^2, \forall k = 0, 1, ... \tag{3.28}$$

We will prove by induction that

$$F_t(z^{k+1+l}) \le \Gamma^k - \frac{\rho^0}{2}\|z^{k+1+l} - w^{k+l}\|^2, \tag{3.29}$$

for all $l = 0, 1, ..., q$. Indeed, (3.29) holds for $l = 0$ because of (3.28). Assume that (3.29) holds for $l = 0, 1, ..., p-1$ with $1 \le p \le q$, we have

$$F_t(z^{k+1+l}) \le \Gamma^k, \forall l = 0, 1, ..., p-1.$$

Moreover, substitute k by $k+p$ in (3.28) we get

$$\begin{aligned} F_t(z^{k+p+1}) &\le \Gamma^{k+p} - \frac{\rho^0}{2}\|z^{k+p+1} - w^{k+p}\|^2 \\ &\le \max\left\{\Gamma^k, F_t(z^{k+1}), ..., F_t(z^{k+p})\right\} - \frac{\rho^0}{2}\|z^{k+p+1} - w^{k+p}\|^2, \end{aligned}$$

therefore

$$F_t(z^{k+p+1}) \le \Gamma^k - \frac{\rho^0}{2}\|z^{k+p+l} - w^{k+p}\|^2,$$

that means (3.29) holds for $l = p$, and consequently for all $l = 0, 1, ..., q$. It follows that

$$\begin{aligned} \Gamma^{k+q+1} &= \max\left\{F_t(z^l) : l = k+1, ..., k+q+1\right\} \\ &\le \Gamma^k - \frac{\rho^0}{2}\min\left\{\|z^{k+1+l} - w^{k+l}\|^2 : l = 0, ..., q\right\} \end{aligned}$$

which is equivalent to

$$\Gamma^k - \Gamma^{k+q+1} \ge \frac{\rho^0}{2}\|z^{\phi(k)} - w^{\phi(k)-1}\|^2. \tag{3.30}$$

ii) Let α be the optimal value of the problem (3.15) then $\alpha > -\infty$. Summing (3.30) over $k = 0, ..., N (N > q)$, we get

$$\begin{aligned} \frac{\rho^0}{2}\sum_{k=0}^{N}\|z^{\phi(k)} - w^{\phi(k)-1}\|^2 &\le \sum_{l=0}^{q}(\Gamma^l - \Gamma^{N+l+1}) \le \sum_{l=0}^{q}\Gamma^l - (q+1)\alpha \\ &\le (q+1)\left[\max\{F_t(z^l) : l = 0, ..., q\} - \alpha\right]. \end{aligned}$$

This leads to the following inequality

$$\sum_{k=0}^{\infty}\|z^{\phi(k)} - w^{\phi(k)-1}\|^2 < \infty,$$

and hence

$$\lim_{k\to\infty}\|z^{\phi(k)} - w^{\phi(k)-1}\| = 0. \tag{3.31}$$

Assume that z^* is a limit point of $\{z^{\phi(k)}\}$, then there exists a sequence $\{k_j\}$ such that

$$\lim_{j\to\infty} z^{\phi(k_j)} = z^*.$$

By (3.31), $\lim_{j\to\infty} w^{\phi(k_j)-1} = z^*$.
Substitute k by $\phi(k_j) - 1$ in (3.26) we obtain

$$\overline{z}^{\phi(k_j)-1} \in \partial G_{\rho^{\phi(k_j)}}(z^{\phi(k_j)}) + N_\Omega(z^{\phi(k_j)}). \tag{3.32}$$

Since $\overline{z}^{\phi(k_j)-1} \in \partial H_{\rho^{\phi(k_j)}}(w^{\phi(k_j)-1})$, one has

$$\overline{z}^{\phi(k_j)-1} = u^{\phi(k_j)-1} + \rho^{\phi(k_j)} w^{\phi(k_j)-1} - t\nabla\theta_\gamma(w^{\phi(k_j)-1}),$$

where $u^{\phi(k_j)-1} \in \partial h(w^{\phi(k_j)-1})$. Using the similar argument as in the proof of Theorem 3.1, we have $\{u^{\phi(k_j)-1}\}$ is bounded. Without loss of generality, assume that

$$\lim_{j\to\infty} u^{\phi(k_j)-1} = u^*.$$

Then $u^* \in \partial h(z^*)$. It follows from (3.32) that

$$u^{\phi(k_j)-1} + \rho^{\phi(k_j)}(w^{\phi(k_j)-1} - z^{\phi(k_j)}) - t\nabla\theta_\gamma(w^{\phi(k_j)-1}) \in \partial g(z^{\phi(k_j)}) + N_\Omega(z^{\phi(k_j)}).$$

Let j approach infinity we get

$$u^* - t\nabla\theta_\gamma(z^*) \in \partial g(z^*) + N_\Omega(z^*),$$

which implies that z^* is a critical point of the problem (3.15). □

When $q = 0$ then $\Gamma^k = F_t(z^k)$ and $\phi(k) = k + 1$. Therefore, owing to Theorem 3.3, we have the following corollary.

Corollary 3.1. *Suppose that $\{z^k\}$ is the sequences generated by ADCA_ρ with $q = 0$. The following statements hold.*

i) *The sequence $\{F_t(z^k)\}$ is decreasing.*

ii) *Every limit point of the sequence $\{z^k\}$ is a critical point of the problem (3.15).*

3.3 A particular case: the objective function has Lipschitz continuous gradient

Let us consider the problem (3.1) when $f(x, y)$ is a continuously differentiable function with Lipschitz continuous gradient on Ω. In this case, the functions $G_\rho(z), H_\rho(z)$ can be chosen as follows

$$G_\rho(z) = \frac{\rho}{2}\|z\|^2, H_\rho(z) = \frac{\rho}{2}\|z\|^2 - [f(z) + t\theta_\gamma(z)].$$

Note that if $\rho \geq \overline{L} + tL$ with $\overline{L}$ is the Lipschitz constant of $\nabla f(z)$, then $H_\rho(z)$ is convex on Ω.
Since $H_\rho(z)$ is differentiable, we have

$$\partial H_\rho(z) = \{\nabla H_\rho(z)\},$$

and consequently the subgradient $\overline{z} = (\overline{x}, \overline{y}) \in \partial H_\rho(z)$ is computed by

$$\overline{z} = \rho z - \nabla f(z) - t\nabla\theta_\gamma(z). \tag{3.33}$$

The subproblem (3.17) becomes

$$\min\left\{\frac{\rho}{2}\|z\|^2 - \langle \overline{z}^k, z\rangle : z \in \Omega\right\}. \tag{3.34}$$

As the objective function and the constraints of this problem are separable in the variables x, y, it can be decomposed into the two following problems.

$$\min\left\{\frac{1}{2}\|x\|^2 - \frac{1}{\rho}\langle \overline{x}^k, x\rangle : x \in X\right\}, \tag{3.35}$$

and

$$\min\left\{\frac{1}{2}\|y\|^2 - \frac{1}{\rho}\langle \overline{y}^k, y\rangle : y \in Y\right\} \tag{3.36}$$

Therefore, the solution to the subproblem (3.34) is $\tilde{z} = (\tilde{x}, \tilde{y})$, where $\tilde{x}$ (resp. $\tilde{y}$) is the projection of $\dfrac{\overline{x}^k}{\rho}$ (resp. $\dfrac{\overline{y}^k}{\rho}$) on X (resp. Y).

The DCA_ρ version in this case replaces the steps 2 and 3.1 in Algorithm 3.2 by

2a. Compute $\overline{z}^k = \rho z^k - \nabla f(z^k) - t\nabla\theta_\gamma(z^k)$.

3.1a. Set $\tilde{z} = (\tilde{x}, \tilde{y})$, where

$$\tilde{x} = \Pi_X\left(\frac{\overline{x}^k}{\rho}\right), \tilde{y} = \Pi_Y\left(\frac{\overline{y}^k}{\rho}\right).$$

The ADCA_ρ version in this case replaces the steps 5 and 6.1 in Algorithm 3.3 by

5a. Compute $\overline{z}^k = \rho w^k - \nabla f(w^k) - t\nabla\theta_\gamma(w^k)$.

6.1a. Set $\tilde{z} = (\tilde{x}, \tilde{y})$, where

$$\tilde{x} = \Pi_X\left(\frac{\overline{x}^k}{\rho}\right), \tilde{y} = \Pi_Y\left(\frac{\overline{y}^k}{\rho}\right).$$

3.4 Application to the second-best toll pricing problem with fixed demands

The toll pricing problem consists of determining tolls to reduce congestion. The authors in [40] mentioned that this problem can be classified as the first-best and the second-best toll pricing problem. The former assumes that every link in the network can be tolled, whereas the latter assumes that some links are not tollable. In this application, we are interested in the Second-Best Toll Pricing (SBTP) problem with fixed demands as in [40] but we consider the case where there are upper bounds for the aggregate flows and tolls. This problem aims to determine tolls to minimize the total travel time when the link flows must satisfy the tolled user equilibrium condition. The following notation is used to state the problem:

$\mathcal{A}$	the set of arcs or links in the network.
a	an index for links in the network.
k	an index for origin-destination (OD) pairs.
K	the set containing indices of all OD pairs.
u^k	the vector of link flows for the k^{th} OD pair.
d_k	the fixed travel demand for the k^{th} OD pair.
x	the aggregate flow vector, i.e. $x = \sum_{k\in K} u^k$.
x^U	the upper bound vector of x.
$s(x)$	the travel time vector whose element $s_a(x)$ denotes the travel time for the link a, $s_a(x) = T_a + C_a x_a^4$.
A	the node-arc incidence matrix of the network, i.e. the elements a_{ij} of A are defined by

$$a_{ij} = \begin{cases} 1 & \text{if link } j \text{ starts at node } i \\ -1 & \text{if link } j \text{ ends at node } i \\ 0 & \text{otherwise} \end{cases} \tag{3.37}$$

b_k	a vector defined as follows: If p and q denote the origin and destination nodes of the k^{th} OD pair then $b_k = (e_p - e_q)d_k$, where e_p, e_q are the p^{th} and q^{th} columns of the identity matrix of order l, respectively (l is the number of nodes in the network)
y	a toll vector.
$\mathcal{A}_t$	the set of tollable arcs.
y_a^U	the upper bound of $y_a, a \in \mathcal{A}_t$.

Using the above notation, the set of all feasible flow vectors can be stated as:

$$\hat{X} = \{x : \text{ there exist } u^k \text{ such that } x = \sum_{k\in K} u^k, Au^k = b_k, u^k \geq 0 \ \forall k \in K\}.$$

Let $X = \{x \in \hat{X} : x \leq x^U\}$. Clearly, X is a polytope.

The considered problem can be formulated as the MPEC below:

$$\begin{array}{ll} \min\limits_{(x,y)} & s(x)^T x \\ \text{s.t.} & x \in X, \\ & 0 \leq y_a \leq y_a^U, a \in \mathcal{A}_t, \\ & y_a = 0, a \notin \mathcal{A}_t, \\ & [s(x) + y]^T (v - x) \geq 0 \ \forall v \in X. \end{array} \tag{3.38}$$

Let

$$\begin{aligned} &f(x,y) = s(x)^T x = \sum_{a\in\mathcal{A}} (T_a + C_a x_a^4) x_a, \ F(x,y) = s(x) + y, \\ &Y = [0, y^U] \text{ with convention } y_a^U = 0 \text{ if } a \notin \mathcal{A}_t, \\ &q_1 = \max_{a\in\mathcal{A}}\{C_a\}, q_2 = \max_{a\in\mathcal{A}}\{x_a^U\}. \end{aligned}$$

Two following propositions show that the functions $f(x,y), F(x,y)$ are continuously differentiable and that the gradient of f as well as the partial Jacobian matrices

J_xF, J_yF of F are Lipschitz continuous on Ω. Therefore the problem (3.38) has the form of the MPEC problem (3.1) in the special case.

Proposition 3.2. *The function $f(x,y)$ is continuously differentiable on $\mathbb{R}^{2n}$ (n is the number of links in the network) and its gradient is Lipschitz continuous on Ω.*

Proof. Clearly, $f(x,y) = \sum_{a\in\mathcal{A}} (T_a + C_a x_a^4)x_a$ is continuously differentiable and its partial gradients are

$$\nabla_x f(x,y) = \begin{bmatrix} T_1 + 5C_1x_1^4 \\ \vdots \\ T_n + 5C_nx_n^4 \end{bmatrix}, \nabla_y f(x,y) = 0. \tag{3.39}$$

For all $(x,y), (\tilde{x},\tilde{y}) \in \Omega$ we have

$$\begin{aligned} \|\nabla f(x,y) - \nabla f(\tilde{x},\tilde{y})\| &= \sqrt{\sum_a (5C_a)^2 (x_a^4 - \tilde{x}_a^4)^2} \\ &= \sqrt{\sum_a (5C_a)^2 (x_a - \tilde{x}_a)^2 (x_a + \tilde{x}_a)^2 (x_a^2 + \tilde{x}_a^2)^2} \\ &\le 20q_1q_2^3\|x - \tilde{x}\| \le 20q_1q_2^3\|(x,y) - (\tilde{x},\tilde{y})\|. \end{aligned}$$

It follows that $\nabla f(x,y)$ is Lipschitz continuous on Ω. □

Proposition 3.3. *The function $F(x,y)$ is continuously differentiable on $\mathbb{R}^{2n}$ and its partial Jacobian matrices are Lipschitz continuous on Ω.*

Proof. It is straightforward to see that $F(x,y)$ is continuously differentiable and its partial Jacobian matrices are

$$J_xF(x,y) = \begin{bmatrix} 4C_1x_1^3 & 0 & \dots & 0 \\ 0 & 4C_2x_2^3 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 4C_nx_n^3 \end{bmatrix}, J_yF(x,y) = I. \tag{3.40}$$

For all $(x,y), (\tilde{x},\tilde{y}) \in \Omega$ we have

$$\begin{aligned} \|J_xF(x,y) - J_xF(\tilde{x},\tilde{y})\| &= \max_{a\in\mathcal{A}}\{4C_a|x_a^3 - \tilde{x}_a^3|\} \\ &= \max_{a\in\mathcal{A}}\{4C_a|x_a - \tilde{x}_a|(x_a^2 + x_a\tilde{x}_a + \tilde{x}_a^2)\} \\ &\le 12q_1q_2^2\|x - \tilde{x}\| \le 12q_1q_2^2\|(x,y) - (\tilde{x},\tilde{y})\|. \end{aligned}$$

It follows that J_xF is Lipschitz continuous on Ω.
Obviously, J_yF is Lipschitz continuous on Ω with Lipschitz constant being 0. □

Since (3.38) takes the form of (3.1) in the case where the objective function has Lipschitz continuous gradient, it can be solved by the versions of DCA_ρ and ADCA_ρ in Section 3.3. For the problem (3.38), the functions G_ρ and H_ρ are given by

$$G_\rho(z) = \frac{\rho}{2}\|z\|^2, H_\rho(z) = \frac{\rho}{2}\|z\|^2 - [f(z) + t\theta_\gamma(z)].$$

The partial gradients of H_ρ are computed by the following formulas:

$$\nabla_x H_\rho(x,y) = \rho x - \nabla_x f(x,y) - t\left\{[J_x F(x,y)]^\top (x - v(x,y)) + F(x,y) - \gamma(x - v(x,y))\right\},$$
$$\nabla_y H_\rho(x,y) = \rho y - t(x - v(x,y)),$$

where $\nabla_x f(x,y)$ and $J_x F(x,y)$ are defined as in (3.39) and (3.40) respectively, $v(x,y) = \Pi_X\left(x - \frac{1}{\gamma}F(x,y)\right)$.
The solution to the subproblem (3.34) in this application is $\tilde{z} = (\tilde{x},\tilde{y})$, where $\tilde{x}$ (resp. $\tilde{y}$) is the projection of $\frac{\overline{x}^k}{\rho}$ (resp. $\frac{\overline{y}^k}{\rho}$) on X (resp. Y) and $\overline{x}^k = \nabla_x H_\rho(z^k), \overline{y}^k = \nabla_y H_\rho(z^k)$.
As Y is a box, $\tilde{y}$ can be explicitly computed by

$$\tilde{y}_a = \begin{cases} \overline{y}^k_a/\rho & \text{if } 0 \le \overline{y}^k_a \le \rho y^U_a, \\ 0 & \text{if } \overline{y}^k_a < 0, \\ y^U_a & \text{if } \overline{y}^k_a > \rho y^U_a. \end{cases} \tag{3.41}$$

Thus, the DCA_ρ version for solving the problem (3.38) replaces the steps 2 and 3.1 in Algorithm 3.2 by

2a. Set $\overline{z}^k = (\overline{x}^k,\overline{y}^k)$, where $\overline{x}^k = \nabla_x H_\rho(z^k), \overline{y}^k = \nabla_y H_\rho(z^k)$.

3.1a. Set $\tilde{z} = (\tilde{x},\tilde{y})$, where $\tilde{x} = \Pi_X\left(\frac{\overline{x}^k}{\rho}\right)$, $\tilde{y}$ is given by (3.41).

The ADCA_ρ version for solving the problem (3.38) replaces the steps 5 and 6.1 in Algorithm 3.3 by

5a. Set $\overline{z}^k = (\overline{x}^k,\overline{y}^k)$, where $\overline{x}^k = \nabla_x H_\rho(w^k), \overline{y}^k = \nabla_y H_\rho(w^k)$.

6.1a. Set $\tilde{z} = (\tilde{x},\tilde{y})$, where $\tilde{x} = \Pi_X\left(\frac{\overline{x}^k}{\rho}\right)$, $\tilde{y}$ is given by (3.41).

3.5 Numerical experiments

We tested the DCA_ρ and ADCA_ρ algorithms for the SBTP problem on three networks: tap-09, tap-15 and siouxfls. The first one consists of 9 nodes, 18 links and 4 Origin-Destination (OD) pairs; the second one comprises 15 nodes, 33 links and 9 OD pairs; the third one is comprised of 24 nodes, 76 links and 552 OD pairs. The detailed data including the parameters of travel time functions and the travel demands is available in MacMPEC [52], a collection of MPEC test problems. The tollable links are selected as in [40]. We first solve the following system problem (SP) and user problem (UP) to obtain the optimal flows x^{SP} and x^{UP}, respectively.

$$\begin{aligned}&(\mathrm{SP}) : \min\{s(x)^T x : x \in \hat{X}\}\\&(\mathrm{UP}) : \text{Find } x \text{ such that } s(x)^T(v - x) \geq 0, \forall v \in \hat{X}.\end{aligned}$$

Then the link a is chosen as a tollable link if x_a^{UP} is greater than x_a^{SP} by a given percentage, say $\alpha\%$. For tap-09 and tap-15, we take $\alpha = 5, 10, 20, 25$; and for the siouxfls, we take $\alpha = 5, 10, 15$.

In order to evaluate the performance of the algorithms DCA_ρ and ADCA_ρ, we compare them with DCA_MPEC and KNITRO solver [2], an advanced solver for nonlinear optimization including MPECs. To use this solver, we reformulate problem (3.38) as the MPCC below

$$\begin{aligned}\min_{(x,u^k,y,\mu^k,\gamma^k,\lambda,\omega)} \quad & s(x)^T x \qquad\qquad (3.42)\\ \text{s.t.} \quad & x \in X, 0 \leq y \leq y^U,\\ & s(x) + y + \lambda + A^\top \mu^k - \gamma^k = 0, \ k \in K,\\ & x + \omega = x^U,\\ & \gamma^k \geq 0, u^k \geq 0, (\gamma^k)^\top u^k = 0, \ k \in K,\\ & \lambda \geq 0, \omega \geq 0, \lambda^\top \omega = 0.\end{aligned}$$

For the above networks, the number of variables and complementarity constraints in the problem (3.42) are shown in the Table 3.1.

Table 3.1 – The number of variables and Complementarity Constraints (CCs)

Network	No. of variables	No. of CCs
tap-09	252	90
tap-15	861	330
siouxfls	97456	42028

Our algorithms are implemented in Matlab R2016b and run on an HP computer Intel(R) Core(TM) i5-3470 CPU, 3.2GHz, 8 Go of RAM. For tap-09, we solve the problem (3.42) by calling KNITRO (student version) in the MATLAB environment. Since the student version limits the size of problem (300 variables and 300 constraints), for tap-15 and siouxfls, KNITRO is called via NEOS server [3]. For KNITRO, the default parameters and the default initial point are used.

In DCA_ρ, ADCA_ρ and DCA_MPEC, we set $\varepsilon = 10^{-5}$. We also add the condition $|\theta_\gamma(x^k, y^k)| < \delta$ to the stopping criteria of all these algorithms. This condition assures that the feasibility error for the variational inequality constraint that is computed by

2. `https://www.artelys.com/en/optimization-tools/knitro`
3. `https://neos-server.org/neos/solvers`

$|\theta_\gamma(x,y)|$ is always less than δ. The parameters are taken as follows: $\rho^0 = 1, \delta = 10^{-5}$ for tap-09 and tap-15; $\rho^0 = 1000, \delta = 3 \times 10^{-4}$ for siouxfls; $\eta_1 = 2, \eta_2 = 1.25, q = 5$. The initial point is $z^0 = (x^0, y^0)$ with x^0 being the solution to the following linear program

$$\min\{T^\top x : x \in \hat{X}\}$$

and $y^0 = 0$.

In the experiments, we compare the value of objective function obtained by DCA_ρ, ADCA_ρ, DCA_MPEC and KNITRO as well as the running time (in seconds) of DCA_ρ, ADCA_ρ and DCA_MPEC. For tap-09, we also make a comparison between the computing time of the algorithms and that of KNITRO. For remaining networks, we do not report the computing time of KNITRO because the proposed algorithms and KNITRO run on the different environment. The computational results are reported in Table 3.2. From these results, we observe that:

- Overall, DCA_ρ and ADCA_ρ are better than KNITRO while DCA_MPEC is comparable with KNITRO. More precisely, ADCA_ρ outperforms KNITRO in 7/11 cases (tap-15 and siouxfls for all α). In the remaining cases, ADCA_ρ is comparable with KNITRO in terms of the objective value, however it is 7.24 to 11.96 times faster than KNITRO. DCA_ρ is also superior to KNITRO in 7/11 cases but it furnishes a worse solution than KNITRO on the data tap-09 with $\alpha = 5$ and $\alpha = 15$. DCA_MPEC is better than KNITRO in 3/11 cases (tap-15 except for $\alpha = 5$) and worse than KNITRO in 3/11 cases. It is noted that for siouxfls, DCA_MPEC does not find a feasible solution after 7200 seconds and KNITRO provides an infeasible solution in all cases. For tap-09, DCA_ρ (resp. DCA) is more efficient than KNITRO in 3/4 (resp. 2/4) cases in terms of computing time.
- Among the DCA based algorithms, ADCA_ρ is the best and DCA_MPEC is the worst in terms of both the objective value and computing time. In terms of objective value, ADCA_ρ gives the best result in 6/11 cases. For the remaining cases, it is comparable with or better than DCA_ρ and DCA_MPEC. In comparison with DCA_MPEC, ADCA_ρ (resp. DCA_ρ) provides a better solution in 8/11 (resp. 6/11) cases. DCA_MPEC finds a better solution than DCA_ρ on the data tap-09 ($\alpha = 5$), however in this case, DCA_ρ is 3.63 times faster than DCA_MPEC. In terms of running time, except for the data tap-09 ($\alpha = 15$), ADCA_ρ is 1.36 to 11.67 times faster than DCA_ρ. Both ADCA_ρ and DCA_ρ are faster than DCA_MPEC in all cases, the ratio of the runtime of ADCA_ρ (resp. DCA_ρ) to that of DCA_MPEC varies between 1.79 to 20.46 (resp. 1.31 and 5.96).

3.6 Conclusions

In this chapter, we studied two DCA based approaches for solving a class of mathematical programs with variational inequality constraints. We first transformed the considered problem into a DC program via a penalty technique, and then proposed a variant of DCA (DCA_ρ) and its accelerated version (ADCA_ρ) to handle this DC program. As an application, we solved the SBTP problem with fixed demands. Numerical experiments on three networks with data taken from MacMPEC show that in general, ADCA_ρ is better than DCA_ρ and both two algorithms are superior to DCA_MPEC and KNITRO.

Table 3.2 – Comparative results of the algorithms
Best results are written in bold. # denotes the number of tollable links.

			Objective value				Computing time (seconds)			
network	α	#	DCA_ρ	ADCA_ρ	DCA_MPEC	KNITRO	DCA_ρ	ADCA_ρ	DCA_MPEC	KNITRO
tap-09	5	6	2348.59	2281.74	2322.34	**2281.72**	9.75	**1.73**	35.40	17.30
	10	4	2314.77	2310.99	2314.96	**2310.19**	30.23	**2.59**	109.94	18.76
	15	3	2361.23	2344.33	2361.27	**2341.38**	**1.99**	**1.99**	3.56	16.54
	20	2	2361.18	2361.17	2361.18	**2361.16**	1.49	**0.78**	1.95	9.33
tap-15	5	12	**2850.11**	**2850.11**	**2850.11**	2957.99	3.35	**1.45**	6.1	-
	10	9	2868.07	**2867.76**	2906.41	2956.10	13.19	**5.31**	51.21	-
	20	6	2873.45	**2872.72**	3074.00	3013.54	21.96	**9.26**	63.51	-
	25	4	2915.55	**2885.53**	2920.79	3267.66	11.16	**8.18**	66.49	-
siouxfls	5	14	10052.39	**9989.41**	-	-	484.45	**201.14**	-	-
	10	6	10055.84	**10028.92**	-	-	440.64	**220.70**	-	-
	15	4	10055.35	**10032.88**	-	-	454.76	**179.44**	-	-

Chapter 4

A class of bilevel optimization problems with binary upper level variables

Abstract: *In this chapter, we consider a class of bilevel optimization problems in which the objective function is the sum of a convex function and the value function of a linear program, and the upper level variables are binary. By using an exact penalty function, we reformulate the bilevel problem as a standard DC program. An appropriate DCA is proposed to solve this DC program. Numerical experiments on a maximum flow network interdiction problem illustrate the efficiency of our algorithm.*

4.1 Introduction

A bilevel programming problem is an optimization problem in which one of its constraints is determined by another optimization problem. This problem can be stated as

$$\min_{x \in X, y \in Y} \{F(x,y) : G(x,y) \le 0, y \in \Psi(x)\}, \tag{4.1}$$

where

$$\Psi(x) = \operatorname{argmin}\{f(x,y) : y \in Y, g(x,y) \le 0\}, \tag{4.2}$$

$F, f : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}, G : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^s, g : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}^l$ are given functions, and X, Y impose additional limitations on the variables, such as upper and lower bounds or integrality requirements.
The problem (4.1) is called the upper level problem and the problem

$$\min\{f(x,y) : y \in Y, g(x,y) \le 0\}$$

1. The material of this chapter is developed from the following work:
Thi Minh Tam Nguyen, Hoai An Le Thi. A DCA Approach for a Maximum Flow Network Interdiction Problem *Submitted.*

is called the lower level problem. The variables x and y are said to be the upper and lower level variables, respectively.

Related works

The bilevel problems have drawn significant attention of the optimization community because they are hard to solve and have many applications in various fields. A survey of solution methods for these problems and their applications can be found in [93]. However, there has been little research on discrete bilevel problems, where some of the variables are integer. We are interested in approaches for solving Bilevel Problems (BP) in which the upper level variables are binary and the lower level variables are continuous. A usual approach is to replace the lower level problem with its KKT conditions and solve the resulting problem by a branch-and-bound algorithm (see, e.g., [14, 24]). When all functions in (4.1) and (4.2) are affine, the BP was reformulated as a bilevel problem with all continuous variables for which algorithms based on branch-and-bound was applied (see, e.g., [3, 69, 99]). If, in addition, $G(x, y) = 0$ and $Y = \mathbb{R}^n_+$, the BP was directly solved by a branch-and-bound algorithm [101] or a heuristic algorithm based on Tabu search [97], it was also converted into a Mixed-Integer Linear Program (MILP) and solved by a Benders decomposition algorithm [17].

Our contributions

In this chapter, we consider a special case of the problem (4.1) where $F(x, y) = \theta(x) + c^\top y$ and $f(x, y) = -c^\top y$. More specifically, we investigate the bilevel problem of the form:

$$\min\{F(x) = \theta(x) + \phi(x) : x \in \{0, 1\}^m\} \tag{4.3}$$

where $\theta : \mathbb{R}^m \to \mathbb{R}$ is a convex function, ϕ is the value function of the following linear program

$$\begin{aligned} \phi(x) = \max_{y\in\mathbb{R}^n} \quad & c^\top y \\ \text{s.t.} \quad & Ay = \xi, \\ & Bx + Dy \le d, \\ & y \ge 0, \end{aligned} \tag{4.4}$$

with $c \in \mathbb{R}^n, \xi \in \mathbb{R}^p, d \in \mathbb{R}^m$ and A, B, D being matrices of sizes $p \times n, m \times m, m \times n$, respectively. We assume throughout this chapter that for all x in $[0, 1]^m$, the feasible set of the lower level optimization problem is nonempty and the optimal value of this problem is finite.

The difficulty of the problem (4.3) resides not only in the nonconvex objective function but also in the binary variables. The special structure of this problem permits us to show that its objective function is a DC function and this bilevel problem becomes a DC program with binary variables. Furthermore, we prove that the objective function of this problem is Lipschitz, therefore using the new results concerning exact penalty in DC programming [48], we penalize the binary constraints and express the DC program with binary variables as a standard DC program for which we propose an appropriate DCA scheme, named DCA_BP (DCA for Bilevel Problem). This algorithm leads to simple subproblems which minimize a convex function on a box. Especially, we point out that when the function θ is linear, we can choose the penalty parameter as an arbitrary positive number and the sequence generated by DCA_BP is always binary. Moreover, DCA_BP in this case is very inexpensive, it consists of successively solving box constrained linear programs whose optimal solutions are explicitly computed. We apply the proposed algorithm to solve the Maximum Flow Network Interdiction (MFNI) problem studied in [89]. In this problem, an interdictor attempts to completely destroy some arcs in a capacitated directed network with the aim of minimizing both post-interdiction maximum flow and the total interdiction cost. The authors in [89] formulated the MFNI problem as a bi-objective problem and found its Pareto-optimal solutions by solving a sequence of single-objective problems using Lagrangian relaxation and a branch-and-bound algorithm. We formulate this problem in the form of (4.3) in which θ is linear. In order to evaluate the quality of solution found by DCA_BP, we use the techniques in [17] to reformulate the application problem as an MILP that can be globally solved by CPLEX software. This formulation requires choosing a sufficiently large parameter. We prove that this parameter can be chosen to equal 1.

The rest of the chapter is organized as follows. Section 4.2 discusses a solution method based on DCA for the considered problem. Section 4.3 presents how to apply the proposed method to deal with an MFNI problem and describes an MILP formulation of this problem. The numerical results are reported in Section 4.4 and Section 4.5 concludes the chapter.

4.2 Solution method

4.2.1 Exact penalty formulation for the problem (4.3)

We first prove the following proposition.

Proposition 4.1. *The objective function $F(x)$ of the problem* (4.3) *is DC and Lipschitz continuous on* $[0,1]^m$.

Proof. Consider the function $\psi(x) = -\phi(x), x \in [0,1]^m$ we have $F(x) = \theta(x) - \psi(x)$ and

$$\begin{aligned} \psi(x) = \min_{y} \quad & -c^\top y \\ \text{s.t.} \quad & Ay = \xi, \\ & Bx + Dy \leq d, \\ & y \geq 0. \end{aligned} \tag{4.5}$$

Clearly, (4.5) is a linear program and its optimal value is finite for all $x \in [0,1]^m$. According to duality theorem, $\psi(x)$ is equal to the optimal value of its Lagrange dual problem, that is, for all $x \in [0,1]^m$,

$$\begin{aligned} \psi(x) = \max_{(\lambda,\mu)} \quad & -\xi^\top \lambda + (Bx-d)^\top \mu \\ s.t. \quad & A^\top \lambda + D^\top \mu - c \geq 0, \\ & \mu \geq 0. \end{aligned} \tag{4.6}$$

Let P be the polyhedron defined by

$$P = \left\{ (\lambda,\mu) : A^\top \lambda + D^\top \mu - c \geq 0, \mu \geq 0 \right\},$$

and

$$f(x,\lambda,\mu) = -\xi^\top \lambda + (Bx-d)^\top \mu, \tag{4.7}$$

we have

$$\psi(x) = \max\{f(x,\lambda,\mu) : (\lambda,\mu) \in P\}.$$

Since $f(x,\lambda,\mu)$ is convex in x, $\psi(x)$ is convex, and consequently $F(x)$ is a DC function on $[0,1]^m$.

Let V be the set of vertices of the polyhedron P and let

$$\overline{V} = \{\mu \in \mathbb{R}^m : \exists \lambda \in \mathbb{R}^p, (\lambda,\mu) \in V\}. \tag{4.8}$$

It is easy to see that $V, \overline{V}$ are nonempty finite sets and

$$\psi(x) = \max\{f(x,\lambda,\mu) : (\lambda,\mu) \in V\}. \tag{4.9}$$

Take arbitrary $x^1, x^2 \in [0,1]^m$, then there exist $(\lambda^1,\mu^1), (\lambda^2,\mu^2) \in V$ such that

$$\psi(x^1) = f(x^1,\lambda^1,\mu^1), \psi(x^2) = f(x^2,\lambda^2,\mu^2).$$

We have

$$\begin{aligned} &\psi(x^1) - \psi(x^2) = f(x^1,\lambda^1,\mu^1) - \max\{f(x^2,\lambda,\mu) : (\lambda,\mu) \in V\} \\ &\leq f(x^1,\lambda^1,\mu^1) - f(x^2,\lambda^1,\mu^1) = (Bx^1-d)^\top \mu^1 - (Bx^2-d)^\top \mu^1 \\ &= (\mu^1)^\top (B(x^1-x^2)) \leq \|B^\top \mu^1\| \|x^1-x^2\|. \end{aligned}$$

It follows that

$$\psi(x^1) - \psi(x^2) \leq K\|x^1-x^2\|, \text{where } K = \max\{\|B^\top \mu\| : \mu \in \overline{V}\} < \infty.$$

By changing the role of x^1 and x^2, we get

$$\psi(x^2) - \psi(x^1) \leq K\|x^2-x^1\|.$$

Therefore

$$|\psi(x^1) - \psi(x^2)| \leq K\|x^1-x^2\|, \ \forall x^1, x^2 \in [0,1]^m,$$

which is equivalent to

$$|\phi(x^1) - \phi(x^2)| \leq K\|x^1-x^2\|, \ \forall x^1, x^2 \in [0,1]^m.$$

This means that $\phi(x)$ is Lipschitz continuous on $[0,1]^m$.
Since $\theta : \mathbb{R}^m \to \mathbb{R}$ is convex, it is Lipschitz continuous on $[0,1]^m$, and so is the function $F(x)$. □

Let $p(x)$ be the penalty function given by

$$p(x) = \frac{1}{2}\sum_{i=1}^{m} x_i(1-x_i).$$

Obviously, p is concave and nonnegative on $[0,1]^m$. Thus, the problem (4.3) can be rewritten as follows

$$\min\{\theta(x) - \psi(x) : \; x \in [0,1]^m, p(x) \leq 0\}. \tag{4.10}$$

We consider the penalized problem:

$$\min\{F_t(x) = \theta(x) - \psi(x) + tp(x) : \; x \in [0,1]^m\} \tag{4.11}$$

By the results concerning exact penalty in DC programming presented in [48], we have the following proposition.

Proposition 4.2. *There exists a nonnegative number t_0 such that for all $t > t_0$, the problems* (4.10) *and* (4.11) *are equivalent in the sense that they have the same optimal value and the same optimal solution set. Moreover, when $\theta(x)$ is linear, we can take $t_0 = 0$.*

The second assertion of this proposition follows from the fact that the objective function is concave and continuous on $[0,1]^m$, and the set of vertice of $[0,1]^m$ is equal to the set $\{x \in [0,1]^m : p(x) \leq 0\}$.

4.2.2 Solving the penalized problem by DCA

In what follows, we use the notation $\psi(x), V, \overline{V}, f(x,\lambda,\mu)$ as in the proof of Proposition 4.1. Let

$$\begin{aligned} I(x) &= \{(\lambda,\mu) \in V : \psi(x) = f(x,\lambda,\mu)\}, \\ \overline{I}(x) &= \{\mu \in \mathbb{R}^m : \exists \lambda \in \mathbb{R}^p, (\lambda,\mu) \in I(x)\}. \end{aligned}$$

By (4.7) and (4.9), one has:

$$\partial\psi(x) = \operatorname{conv}\left\{\nabla_x f(x,\lambda,\mu) : (\lambda,\mu) \in I(x)\right\} = \operatorname{conv}\left\{B^\top \mu : \mu \in \overline{I}(x)\right\}, \forall x \in [0,1]^m,$$

where the second equality follows from $\nabla_x f(x,\lambda,\mu) = B^\top \mu$.

The problem (4.11) can be reformulated as the standard DC program below:

$$\min\{G(x) - H(x) : x \in [0,1]^m\}, \tag{4.12}$$

where

$$G(x) = \theta(x), H(x) = \psi(x) - tp(x).$$

The DCA applied to (4.12) consists of first computing a subgradient

$$\overline{x}^k \in \partial H(x^k),$$

and then solving the following convex problem at each iteration

$$\min\{\theta(x) - (\overline{x}^k)^\top x : x \in [0,1]^m\}. \tag{4.13}$$

Noting that $\partial H(x) = \partial\psi(x) + t\partial(-p)(x)$, therefore a subgradient $\overline{x} \in \partial H(x)$ can be computed as follows

$$\overline{x} = B^\top \mu + t\left(x - \frac{1}{2}\right), \text{ where } \mu \in \overline{I}(x). \tag{4.14}$$

The algorithm is summarized as follows.

Algorithm 4.1 DCA_BP

```
Initialization: Choose a point $x^0 \in [0,1]^m$ and a sufficiently small positive number $\varepsilon$.
Set $k := 0$.
    1. Compute $\overline{x}^k \in \partial H(x^k)$ using (4.14).
    2. Compute $x^{k+1}$ as a solution to the convex problem (4.13).
    3. If $\|x^{k+1} - x^k\| < \varepsilon(\|x^k\| + 1)$ then stop, the computed solution is $x^{k+1}$. Otherwise,
       set $k := k + 1$ and go to Step 1.
```

The convergence properties of DCA_BP is shown in following theorem.

Theorem 4.1. *Suppose that $\{x^k\}$ is the sequence generated by DCA_BP. Then the following statements hold:*

(i) *The sequence $\{F_t(x^k)\}$ is decreasing.*

(ii) *Every limit point of the sequence $\{x^k\}$ is a critical point of the problem* (4.12).

(iii) *In the case where $\theta(x)$ is linear, $x^k \in \{0,1\}^m$ $\forall k \geq 1$.*

Proof. Since $F_t(x)$ is continuous on $[0,1]^m$ and $[0,1]^m$ is compact, the optimal value of the problem (4.12) is finite and the sequences $\{x^k\}, \{\overline{x}^k\}$ are bounded. As a consequence of Theorem 1.2 presented in Chapter 1, (i) and (ii) hold.
Suppose that $\theta(x) = q^\top x$, the subproblem (4.13) becomes

$$\min\{(q - \overline{x}^k)^\top x : x \in [0,1]^m\}.$$

whose solution is computed by

$$x_i^{k+1} = \begin{cases} 0 & \text{if } q_i - \overline{x}_i^k \geq 0, \\ 1 & \text{if } q_i - \overline{x}_i^k < 0. \end{cases}$$

This means (iii) holds. □

4.3 Application to a maximum flow network interdiction problem

4.3.1 Related works

The Maximum Flow Network Interdiction (MFNI) problems have been studied since 1960s with important applications in the military and security fields. In these problems, a network user attempts to maximize the amount of flow through a capacitated network while an interdictor tries to minimize this maximum flow by breaking some arcs in the network using available resources [103]. The MFNI problems can be considered in different contexts in which the interdictor partially or fully destroys arcs, the network has a source node and a sink node or has multiple sources and sinks. Some works related to MFNI problems can be found in [23, 70, 102, 103]. There are two approaches proposed in these works. The first one uses the dual network that is only defined when the primal network is planar. The second one uses a branch-and-bound algorithm that can be computationally expensive. Continuing these studies, Burch et al. [9] suggested an algorithm based on decomposition to approximate the optimal solution of an MFNI problem. More recently, Altner et al. [13] presented two classes of valid inequalities for the MFNI problem using integer linear programming formulation of Wood [103]. In all the problems mentioned above, the interdictor was only interested in minimizing the post-interdiction maximum flow. However, in practice, he/she can want to minimize both this maximum flow and the total interdiction cost. Such a MFNI problem was investigated in [89]. They formulated this problem as a bi-objective problem and found its Pareto-optimal solutions by solving a sequence of single-objective problems using Lagrangian relaxation and a branch-and-bound algorithm. Besides, some extensions of MFNI problems have been also studied such as considering a MFNI on a multicommodity flow network [2, 57] or considering the distance between an arc and an attack location [95] (in this case, the attack location may be a some point near the network).

In this work, we consider the MFNI problem studied in [89] in which an interdictor attempts to completely destroy some arcs in a capacitated directed network with the aim of minimizing both post-interdiction maximum flow and the total cost for interdiction. However, we formulate it in the form of the problem (4.3). This formulation allows us to balance the maximum flow and interdiction cost according to the choice of a parameter α.

4.3.2 Problem formulation

Consider a capacitated directed network with the set of nodes $\mathcal{N}$, the set of arcs $\mathcal{A}$ and two special nodes: the source node s and the sink node t. The following notation is used to formulate the problem.

- $\mathcal{A}_{\text{i-}}$ is the set of arcs in $\mathcal{A}$ with i as the start node.

- $\mathcal{A}_{\text{-i}}$ is the set of arcs in $\mathcal{A}$ with i as the end node.
- $u = (u_a)_{a\in\mathcal{A}}, u_a$ is the capacity of the arc a.
- $x = (x_a)_{a\in\mathcal{A}}, x_a = 1$ if the arc a is interdicted and $x_a = 0$ otherwise.
- $r = (r_a)_{a\in\mathcal{A}}, r_a$ is the interdiction cost for the arc a.
- $y = (y_a)_{a\in\mathcal{A}}, y_a$ is the amount of flow on the arc a.
- f is the total amount of flow from the node s to the node t.

After being interdicted, the capacity of the arc a is $(1-x_a)u_a$, therefore the maximum flow problem [18] becomes

$$\begin{aligned} \phi(x) = \max_{y,f} \quad & f \\ \text{s.t.} \quad & \sum_{a\in\mathcal{A}_{i-}} y_a - \sum_{a\in\mathcal{A}_{-i}} y_a = \begin{cases} 0 & \text{if } i \in \mathcal{N}\setminus\{s,t\} \\ f & \text{if } i = s \\ -f & \text{if } i = t \end{cases} \\ & 0 \le y_a \le (1-x_a)u_a,\ a \in \mathcal{A}. \end{aligned} \tag{4.15}$$

Since the total interdiction cost is $\sum_{a\in\mathcal{A}} r_a x_a$, the considered MFNI problem can be formulated as follows

$$\min_{x\in\{0,1\}^n} \alpha \sum_{a\in\mathcal{A}} r_a x_a + \phi(x), \tag{4.16}$$

where n is the number of arcs in the network, α is a positive parameter that controls the balance between the total cost of interdiction and the maximum flow.
The problem (4.16) can be reformulated as

$$\min_{x\in\{0,1\}^n} \alpha r^\top x + \phi(x), \tag{4.17}$$

with

$$\begin{aligned} \phi(x) = \max_{y\in\mathbb{R}^n} \quad & \left(\sum_{a\in\mathcal{A}_{s-}} y_a - \sum_{a\in\mathcal{A}_{-s}} y_a \right) \\ \text{s.t.} \quad & \sum_{a\in\mathcal{A}_{i-}} y_a - \sum_{a\in\mathcal{A}_{-i}} y_a = 0, i \in \mathcal{N}\setminus\{s,t\}, \\ & \sum_{a\in\mathcal{A}_{s-}} y_a - \sum_{a\in\mathcal{A}_{-s}} y_a + \sum_{a\in\mathcal{A}_{t-}} y_a - \sum_{a\in\mathcal{A}_{-t}} y_a = 0, \\ & 0 \le y_a \le u_a(1-x_a), a \in \mathcal{A}. \end{aligned}$$

Clearly, the problem (4.17) takes the form of (4.3) in which $\theta(x) = \alpha r^\top x, c$ is the transpose of the s^{th} row of the node-arc incidence matrix M, $p = \overline{p} - 1$($\overline{p}$ is the number of nodes in the network), $\xi = 0, d = u, A$ is the matrix obtained from M by deleting the s^{th} and t^{th} rows and adding a new row that is the sum of two deleted rows to the bottom of the resulting matrix, B is the diagonal matrix whose main diagonal consists of the elements of vector u, and D is the identity matrix of order n. Therefore, the problem (4.17) can be solved by DCA_BP. Note that since the function $\theta(x)$ in (4.17) is linear, the solution found by DCA_BP is always binary. For the problem (4.17), the DC components of the objective function are

$$G(x) = \alpha r^\top x, H(x) = -\phi(x) - tp(x)$$

A subgradient $\overline{x}^k \in \partial H(x^k)$ is computed by using (4.14). The subproblem at iteration k is

$$\min\{(\alpha r - \overline{x}^k)^\top x : x \in [0,1]^n\}.$$

whose solution can be computed by

$$x_i^{k+1} = \begin{cases} 0 & \text{if } \alpha r_i - \overline{x}_i^k \geq 0, \\ 1 & \text{if } \alpha r_i - \overline{x}_i^k < 0. \end{cases}$$

In the next subsection, we present an MILP formulation of the problem (4.17) based on the techniques in [17].

4.3.3 MILP formulation of (4.17)

As mentioned above, the problem (4.17) takes the form

$$\min\{\alpha r^\top x + \phi(x) : x \in \{0,1\}^n\} \tag{4.18}$$

where

$$\phi(x) = \max_{y \in \mathbb{R}^n}\{c^\top y : Ay = \xi, Bx + Dy \leq d, y \geq 0\}$$

It is straightforward to see that the problem (4.18) is equivalent to the problem (4.19) below in the sense that x^* is a solution to (4.18) if and only if there exists y^* such that (x^*, y^*) is a solution to (4.19).

$$\begin{aligned} \min_{x,y} \quad & \alpha r^\top x + c^\top y & (4.19) \\ \text{s.t.} \quad & x \in \{0,1\}^n, \\ & y \in \operatorname{argmin}\{-c^\top y : Ay = \xi, Bx + Dy \leq d, y \geq 0\}. \end{aligned}$$

Using the KKT optimality conditions for the lower level optimization problem, the problem (4.19) can be reformulated as the MPCC problem below

$$\begin{aligned} \min_{x,y,\lambda,\mu} \quad & \alpha r^\top x + c^\top y & (4.20) \\ \text{s.t.} \quad & x \in \{0,1\}^n, & (4.21) \\ & Ay = \xi, & (4.22) \\ & Bx + Dy \leq d, & (4.23) \\ & A^\top \lambda + D^\top \mu - c \geq 0, & (4.24) \\ & y \geq 0, \mu \geq 0, & (4.25) \\ & \mu^\top (d - Bx - Dy) = 0, & (4.26) \\ & y^\top (-c + A^\top \lambda + D^\top \mu) = 0. & (4.27) \end{aligned}$$

The constraints (4.26), (4.27) can be replaced with the constraint

$$-c^\top y \leq -\lambda^\top \xi + \mu^\top (Bx - d) \tag{4.28}$$

Indeed, suppose that (4.26) and (4.27) hold, we have

$$-c^\top y + \lambda^\top Ay + \mu^\top Dy = 0.$$

This implies that

$$-c^\top y = -\lambda^\top \xi + \mu^\top (Bx - d),$$

and consequently (4.28) holds.

Conversely, assume that (4.28) holds. It follows from (4.24) and (4.25) that

$$-c^\top y \geq -(\lambda^\top Ay + \mu^\top Dy). \tag{4.29}$$

This inequality combined with (4.28) leads to the inequality:

$$-(\lambda^\top Ay + \mu^\top Dy) \leq -\lambda^\top \xi + \mu^\top (Bx - d),$$

which is equivalent to

$$\mu^\top (d - Bx - Dy) \leq 0. \tag{4.30}$$

Moreover, by (4.23) and (4.25), the left side of (4.30) is greater than or equal to 0. Therefore, equality in (4.30) holds and hence equality in (4.29) also holds. That means (4.26) and (4.27) hold.

Clearly, (4.28) can be rewritten as follows

$$-c^\top y \leq -\xi^\top \lambda + \sum_{i=1}^{n} \mu_i u_i (x_i - 1). \tag{4.31}$$

The inequality (4.31) can be replaced by the following linear constraints with U being a sufficiently large constant.

$$\begin{align}
& -c^\top y \leq -\xi^\top \lambda + \sum_{i=1}^{n} u_i z_i, \tag{4.32}\\
& z_i \leq -\mu_i + U x_i, \ i = 1, ..., n, \tag{4.33}\\
& z_i \geq -\mu_i, \ i = 1, ..., n, \tag{4.34}\\
& z_i \geq -U(1 - x_i), \ i = 1, ..., n, \tag{4.35}\\
& z_i \leq 0, \ i = 1, ..., n \tag{4.36}
\end{align}$$

Thus, the problem (4.17) can be reformulated as the next mixed-integer linear program

$$\begin{align}
\min_{x,y,\lambda,\mu,z} \quad & \alpha r^\top x + c^\top y \tag{4.37}\\
\text{s.t.} \quad & (4.21) - (4.25), (4.32) - (4.36).
\end{align}$$

Remark 4.1. *It is worth noting that* (λ, μ) *is a solution to the problem* (4.6), *which is the dual of the lower level problem. Therefore, if* $(x^*, y^*, \lambda^*, \mu^*)$ *is a solution to the problem* (4.20) *then there exist a vertex* $(\widehat{\lambda}, \widehat{\mu})$ *of the polyhedron* $P = \big\{(\lambda, \mu) : A^\top \lambda + D^\top \mu - c \geq 0, \mu \geq 0\big\}$ *such that* $(x^*, y^*, \widehat{\lambda}, \widehat{\mu})$ *is also a solution to* (4.20).

The following proposition shows that if $(\widehat{\lambda}, \widehat{\mu})$ is a vertex of P then $\widehat{\mu}_i \leq 1\ \forall i$, and consequently we can choose $U = 1$.

Proposition 4.3. *if $(\widehat{\lambda}, \widehat{\mu})$ is a vertex of P then $\widehat{\mu}_i \leq 1\ \forall i$.*

Proof. Consider the equation system $E\gamma = b$, where

$$E = \begin{bmatrix} A^\top & I_n \\ 0 & I_n \end{bmatrix},\ \gamma = \begin{bmatrix} \lambda \\ \mu \end{bmatrix},\ b = \begin{bmatrix} c \\ 0 \end{bmatrix},$$

and I_n is the identity matrix of order n.
Since each vertex of P is determined by $p + n$ linearly independent equations from the above equation system, there exist $\overline{E}$ is a nonsingular submatrix of E and $\bar{b}$ is a subvector of b such that $(\widehat{\lambda}, \widehat{\mu})$ is a solution to the system $\overline{E}\gamma = \bar{b}$.
By the Cramer's rule, we have

$$\widehat{\mu}_i = \frac{\det(\overline{E}_{p+i})}{\det(\overline{E})},\ i = 1, ..., n,$$

here $\overline{E}_{p+i}$ is the matrix formed by replacing the $(p + i)^{\text{th}}$ column of $\overline{E}$ by $\bar{b}$.
We will prove that $[E\ b]$ is a totally unimodular matrix, i.e. the determinant of each square submatrix of $[E\ b]$ is equal to 0, 1, or -1.
As the matrix A (resp. M) has exactly two nonzero elements 1 and -1 in each column, according to Proposition 2.6 in [74], it is a totally unimodular matrix.
Let $\overline{A}$ be the matrix of size $(p + 1) \times n$ defined by

$$\overline{A} = \begin{bmatrix} A \\ c^\top \end{bmatrix}$$

Then the p^{th} row of $\overline{A}$ is the sum of the s^{th} and t^{th} rows of M and the $(p + 1)^{\text{th}}$ row of $\overline{A}$ is the s^{th} row of M. We will show that $\overline{A}$ is also a totally unimodular matrix.
Indeed, suppose that $\overline{B}$ is a square submatrix of $\overline{A}$.

- If the p^{th} row of $\overline{A}$ does not appear in $\overline{B}$ then $\overline{B}$ is a submatrix of M, that is $\det(\overline{B}) \in \{0, 1, -1\}$.
- If the $(p + 1)^{\text{th}}$ row of $\overline{A}$ does not appear in $\overline{B}$ then $\overline{B}$ is a submatrix of A, and hence $\det(\overline{B}) \in \{0, 1, -1\}$.
- If the last two rows of $\overline{A}$ appear in B then

$$\det(\overline{B}) = \det(\overline{B}_1) + \det(\overline{B}_2),$$

where $\overline{B}_1$ (resp. $\overline{B}_2$) is the matrix obtained from B by replacing the row corresponding to the p^{th} row of $\overline{A}$ with the row corresponding to the s^{th} row (resp. the t^{th} row) of M. Moreover, $\det(\overline{B}_1) = 0$ because the last two rows of $\overline{B}_1$ are identical, $\det(\overline{B}_2) \in \{0, 1, -1\}$ as $\overline{B}_2$ is a submatrix of M. It follows that $\det(\overline{B}) \in \{0, 1, -1\}$.

Therefore $\overline{A}$ is a totally unimodular matrix. This implies that $\overline{A}^\top$ and consequently $[E\ b]$ is also a totally unimodular matrix. Since $\overline{E}_{p+i}$ and $\overline{E}$ are square submatrix of $[E\ b]$ and $\overline{E}$ is nonsingular, we have $\det(\overline{E}) \in \{1, -1\}, \det(\overline{E}_{p+i}) \in \{0, 1, -1\}$. As a result, $\widehat{\mu}_i \leq 1\ \forall i$. □

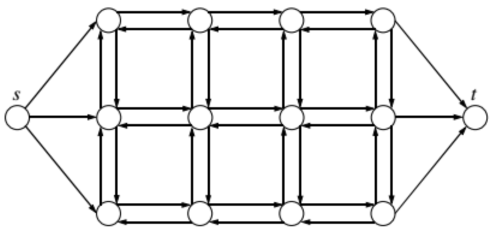

Figure 4.1 – Rectangular grid network with $n_1 = 3, n_2 = 4$ [89]

4.4 Numerical results

The proposed algorithm was tested on some rectangular grid networks as in [89]. Figure 4.1 illustrates a network of this kind. Each network consists of $m = n_1n_2 + 2$ nodes and $n = 2n_2(2n_1 - 1)$ arcs where n_1, n_2 are the number of rows and columns of nodes respectively. Arcs connecting the source node or sink node to other nodes have infinite capacities and are impossible to be interdicted. The capacities of the other arcs are randomly drawn from the discrete uniform distribution on $[1, 49]$. We also consider three variants of these networks as in [89]. In the first one (A1), the interdiction cost for each arc is 1, in the second one (A2), the interdiction cost for each arc is 1 or 2 and in the third one (A3), the interdiction cost for each arc is 1, 2 or 3. For each variant, we generate 10 networks with different sizes. We take $\alpha = 1$.

To evaluate the performance of the DCA_BP, we compare it with CPLEX software. Our algorithm is implemented in Matlab R2016b and run on an HP computer Intel(R) Core(TM) i5-3470 CPU, 3.2GHz, 8 Go of RAM. The problem (4.37) is solved by calling CPLEX 12.6 in the MATLAB environment. In DCA_BP, we take $\varepsilon = 10^{-6}$, the penalty parameter $t = 1$ and start DCA_BP from the point x^0 where $x^0_a = 1,\ \forall a \in \mathcal{A}$. For CPLEX, the default parameters and the default initial point are used.

In the experiments, we compare the value of objective function obtained by DCA_BP and CPLEX as well as their computing time (in seconds). The computational results are reported in the tables 4.1, 4.2 and 4.3. From these results, we observe that:

- In terms of the quality of solutions, DCA_BP provides a global solution in all cases except for data 10 in the networks A3. In that case, the relative difference between the objective function value obtained by DCA_BP and the optimal value is small (3.3%).
- In terms of computing time, DCA_BP is faster than CPLEX in all cases. For the medium and large networks (data 6-10), the ratio of the runtime of DCA_BP to that of CPLEX ranges from 4.45 to 13.48. This ratio varies between 1.3 and 8.86 for the remaining networks.

Table 4.1 – Comparative results of DCA_BP and CPLEX on the networks A1

					Objective value		Computing time	
data	n_1	n_2	m	n	DCA_BP	CPLEX	DCA_BP	CPLEX
1	3	4	14	40	3	3	0.0154	0.0200
2	4	8	34	112	4	4	0.0080	0.0262
3	5	5	27	90	5	5	0.0118	0.0243
4	6	10	62	220	6	6	0.0076	0.0375
5	8	12	98	360	8	8	0.0150	0.0437
6	10	20	202	760	10	10	0.0123	0.1187
7	15	25	377	1450	15	15	0.0282	0.3802
8	20	40	802	3120	20	20	0.1343	1.0173
9	30	60	1820	7080	30	30	0.5037	4.5644
10	40	80	3202	12640	40	40	1.3753	9.0624

Table 4.2 – Comparative results of DCA_BP and CPLEX on the networks A2

					Objective value		Computing time	
data	n_1	n_2	m	n	DCA_BP	CPLEX	DCA_BP	CPLEX
1	3	4	14	40	6	6	0.0114	0.0215
2	4	8	34	112	8	8	0.0083	0.0203
3	5	5	27	90	10	10	0.0086	0.0190
4	6	10	62	220	11	11	0.0067	0.0591
5	8	12	98	360	15	15	0.0081	0.0513
6	10	20	202	760	19	19	0.0145	0.1455
7	15	25	377	1450	29	29	0.0357	0.4634
8	20	40	802	3120	38	38	0.1429	1.2800
9	30	60	1820	7080	58	58	0.5700	3.6733
10	40	80	3202	12640	77	77	1.6043	10.3670

4.5 Conclusions

We investigated a class of bilevel problems in which the objective function is the sum of a convex function and the value function of a linear program and the upper level variables are binary. We first converted the bilevel problem into a standard DC program via an exact penalty function and then designed a suitable DCA scheme to solve the resulting problem. As an application, we considered an MFNI problem. Numerical experiments on several randomly generated networks indicate that our method solves this problem quite efficiently, it usually provides a global solution in a short time.

Table 4.3 – Comparative results of DCA_BP and CPLEX on the networks A3

					Objective value		Computing time	
data	n_1	n_2	m	n	DCA_BP	CPLEX	DCA_BP	CPLEX
1	3	4	14	40	7	7	0.0094	0.0213
2	4	8	34	112	9	9	0.0073	0.0231
3	5	5	27	90	10	10	0.0094	0.0209
4	6	10	62	220	13	13	0.0075	0.0344
5	8	12	98	360	18	18	0.0080	0.0461
6	10	20	202	760	21	21	0.0155	0.1437
7	15	25	377	1450	33	33	0.0385	0.3905
8	20	40	802	3120	45	45	0.1518	1.3510
9	30	60	1820	7080	66	66	0.5961	3.5409
10	40	80	3202	12640	91	88	1.7269	7.6849

Chapter 5

Continuous equilibrium network design problem

Abstract: *In this chapter, we consider one of the most challenging problems in transportation, namely the Continuous Equilibrium Network Design Problem (CENDP). This problem is to determine capacity expansions of existing links in order to minimize the total travel cost plus the investment cost for link capacity improvements, when the link flows are constrained to be in equilibrium. We use the MPCC model for the CENDP and recast it as a general DC via the use of a penalty technique. A DCA scheme is developed to solve the resulting problem. Numerical results indicate the efficiency of our method vis-à-vis some existing algorithms.*

5.1 Introduction

The network design problem is one of the critical problems in transportation due to increasing demand for travel on roads. The purpose of this problem is to select location to build new links or to determine capacity improvements of existing links so as to optimize transportation network in some sense. A network design problem is said to be continuous if it deals with divisible capacity expansions (expressed by continuous variables). The CENDP consists of determining capacity enhancement of existing links to minimize the sum of the total travel cost and the expenditure for link capacity improvement, when the link flows are restricted to be in equilibrium. The term "equilibrium" in this problem refers to the deterministic user equilibrium which is defined that for each origin-destination pair, at equilibrium, the travel costs on all utilized paths equal and do not exceed the travel cost on any unused path. The CENDP is generally formulated as a bi-level program or mathematical program with equilibrium constraints. In general, solving these problems is intractable because of the nonconvexity of both the objective function and feasible region.

1. This chapter is published under the title:
Thi Minh Tam Nguyen, Hoai An Le Thi: A DC Programming Approach to the Continuous Equilibrium Network Design Problem. In "Advanced Computational Methods for Knowledge Engineering", Advances in Intelligent Systems and Computing, Volume 453, pp.3-16, Springer 2016.

Related works

Abdulaal and LeBlanc [1] are presumed to be pioneers studying the CENDP. They stated this problem as a bi-level program and transformed it into an unconstrained problem which was solved by the Hooke-Jeeves' algorithm. To date, a number of approaches have been proposed to address this CENDP, for example the equilibrium decomposed optimization heuristics [96], sensitivity analysis based heuristic methods [20], simulated annealing approach [19], augmented Lagrangian method [71], gradient-based approaches [11]. More recently, David Z.W. Wang et al. [100] suggested a method for finding the global solution to the linearized CENDP. These authors formulated the CENDP as an MPCC and converted complementarity constraints into the mixed-integer linear constraints. In addition, the travel cost functions were linearized by introducing binary variables. As a result, the MPCC became a mixed-integer linear program that was solved by using optimization software package CPLEX. However, their method for solving the MPCC produces many new variables including a considerable number of the binary variables and the mixed-integer linear program itself is a hard problem.

Our contributions

This chapter aims to give a new approach based on DC programming and DCA to solve the MPCC formulation for the CENDP. The difficulty of this MPCC problem resides in the complementarity constraints and the non-convex travel cost functions. By introducing new variables and using a penalty technique, we reformulate the MPCC problem as a general DC program for which an appropriate DCA, named DCA_CENDP, is developed.

The rest of the chapter is organized as follows. In section 5.2, we present the MPCC formulation of the CENDP. Section 5.3 discusses a solution method for the CENDP. The numerical results are reported in section 5.4 and some conclusions are given in section 5.5.

5.2 Problem formulation

The following notation is used throughout this paper.

A	the set of links in the network.
W	the set of origin-destination (OD) pairs.
R_w	the set of paths connecting the OD pair $w \in W$.
q_w	the fixed travel demand for OD pair w.
δ_{ap}^w	the indicator variables, $\delta_{ap}^w = 1$ if link a is on path p between OD pair w, $\delta_{ap}^w = 0$ otherwise.
f_p^w	the flow on path p connecting OD pair w, $f = [f_p^w]$.
x_a	the flow on link a, $x = [x_a]$,

$$x_a = \sum_{w \in W} \sum_{p \in R_w} \delta_{ap}^w f_p^w .$$

y_a	the capacity of link a after expansion, $y = [y_a]$.
$\underline{y_a}$	the capacity of link a before expansion, $\underline{y} = [\underline{y_a}]$.
$\overline{y_a}$	the upper bound of y_a.
π_w	the minimum travel cost between OD pair w.
$g_a(y_a)$	the improvement cost for link a.
θ	the relative weight of improvement costs and travel costs.
$t_a(x_a,y_a)$	the travel cost on link a,

$$t_a(x_a, y_a) = A_a + B_a \left(\frac{x_a}{y_a} \right)^4 .$$

c_p^w	the travel cost on path p between OD pair w, $c = [c_p^w]$,

$$c_p^w = \sum_{a \in A} \delta_{ap}^w t_a(x_a,y_a).$$

In this chapter, we assume that g_a is convex.

As mentioned in [100], the CENDP can be formulated as the following MPCC:

$$\min_{x,y,f,c,\pi} \quad \sum_{a \in A} t_a(x_a,y_a) x_a + \theta \sum_{a \in A} g_a(y_a) \tag{5.1}$$

subject to:

(i) Demand conservation and capacity expansion constraints:

$$\sum_{p \in R_w} f_p^w = q_w, \; w \in W, \tag{5.2}$$

$$\underline{y_a} \le y_a \le \overline{y_a}, \; a \in A. \tag{5.3}$$

(ii) Deterministic user equilibrium constraints:

$$f_p^w (c_p^w - \pi_w) = 0, \; p \in R_w, w \in W, \tag{5.4}$$

$$c_p^w - \pi_w \ge 0, \; p \in R_w, w \in W. \tag{5.5}$$

(iii) Definitional constraints:

$$x_a = \sum_{w\in W}\sum_{p\in R_w}\delta_{ap}^w f_p^w,\ x_a \geq 0,\ a\in A, \tag{5.6}$$

$$c_p^w = \sum_{a\in A}\delta_{ap}^w\left[A_a + B_a\left(\frac{x_a}{y_a}\right)^4\right],\ p\in R_w, w\in W, \tag{5.7}$$

$$f_p^w \geq 0,\ p\in R_w, w\in W. \tag{5.8}$$

5.3 Solution method by DC programming and DCA

It is worth noting that [100]

$$\begin{aligned}\sum_{a\in A} t_a(x_a,y_a)x_a = & \sum_{a\in A} t_a(x_a,y_a)\sum_{w\in W}\sum_{p\in R_w}\delta_{ap}^w f_p^w = \sum_{w\in W}\sum_{p\in R_w}\sum_{a\in A} t_a(x_a,y_a)\delta_{ap}^w f_p^w \\ = & \sum_{w\in W}\sum_{p\in R_w} c_p^w f_p^w = \sum_{w\in W}\sum_{p\in R_w} f_p^w\pi_w = \sum_{w\in W} q_w\pi_w.\end{aligned}$$

Therefore, the objective function of the problem (5.1) is equal to

$$\sum_{w\in W} q_w\pi_w + \theta\sum_{a\in A} g_a(y_a)$$

which is convex. However, the problem (5.1) is still a difficult problem due to the nonconvexity of the feasible region which stems from complementarity constraints and the non-convex travel cost functions. To handle this problem, the complementarity constraints

$$\left\{f_p^w(c_p^w - \pi_w) = 0, f_p^w \geq 0, c_p^w - \pi_w \geq 0\right\}$$

are replaced by

$$\left\{\min(f_p^w, c_p^w - \pi_w) \leq 0, f_p^w \geq 0, c_p^w - \pi_w \geq 0\right\}.$$

Besides, for $a\in A$, the new variables $u_a = \frac{x_a^2}{y_a^2}$ are introduced to lessen the level of complexity of the constraints (5.7).

The problem (5.1) can be rewritten as follows:

$$\begin{aligned}\min_{x,y,f,c,\pi,u}\quad & \sum_{w\in W} q_w\pi_w + \theta\sum_{a\in A} g_a(y_a) \\ \text{s.t.}\quad & (5.2), (5.3), (5.5), (5.6), (5.8), \\ & \min(f_p^w, c_p^w - \pi_w) \leq 0,\ p\in R_w, w\in W, \\ & c_p^w - \sum_{a\in A}\delta_{ap}^w(A_a + B_a u_a^2) = 0,\ p\in R_w, w\in W, \\ & \frac{x_a^2}{y_a} - u_a y_a = 0,\ a\in A.\end{aligned}$$

This problem is equivalent to the following problem:

$$\begin{aligned}
\min_{x,y,f,c,\pi,u,v} \quad & \sum_{w\in W} q_w \pi_w + \theta \sum_{a\in A} g_a(y_a) && (5.9)\\
\text{s.t.} \quad & (5.2), (5.3), (5.5), (5.6), (5.8), \\
& \min(f_p^w, c_p^w - \pi_w) \le 0,\ p \in R_w, w \in W, && (5.10)\\
& c_p^w - \sum_{a\in A} \delta_{ap}^w (A_a + B_a u_a^2) \le 0,\ p \in R_w, w \in W, && (5.11)\\
& \sum_{a\in A} \delta_{ap}^w (A_a + B_a u_a^2) - c_p^w \le 0,\ p \in R_w, w \in W, && (5.12)\\
& \frac{x_a^2}{y_a} - v_a \le 0,\ a \in A, && (5.13)\\
& v_a - u_a y_a \le 0,\ a \in A, && (5.14)\\
& u_a y_a - \frac{x_a^2}{y_a} \le 0,\ a \in A. && (5.15)
\end{aligned}$$

The problem (5.9) can be solved by transforming the constraints (5.10), (5.11) and (5.13)-(5.15) into DC constraints and developing a DCA for the resulting problem. However, when the total number of paths is large this method produces many DC constraints. To diminish the number of these DC constraints, the constraints (5.10), (5.11) of the problem (5.9) are penalized and we obtain the problem:

$$\begin{aligned}
\min \quad & F(X) = \sum_{w\in W} q_w \pi_w + \theta \sum_{a\in A} g_a(y_a) + t_1 F_1(X) + t_2 F_2(X) && (5.16)\\
\text{s.t.} \quad & X \in P, \\
& \sum_{a\in A} \delta_{ap}^w (A_a + B_a u_a^2) - c_p^w \le 0,\ p \in R_w, w \in W, \\
& x_a^2 - v_a y_a \le 0,\ a \in A, \\
& v_a - u_a y_a \le 0,\ a \in A, \\
& u_a y_a - \frac{x_a^2}{y_a} \le 0,\ a \in A,
\end{aligned}$$

where

$$F_1(X) = \sum_{p,w} \min(f_p^w, c_p^w - \pi_w), F_2(X) = \sum_{p,w} \left(c_p^w - \sum_{a\in A} \delta_{ap}^w (A_a + B_a u_a^2) \right),$$
$$P = \{X = (x, y, f, c, \pi, u, v) \mid X \text{ satisfies } (5.2), (5.3), (5.5), (5.6), (5.8) \text{ and } u_a \ge 0,\ \forall a\}.$$

Obviously, P is a polyhedral convex set in $\mathbb{R}^n$ with $n = 4nL + 2nP + nOD$ (nL is the number of links, nP is the total number of paths, nOD is the number of OD pairs).

It is easy to prove that if an optimal solution X^* to (5.16) satisfies $F_1(X^*) = 0$ and $F_2(X^*) = 0$ then it is an optimal solution to the original problem. Furthermore, according to Theorem 1.5 in Chapter 1, for large numbers t_1, t_2, the optimal solution to the problem (5.16) will be in a region where F_1, F_2 are relatively small. For this reason, we consider the penalized problem (5.16) with sufficiently large values of t_1, t_2. Using the equalities

$$MN = \frac{1}{2}(M+N)^2 - \frac{1}{2}\left(M^2+N^2\right) = \frac{1}{2}\left(M^2+N^2\right) - \frac{1}{2}(M-N)^2,$$

the problem (5.16) can be reformulated as the general DC program below:

$$\begin{aligned} \min \quad & G(X) - H(X) && (5.17)\\ \text{s.t.} \quad & X \in P, \\ & \sum_{a\in A} \delta_{ap}^w (A_a + B_a u_a^2) - c_p^w \le 0,\ p \in R_w, w \in W, \\ & G_{ia}(X) - H_{ia}(X) \le 0,\ i = 1,2,3;\ a \in A, \end{aligned}$$

$$\begin{aligned} \text{where} \quad G(X) &= \sum_{w\in W} q_w \pi_w + \theta \sum_{a\in A} g_a(y_a) + t_2 \sum_{p,w} \left(c_p^w - \sum_{a\in A} \delta_{ap}^w A_a \right), \\ H(X) &= t_1 \sum_{p,w} \max(-f_p^w, -c_p^w + \pi_w) + t_2 \sum_{p,w} \left(\sum_{a\in A} \delta_{ap}^w B_a u_a^2 \right), \\ G_{1a}(X) &= x_a^2 + \frac{1}{2}\left(v_a^2 + y_a^2\right),\ H_{1a}(X) = \frac{1}{2}(v_a + y_a)^2, \\ G_{2a}(X) &= v_a + \frac{1}{2}\left(u_a^2 + y_a^2\right),\ H_{2a}(X) = \frac{1}{2}(u_a + y_a)^2, \\ G_{3a}(X) &= \frac{1}{2}(u_a^2 + y_a^2),\ H_{3a}(X) = \frac{x_a^2}{y_a} + \frac{1}{2}(u_a - y_a)^2. \end{aligned}$$

Adapting Algorithm 1.3 for general DC programs presented in Chapter 1, we propose a DCA for solving the problem (5.17). At each iteration k, we compute

$$Y^k \in \partial H(X^k),\ Y_{ia}^k \in \partial H_{ia}(X^k),\ i = 1,2,3;\ a \in A$$

and then solve the following convex problem:

$$\begin{aligned} \min \quad & G(X) - \langle Y^k, X\rangle + ts && (5.18)\\ \text{s.t.} \quad & X \in P, \\ & \sum_{a\in A} \delta_{ap}^w (A_a + B_a u_a^2) - c_p^w \le 0,\ p \in R_w, w \in W, \\ & G_{ia}(X) - H_{ia}(X^k) - \langle Y_{ia}^k, X - X^k\rangle \le s,\ i = 1,2,3;\ a \in A, \\ & s \ge 0. \end{aligned}$$

A subgradient $Y = (\widehat{x}, \widehat{y}, \widehat{f}, \widehat{c}, \widehat{\pi}, \widehat{u}, \widehat{v}) \in \partial H(X)$ can be selected as follows:

$$\widehat{x} = \widehat{y} = 0,\ \widehat{f_p^w} = \begin{cases} -t_1 & \text{if } f_p^w < c_p^w - \pi_w \\ 0 & \text{otherwise} \end{cases},\ \widehat{c_p^w} = \begin{cases} 0 & \text{if } f_p^w < c_p^w - \pi_w \\ -t_1 & \text{otherwise} \end{cases} \tag{5.19}$$

$$\widehat{\pi_w} = \begin{cases} 0 & \text{if } f_p^w < c_p^w - \pi_w \\ t_1 & \text{otherwise} \end{cases},\ \widehat{u_a} = 2t_2 \sum_{p,w} \delta_{ap}^w B_a u_a,\ \widehat{v} = 0. \tag{5.20}$$

Since H_{ia} is differentiable, $\partial H_{ia}(X) = \{\nabla H_{ia}(X)\}$, $i = 1, 2, 3;\ a \in A$. Consider $Y = (\widehat{x}, \widehat{y}, \widehat{f}, \widehat{c}, \widehat{\pi}, \widehat{u}, \widehat{v})$, we have

$$Y = \nabla H_{1a}(X) \Leftrightarrow \widehat{x} = \widehat{u} = 0, \widehat{f} = \widehat{c} = 0, \widehat{\pi} = 0,\ \widehat{y_a} = \widehat{v_a} = v_a + y_a, \tag{5.21}$$

$$Y = \nabla H_{2a}(X) \Leftrightarrow \widehat{x} = \widehat{v} = 0, \widehat{f} = \widehat{c} = 0, \widehat{\pi} = 0,\ \widehat{y_a} = \widehat{u_a} = u_a + y_a, \tag{5.22}$$

$$Y = \nabla H_{3a}(X)$$

$$\Leftrightarrow \widehat{x_a} = \frac{2x_a}{y_a}, \widehat{y_a} = -\frac{x_a^2}{y_a^2} + y_a - u_a, \widehat{f} = \widehat{c} = 0, \widehat{\pi} = 0, \widehat{v} = 0, \widehat{u_a} = u_a - y_a. \tag{5.23}$$

The general DCA for solving (5.17) is summarized in the following algorithm.

Algorithm 5.1 DCA_CENDP for solving the problem (5.17).

```
Initialization. Choose an initial point $X^0 \in \mathbb{R}^n$ and a penalty parameter $t = t_0$. Set $k = 0$ and let $\varepsilon_1, \varepsilon_2$ be sufficiently small positive numbers.
Repeat
    1. Compute $Y^k \in \partial H(X^k),\ Y_{ia}^k \in \partial H_{ia}(X^k),\ i = 1, 2, 3;\ a \in A$ via (5.19)-(5.23).
    2. Compute $(X^{k+1}, s^{k+1})$ as a solution to (5.18).
    3. $k \leftarrow k + 1$.
Until $\left(\frac{\|X^k - X^{k-1}\|}{\|X^{k-1}\| + 1} < \varepsilon_1 \text{ or } \frac{|F(X^k) + ts^k - (F(X^{k-1}) + ts^{k-1})|}{|F(X^{k-1}) + ts^{k-1}| + 1} < \varepsilon_1\right)$ and $s \le \varepsilon_2$.
```

5.4 Numerical results

In order to illustrate the efficiency of the proposed algorithm, numerical experiments were performed on a network consisting of 6 nodes and 16 links (Figure 5.1). This network has been used to test the different algorithms for solving the CENDP (see, e.g., [11, 19, 71, 96, 100]). The travel demand for this network was considered in two cases (given in Table 5.1). The detailed data including the parameters of travel cost functions and enhancement cost functions, the capacity of links before expansion, the upper bound of the capacity improvements can be found in [19]. Our algorithm was compared with eight other methods. The abbreviations for these methods and their sources are listed in Table 5.2. We also made a comparison with a good lower bound of the CENDP (mentioned in [11] and labeled as “SO”).

Table 5.1 – Level of travel demand

	Case I	Case II
Demand from node 1 to node 6	5	10
Demand from node 6 to node 1	10	20
Total travel demand	15	30

In the DCA_CENDP, we took $\varepsilon_1 = \varepsilon_2 = 10^{-6}$, the initial point was chosen as follows:

$$X^0 = (x^0, y^0, f^0, c^0, \pi^0, u^0, v^0)$$

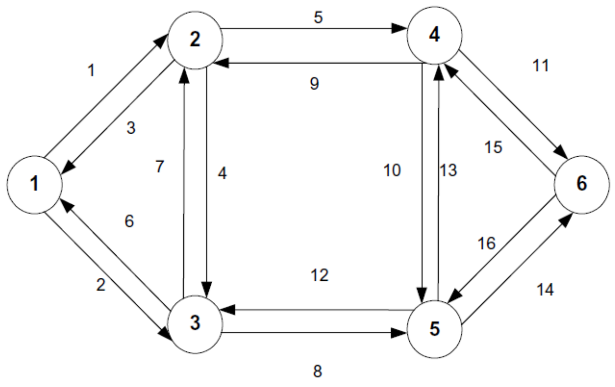

Figure 5.1 – 16-link network [11]

Table 5.2 – Abbreviation of method names

Abbreviation	Name	Source
EDO	Equilibrium Decomposed Optimization	[96]
SA	Simulated Annealing algorithm	[19]
AL	Augmented Lagrangian algorithm	[71]
GP	Gradient Projection method	[11]
CG	Conjugate Gradient projection method	[11]
QNEW	Quasi-NEWton projection method	[11]
PT	PARATAN version of gradient projection method	[11]
MILP	Mixed-Integer Linear Program transformation	[100]
DCA_CENDP	DC Algorithm for the CENDP	this chapter

with

$$x^0 = y^0 = u^0 = v^0 = 0,\ f^0 = 0,\ c_p^{w,0} = \sum_{a \in A} \delta_{ap}^w A_a,\ \pi_w^0 = \min\left\{c_p^{w,0} : p \in R_w\right\}$$

The penalty parameters were taken to be $t_1 = 350, t_2 = 100, t = 500$ and $t_1 = 400, t_2 = 100, t = 450$ for the cases I and II respectively.

The computational results for the cases I and II are summarized in Table 5.3 and Table 5.4 respectively. These results are taken from the previous works (see [11, 100]) except for the results of DCA_CENDP. In Tables 5.3 and 5.4, we use the following notations:

- $\triangle y_a$: the capacity enhancement on link a.
- Obj: the value of the objective function.
- Gap: the relative difference with respect to SO defined by

$$\text{Gap}(\%) = \frac{100(\text{Obj} - \text{LB})}{\text{LB}}$$

where LB is the lower bound of the CENDP.

From Table 5.3 and Table 5.4, we observe that the value of the objective function provided by DCA_CENDP is quite close to the lower bound of the CENDP (the relative

Table 5.3 – Numerical results of DCA_CENDP and the existing algorithms in **case I**

	EDO	**SA**	**AL**	**GP**	**CQ**	**QNEW**	**PT**	**MILP**	**DCA_CENDP**	**SO**
$\triangle y_1$	0	0	0	0	0	0	0	0	0	0
$\triangle y_2$	0	0	0	0	0	0	0	0	0	0
$\triangle y_3$	0.13	0	0.0062	0	0	0	0	0	0	0
$\triangle y_4$	0	0	0	0	0	0	0	0	0	0
$\triangle y_5$	0	0	0	0	0	0	0	0	0	0
$\triangle y_6$	6.26	3.1639	5.2631	5.8302	6.1989	6.0021	5.9502	4.41	4.8487	5.9979
$\triangle y_7$	0	0	0.0032	0	0	0	0	0	0	0
$\triangle y_8$	0	0	0	0	0	0	0	0	0	0
$\triangle y_9$	0	0	0	0	0	0	0	0	0	0
$\triangle y_{10}$	0	0	0	0	0	0	0	0	0	0
$\triangle y_{11}$	0	0	0.0064	0	0	0	0	0	0	0
$\triangle y_{12}$	0	0	0	0	0	0	0	0	0	0
$\triangle y_{13}$	0	0	0	0	0	0	0	0	0	0
$\triangle y_{14}$	0	0	0	0	0	0	0	0	0	0
$\triangle y_{15}$	0.13	0	0.7171	0.87	0.0849	0.1846	0.5798	0	0	0.1449
$\triangle y_{16}$	6.26	6.7240	6.7561	6.1090	7.5888	7.5438	7.1064	7.7	7.5621	7.5443
Obj	201.84	**198.104**	202.991	202.24	199.27	198.68	200.60	199.781	**199.656**	193.39
Gap	4.37	2.44	4.96	4.58	3.04	2.74	3.73	3.30	3.24	0

Table 5.4 – Numerical results of DCA_CENDP and the existing algorithms in **case II**

	EDO	**SA**	**AL**	**GP**	**CQ**	**QNEW**	**PT**	**MILP**	**DCA_CENDP**	**SO**
$\triangle y_1$	0	0	0	0.1013	0.1022	0.0916	0.101	0	0	0.1165
$\triangle y_2$	4.88	0	4.6153	2.1818	2.1796	2.1521	2.1801	4.41	4.6144	2.1467
$\triangle y_3$	8.59	10.1740	9.8804	9.3423	9.3425	9.1408	9.3339	10.00	9.9166	9.3447
$\triangle y_4$	0	0	0	0	0	0	0	0	0	0
$\triangle y_5$	0	0	0	0	0	0	0	0	0	0
$\triangle y_6$	7.48	5.7769	7.5995	9.0443	9.0441	8.8503	9.0361	7.42	7.3442	9.0424
$\triangle y_7$	0.26	0	0.0016	0	0	0	0	0	0	0
$\triangle y_8$	0.85	0	0.6001	0.008	0.0074	0.0114	0.0079	0.54	0.5922	0
$\triangle y_9$	0	0	0.001	0	0	0	0	0	0	0
$\triangle y_{10}$	0	0	0	0	0	0	0	0	0	0
$\triangle y_{11}$	0	0	0	0	0	0	0	0	0	0
$\triangle y_{12}$	0	0	0.1130	0.0375	0.0358	0.0377	0	0	0	0
$\triangle y_{13}$	0	0	0	0	0	0	0	0	0	0
$\triangle y_{14}$	1.54	0	1.3184	0.0089	0.0083	0.0129	0.0089	1.18	1.3153	0
$\triangle y_{15}$	0.26	0	2.7265	1.9433	1.9483	1.9706	1.9429	0	0	1.7995
$\triangle y_{16}$	12.52	17.2786	17.5774	18.9859	18.986	18.575	18.9687	19.50	20	18.9875
Obj	540.74	528.497	532.71	534.017	534.109	534.08	534.02	523.627	**522.644**	512.013
Gap	5.61	3.22	4.04	4.30	4.32	4.31	4.30	2.27	2.08	0

difference is 3.24% and 2.08 % in the cases I and II respectively). In the case I with the lower travel demand, DCA_CENDP outperforms 5 out of 8 algorithms. Although the value of the objective function given by DCA_CENDP is not better than those given by SA, CQ and QNEW, the relative difference between this value and the best one (found by SA, a global optimization technique) is fairly small (0.78%). In the case II, when the travel demand increases, DCA_CENDP yields the value of the objective function that is superior to those computed by all the other methods.

5.5 Conclusions

We proposed an algorithm based on DC programming and DCA_CENDP for handling the CENDP. We considered the MPCC formulation for this problem and employed a penalty technique to reformulate it as a general DC program. Numerical experiments on a small network indicate that DCA_CENDP solves the CENDP more effectively than the existing methods. Besides, the value of the objective function provided by DCA_CENDP is quite close to the lower bound of the CENDP. This shows that the solution found by DCA_CENDP may be global, even though DCA_CENDP is a local optimization approach.

Conclusions

In this dissertation, we studied three subclasses of mathematical programs with equilibrium constraints (MPEC) and some important applications in networks, resource allocations and transportation. Our approaches are based on DC (Difference of Convex functions) programming and DCA (DC Algorithm), which are powerful solvers for several difficult nonconvex optimization problems. The key idea of our work is the penalty techniques which constitute the bridge between MPEC and DC programs. In this spirit, the choice of penalty functions was carefully studied and validated by theoretical tools in order to get appropriate DC programs which are equivalent to original problems (the DCLCC and the bilevel program, Chapters 2 and 4 respectively) and are favorable for using DCA (the MPEC with DC objective function, Chapter 3). And throughout the thesis, how to design efficient DCA based algorithms for the resulting DC programs is our constant concern.

For DCLCC, the largest class of mathematical programs with linear complementarity constraints (Chapter 2), using four penalty functions based on the min/Fischer-Burmeister function to penalize the complementarity constraints, we reformulated the DCLCC as standard DC programs. Generally speaking, standard DC programs (convex constraints) are easier than general DC programs (including DC constrains). However, it is not the same in our work: for two of four DC programs, reformulating them as general DC programs are more suitable for investigating DCA. Interestingly, numerical results to the quadratic programs with linear complementarity constraints and the asymmetric eigenvalue complementarity problems showed that the two corresponding general DCAs are more efficient to tackle the complementarity constraints than the two standard DCAs. Moreover, for these two problems, all DCA schemes consist of solving iteratively convex quadratic problems with linear constraints. The numerical experiments on several benchmark data indicated the efficiency of our approaches and their superiority comparing with the KNITRO solver and a previous DCA scheme for the asymmetric eigenvalue complementarity problems. Our approaches can be applied to DCLCC problems with a large number of complementarity constraints.

For DC programs with variational inequality constraints (which is also a large class of problems of MPEC, Chapter 3), we have voluntarily chosen the penalty function having special properties (it is a continuously differentiable function with Lipschitz continuous gradient) so that the penalized problem is a DC program with an efficient DC decomposition. Furthermore, we proposed a variant of DCA, DCA_ρ, for improving DCA. Moreover, we improve further DCA_ρ, by its accelerated version, ADCA_ρ, that incorporates an extrapolation step at each iteration. We proved that the convergence properties of DCA still valid for these advanced versions. These DCAs become quite simple for the special case where the objective function is continuously differentiable with Lipschitz continuous gradient. Numerical experiments on an important application - the second-best toll pricing problem with fixed demands, through three networks with data taken from MacMPEC show that in general, ADCA_ρ is better than DCA_ρ and both two algorithms are superior to DCA and KNITRO. These DCA versions could be useful for other applications of MPEC.

As for the bilevel problem studied in Chapter 4, exploiting its special structure, we can formulate it as a DC program with binary variables. Thanks to exact penalty techniques for DC programs recently developed, we reformulated this combinatorial problem as a standard DC program and then proposed a suitable DCA scheme for the resulting problem. The proposed DCA enjoys interesting properties when the convex part in the objective function is linear: the sequence generated by DCA is always binary. Furthermore, the DCA scheme in this case is very inexpensive, it successively solves box constrained linear programs whose solutions are explicitly computed. In our experiments, we considered an maximum flow network interdiction problem and also proposed a mixed integer linear program formulation, which can be globally solved by the CPLEX software. Numerical results on several randomly generated networks indicated that our method is efficient, it usually provides a global solution in a short time.

Finally, for handling the continuous equilibrium network design problem (CENDP, Chapter 5), a very hard application of MPCC, the min function was used in our penalty technique and the MPCC was reformulated as a general DC program. Even when the complementarity constraints are penalized, this general DC program is still very hard. To date, the existing approach for the MPCC formulation of the CENDP can deal only with small networks. Preliminary numerical results showed the superiority of DCA with respect to eight existing methods.

In summary, we have offered several DCA based schemes for MPECs, a classical and challenging topic of nonconvex optimization. As we addressed large classes of problems, these DCAs can be exploited to solve numerous models of MPECs and their applications in various domains. The promising numerical results suggested us to develop DC programming and DCA for other classes and other applications of MPECs.

We can improve further the proposed approaches to get a better performance by studying the remaining issues which have not yet been considered in this dissertation.

The first is the value of penalty parameters. In one hand, the penalty techniques require a sufficiently large penalty parameter, but such a value can not be computed exactly in general. In another hand, a quite large penalty parameter may make DCA inefficient. While a good theoretical choice for this parameter is still an open question, we believe that advanced DCA version such as DCA_ρ, could be good to overcome this issue from computational point of views.

The second concerns the methods for convex subproblems in DCA based algorithms.

To deal with large-scale setting, scalable and efficient algorithms for convex subproblems should be investigated. For instance, we have used CPLEX solvers for convex quadratic programs. Fast and scalable algorithms such as the combined DCA-interior point methods [83] should be used for large-scale problems.

The third issue is the development of advanced DCA based algorithms such as DCA$_\rho$, ADCA$_\rho$ to DCLCC and other problems considered in this thesis.

Finally, our work has been only concerned with deterministic models while uncertainty is often presented in practice. Thus, it could be interesting to study stochastic models and methods to deal with problems involving uncertain data. In future works, we will investigate DC programming and DCA for stochastic variational inequality problems and stochastic complementarity problems.

Bibliography

[1] Abdulaal, M. and LeBlanc, L. J. (1979). Continuous equilibrium network design models. *Transportation Research Part B*, 13:19–32.

[2] Akgün, I., Barbaros, C. T., and Wood, R. K. (2011). The multi-terminal maximum-flow network-interdiction problem. *European Journal of Operational Research*, 211:241–251.

[3] Audet, C., Hansen, P., Jaumard, B., and Savard, G. (1997). Links between linear bilevel and mixed 0-1 programming problems. *J. Optim. Theory Appl.*, 93(2):273–300.

[4] Audet, C., Savard, G., and Zghal, W. (2007). New Branch-and-Cut Algorithm for Bilevel Linear Programming. *J. Optim. Theory. Appl.*, 134:353–370.

[5] Bard, J. F. (1991). Some properties of the bilevel programming problem. *J. Optim. Theory Appl.*, 68:371–378.

[6] Bard, J. F. and Mooreise, J. T. (1990). A branch and bound algorithm for the bilevel programming problem. *SIAM J. Sci. Stat. Comput.*, 11(2):281–292.

[7] Bazaraa, M. S., Sherali, H. D., and Shetty, C. M. (2006). *Nonlinear programming Theory and Algorithms*. John Wiley and Sons, Inc., third edition.

[8] Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1):183–202.

[9] Burch, C., Carr, R., Krumke, S., Marathe, M., Phillips, C., and E., S. (2003). *Network Interdiction and Stochastic Integer Programming*, volume 22 of *Operations Research/Computer Science Interfaces Series*, chapter A Decomposition-Based Pseudoapproximation Algorithm for Network Flow Inhibition, pages 51–68. Springer, Boston, MA.

[10] Burdakov, O. P., Kanzow, C., and Schwartz, A. (2016). Mathematical programs with cardinality constraints: Reformulation by complementarity-type constraints and a regularization method. *SIAM J. Optim.*, 26(1):397–425.

[11] Chiou, S.-W. (2005). Bilevel programming for the continuous transport network design problem. *Transportation research part B*, 39:361–383.

[12] Demiguel, V., Friedlander, M. P., Nogales, F. J., and Scholtes, S. (2005). A two-sided relaxation scheme for mathematical programs with equilibrium constraints. *SIAM J. Optim.*, 16(2):587–609.

[13] Douglas S. Altner, D. S., Ergun, O., and Uhan, N. A. (2010). The maximum flow network interdiction problem: Valid inequalities, integrality gaps, and approximability. *Operations Research Letters*, 38:33–38.

[14] Edmunds, T. A. and Bard, J. F. (1992). An algorithm for the mixed-integer nonlinear bilevel programming problem. *Annals of Operations Research*, 34:149–162.

[15] Facchinei, F., Jiang, H., and Qi, L. (1999). A smoothing method for mathematical programs with equilibrium constraints. *Math. Program.*, 85:107–134.

[16] Facchinei, F. and Pang, J. S. (2003). *Finite-Dimensional Variational Inequalities and Complementarity Problems*, volume II. Springer.

[17] Fontaine, P. and Minner, S. (2014). Benders decomposition for discrete-continuous linear bilevel problems with application to traffic network design. *Transportation Research Part B*, 70:163–172.

[18] Ford, L. and Fulkerson, D. (1962). *Flows in Networks.* Princeton University Press, Princeton, New Jersey.

[19] Friesz, T. L., Cho, H. J., Mehta, N. J., Tobin, R. L., and Anandalingam, G. (1992). A simulated annealing approach to the network design problem with variational inequality constraints. *Transportation Science*, 26(1):18–26.

[20] Friesz, T. L., Tobin, R. L., Cho, H. J., and Mehta, N. J. (1990). Sensitivity analysis based heuristic algorithms for mathematical programs with variational inequality constraints. *Math. Program.*, 48:265–284.

[21] Fukushima, M., Luo, Z. Q., and Pang, J. S. (1998). A globally convergent sequential quadratic programming algorithm for mathematical programs with linear complementarity constraints. *Comput. Optim. Appl.*, 10:5–34.

[22] Fukushima, M. and Pang, J. S. (1999). Convergence of a smoothing continuation method for mathematical problems with complementarity constraints. In Théra, M. and Tichatschke, R., editors, *Ill-posed Variational Problems and Regularization Techniques*, volume 477 of *Lecture Notes in Economics and Mathematical Systems*, pages 99–101. Springer, Berlin, Heidelberg.

[23] Ghare, P., Montgomery, D., and Turner, W. (1971). Optimal interdiction policy for a flow network. *Naval Research Logistics Quarterly*, 18(1):37–45.

[24] Gümüs, Z. and Floudas, C. (2005). Global optimization of mixed-integer bilevel programming problems. *Comput. Manag. Sci.*, 2:181–212.

[25] Haddou, M. (2009). A new class of smoothing methods for mathematical programs with equilibrium constraints. *Pacific Journal of Optimization*, 5(1):86–96.

[26] Harker, P. T. and Choi, S. (1991). A penalty function appoach for mathematical programs with variational inequality constraints. *Information and Decision Technologies*, 17:41–50.

[27] Hobbs, B. F., Metzler, C. B., and Pang, J. S. (2000). Strategic gaming analysis for electric power systems: an MPEC approach. *IEEE Trans. Power Syst.*, 15(2):638–645.

[28] Hu, J., Mitchell, J. E., Pang, J. S., and Yu, B. (2012). On linear programs with linear complementarity constraints. *J. Global Optim.*, 53:29–51.

[29] Hu, X. and Ralph, D. (2004). Convergence of a penalty method for mathematical programming with equilibrium constraints. *J. Optim. Theory Appl.*, 123:365–390.

[30] Huang, X. X., Yang, X. Q., and Zhu, D. L. (2006). A sequential smooth penalization approach to mathematical programs with complementarity constraints. *Numer. Func. Anal. Opt.*, 27(1):71–98.

[31] Jara-Moroni, F., Pang, J. S., and Wächter, A. (2018). A study of the difference-of-convex approach for solving linear programs with complementarity constraints. *Math. Program., Special Issue: DC Programming - Theory, Algorithms and Applications*, 169(1):221–254.

[32] Jiang, H. and Ralph, D. (2003). Extension of quasi-newton methods to mathematical programs with complementarity constraints. *Comput. Optim. Appl.*, 25:123–150.

[33] Jiang, S., Zhang, J., Chen, C., and Lin, G. (2018). Smoothing partial exact penalty splitting method for mathematical programs with equilibrium constraints. *J. Glob. Optim.*, 70(1):223–236.

[34] Júdice, J. J. (2014). Optimization with linear complementarity constraints. *Pesquisa Operacional*, 34:559–584.

[35] Júdice, J. J., Sherali, H. D., and Ribeiro, I. M. (2007). The eigenvalue complementarity problem. *Comput. Optim. Appl.*, 37:139–156.

[36] Júdice, J. J., Sherali, H. D., Ribeiro, I. M., and Faustino, A. M. (2006). A complementarity-based partitioning and disjunctive cut algorithm for mathematical programming problems with equilibrium constraints. *J. Global Optim.*, 36:89–114.

[37] Kadrani, A., Dussault, J. P., and Benchakroun, A. (2009). A new regularization scheme for mathematical programs with complementarity constraints. *SIAM J. Optim.*, 20(1):78–103.

[38] Kanzow, C. and Schwartz, A. (2013). A new regularization method for mathematical programs with complementarity constraints with strong convergence properties. *SIAM J. Optim.* , 23(2):770–798.

[39] Kunapuli, G., Bennett, K. P., Hu, J., and Pang, J. S. (2008). Classification model selection via bilevel programming. *Optim. Methods Softw.*, 23:475–489.

[40] Lawphongpanich, S. and Hearn, D. W. (2004). An MPEC approach to second-best toll pricing. *Math. Program. Ser.B*, 101:33–55.

[41] Le Thi, H. A. (1997). Contribution à l'optimisation non convexe et l'optimisation globale: Théorie, algorithmes et applications. *Habilitation à Diriger des Recherches, Université de Rouen, France.*

[42] Le Thi, H. A. (2005). DC Programming and DCA: `http://www.lita.univ-lorraine.fr/~lethi/index.php/en/research/dc-programming-and-dca.html`(homepage).

[43] Le Thi, H. A., Huynh, V. N., and Pham Dinh, T. (2014). DC Programming and DCA for general DC Programs. In Do van, T., Le Thi, H. A., and Nguyen, N. T., editors, *Advanced Computational Methods for Knowledge Engineering*, Advances in Intelligent Systems and Computing, pages 15–35. Springer, Cham.

[44] Le Thi, H. A., Moeini, M., Pham Dinh, T., and Júdice, J. J. (2012a). A DC programming approach for solving the symmetric Eigenvalue Complementarity Problem. *Comput. Optim. Appl.*, 51(3):1097–1117.

[45] Le Thi, H. A. and Pham Dinh, T. (2005). The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Ann. Oper. Res.*, 133:23–46.

[46] Le Thi, H. A. and Pham Dinh, T. (2011). On solving linear complementarity problems by DC programming and DCA. *Comput. Optim. Appl.*, 50:507–524.

[47] Le Thi, H. A. and Pham Dinh, T. (2018). DC programming and DCA: thirty years of developments. *Math. Program., Special Issue: DC Programming - Theory, Algorithms and Applications*, 169(1):5–68.

[48] Le Thi, H. A., Pham Dinh, T., and Huynh, V. N. (2012b). Exact penalty and error bounds in DC programming. *J. Global Optim.*, 52:509–535.

[49] Le Thi, H. A., Pham Dinh, T., Nguyen Canh, N., and Nguyen, V. T. (2009). DC programming techniques for solving a class of nonlinear bilevel programs. *J. Global Optim.*, 44:313–337.

[50] Le Thi, H. A., Phan, D. N., and Pham Dinh, T. (2018). Advanced DCA based algorithms for nonconvex programming. *Technical Report, University of Lorraine, 37 pages.*

[51] Le Thi, H. A., Tran, D. Q., and Pham Dinh, T. (2012c). A DC programming approach for a class of bilevel programming problems and its application in portfolio selection. *Numer. Algebra Control Optim.*, 2(1):167–185.

[52] Leyffer, S. (2000). MacMPEC `http://wiki.mcs.anl.gov/leyffer/index.php/MacMPEC` (webpage).

[53] Leyffer, S., Lopez-Calva, G., and Nocedal, J. (2006). Interior methods for mathematical programs with complementarity constraints. *SIAM J. Optim.*, 17(1):52–77.

[54] Li, C., Yang, H., Zhu, D., and Meng, Q. (2012a). A global optimization method for continuous network design problems. *Transportation research Part B*, 46:1144–1158.

[55] Li, H. and Lin, Z. (2015). Accelerated proximal gradient methods for nonconvex programming. In *Advances in Neural Information Processing Systems 28*.

[56] Li, Y., Tan, T., and Li, X. (2012b). A log-exponential smoothing method for mathematical programs with complementarity constraints. *Appl. Math. Comput.*, 218:5900–5909.

[57] Lim, C. and Smith, J. C. (2007). Algorithms for discrete and continuous multicommodity flow network interdiction problems. *IIE Transactions*, 39(1):15–26.

[58] Lin, G. H. and Fukushima, M. (2003). Some exact penalty results for nonlinear programs and mathematical programs with equilibrium constraints. *J. Optim. Theory Appl.*, 118(1):67–80.

[59] Lin, G. H. and Fukushima, M. (2005). Modified relaxation scheme for mathematical programs with complementarity constraints. *Ann. Oper. Res.*, 133:63–84.

[60] Lipp, T. and Boys, S. (2016). Variations and extension of the convex-concave procedure. *Optim. Eng.*, 17:263–287.

[61] Liu, G., Ye, J., and Zhu, J. (2008). Partial exact penalty for mathematical programs with equilibrium constraints. *Set-Valued Anal.*, 16:785–804.

[62] Liu, G. S. and Ye, J. J. (2007). Merit-function piecewise SQP algorithm for mathematical programs with equilibrium constraints. *J. Optim. Theory Appl.*, 135:623–641.

[63] Liu, G. S. and Zhang, J. Z. (2002). A new branch and bound algorithm for solving quadratic programs with linear complementarity constraints. *J. Comput. Appl. Math.*, 146:77–87.

[64] Liu, X. and Sun, J. (2004). Generalized stationary points and an interior-point method for mathematical programs with equilibrium constraints. *Math. Program.*, 101:231–261.

[65] Luo, Z. Q., Pang, J. S., Ralph, D., and Wu, S. Q. (1996a). Exact penalization and stationarity conditions of mathematical programs with equilibrium constraints. *Mathematical Programming*, 75:19–76.

[66] Luo, Z. Q., Pang, J. S., and Raplp, D. (1996b). *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press.

[67] Mangasarian, O. L. and Pang, J. S. (1997). Exact penalty for mathematical programs with linear complementarity constraints. *Optimization*, 42:1–8.

[68] Marcotte, P. and Zhu, D. (1996). Exact and inexact penalty methods for the generalized bilevel programming problem. *Math. Program.*, 74:141–157.

[69] Mari, R. (2014). *Integer Bilevel Linear Programming Problems: New Results and Applications*. PhD thesis, University of Rome.

[70] McMasters, A. and Mustin, T. (1970). Optimal interdiction of a supply network. *Naval Research Logistics Quarterly*, 17:261–268.

[71] Meng, Q., Yang, H., and Bell, M. G. H. (2001). An equivalent continuously differentiable model and a locally convergent algorithm for the continuous network design problem. *Transportation Research Part B*, 35:83–105.

[72] Muu, L. and Nguyen, V. Q. (2007). On branch-and-bound algorithms for global optimal solutions to mathematical programs with affine equilibrium constraints. *Vietnam Journal of Mathematics*, 35(4):523–539.

[73] Muu, L. D., Tran Dinh, Q., Le Thi, H. A., and Pham Dinh, T. (2012). A new decomposition algorithm for globally solving mathematical programs with affine equilibrium constraints. *Acta Math. Vietnam.*, 37(2):201–218.

[74] Nemhauser, G. and Wolsey, L. (1988). *Integer and Combinatorial Optimization*. John Wiley and Sons, Inc.

[75] Niu, Y. S., Júdice, J., Le Thi, H. A., and Pham Dinh, T. (2015). Solving the quadratic eigenvalue complementarity problem by DC programming. In Le Thi, H. A., Pham Dinh, T., and Nguyen, N. T., editors, *Modelling, Computation and Optimization in Information Systems and Management Sciences*, volume 359 of *Advances in Intelligent Systems and Computing*, pages 203–214.

[76] Niu, Y. S., Pham Dinh, T., Le Thi, H. A., and Júdice, J. J. (2013). Efficient DC programming approaches for the asymmetric eigenvalue complementarity problem. *Optim. Methods Softw.*, 28(4):812–829.

[77] Outrata, J. (1994). On optimization problems with variational inequality constraints. *SIAM J. Optim.*, 4(2):340–357.

[78] Outrata, J. and Zowe, J. (1995). A numerical approach to optimization problems with variational inequality constraints. *Math. Program.*, 68:105–130.

[79] Pang, J. S. and Leyffer, S. (2004). On the global minimization of the Value-at-Risk. *Optim. Methods Softw.*, 19:611–631.

[80] Pham, N. A. and Le, D. M. (2006). Lagrangian duality algorithms for finding a global optimal solution to mathematical programs with affine equilibrium constraints. *Nonlinear Dynamics and Systems Theory*, 6(3):225–244.

[81] Pham Dinh, T. and Le Thi, H. A. (1997). Convex analysis approach to D.C. programming: theory, algorithms and applications. *Acta Math. Vietnam.*, 22(1):289–355.

[82] Pham Dinh, T. and Le Thi, H. A. (2014). Recent advances in DC programming and DCA. In Nguyen, N. T. and Le Thi, H. A., editors, *Transactions on Computational Collective Intelligence XIII*, volume 8342 of *Lecture Notes in Computer Science*, pages 1–37. Springer, Berlin, Heidelberg.

[83] Pham Dinh, T., Le Thi, H. A., and Akoa, F. (2008). Combining DCA (DC Algorithms) and interior point techniques for large-scale nonconvex quadratic programming. *Optim. Methods Softw.*, 23(4):609–629.

[84] Phan, D. N., Le, H. M., and Le Thi, H. A. (2018). Accelerated difference of convex functions algorithm and its application to sparse binary logistic regression. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence, IJCAI-18, Stockholm, Sweden, July 13-19, 2018.*

[85] Pieper, H. (2001). *Algorithms for Mathematical Programs with Equilibrium Constraints with applications to deregulated electricity market.* PhD thesis, Standford University.

[86] R. T.Hiriart-Urruty, J.-B. and Lemaréchal, C. (2001). *Fundamentals of Convex Analysis.* Springer-Verlag Berlin Heidelberg.

[87] Raghunathan, A. U. and Biegler, L. T. (2002). Barrier methods for mathematical programs with complementarity constraints (MPCCs). *Technical report, Carnegie Mellon University, Department of Chemical Engineering, Pittsburgh, PA.*

[88] Rockafellar, R. T. (1970). *Convex Analysis.* Princeton University Press.

[89] Royset, J. O. and Wood, R. K. (2007). Solving the bi-objective maximum-flow network-interdiction problem. *INFORMS Journal on Computing*, 19(2):175–184.

[90] Scheel, H. and Scholtes, S. (2000). Mathematical programs with complementarity constraints: stationary, optimality, and sensitivity. *Math. Oper. Res.*, 25(1):1–22.

[91] Scholtes, S. (2001). Convergence properties of a regularization scheme for mathematical programs with complementarity constraints. *SIAM J. Optim.*, 11(4):918–936.

[92] Scholtes, S. and Stöhr, M. (1999). Exact penalization of mathematical programs with equilibrium constraints. *SIAM J. control Optim.*, 37(2):617–652.

[93] Sinha, A., Malo, P., and Deb, K. (2018). A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE transactions on evolutionary computation*, 22(2):276–295.

[94] Steffensen, S. and Ulbrich, M. (2010). A new relaxation scheme for mathematical programs with equilibrium constraints. *SIAM J. Optim.*, 20(5):2504–2539.

[95] Sullivan, K. M. and Smith, J. C. (2014). Exact algorithms for solving a euclidean maximum flow network interdiction problem. *Networks*, 64(2):109–124.

[96] Suwansirikul, C., Friesz, T. L., and Tobin, R. L. (1987). Equilibrium decomposed optimization: a heuristic for continuous equilibrium network design problem. *Transportation Science*, 21(4):254–263.

[97] U.P., W. and A.D., H. (1996). A simple tabu search method to solve the mixed-integer linear bilevel programming problem. *European Journal of Operational Research*, 88:563–571.

[98] Veelken, S. (2008). *A new relaxation scheme for mathematical programs with equilibrium constraints*. PhD thesis, Technische Universität München.

[99] Vicente, L. N., Savard, G., and Judice, J. J. (1996). Discrete linear bilevel programming problem. *J. Optim. Theory Appl.*, 89(3):597–614.

[100] Wang, D. Z. W. and Lo, H. K. (2010). Global optimum of the linearized network design problem with equilibrium flows. *Transportation Research Part B*, 44(4):482–492.

[101] Wen, U. P. and Yang, Y. H. (1990). Algorithms for solving the mixed integer two-level linear programming problem. *Computers and Operations Research*, 17(2):133–142.

[102] Wollmer, R. (1964). Removing arcs from a network. *Operations Research*, 12(6):934–940.

[103] Wood, R. (1993). Deterministic network interdiction. *Mathematical and Computer Modelling*, 17(2):1–18.

[104] Yao, Q., Kwok, J. T., Gao, F., Chen, W., and Liu, T.-Y. (2017). Efficient inexact proximal gradient algorithm for nonconvex problems. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*, pages 3308–3314.

[105] Ye, J. J., Zhu, D. L., and Zhu, Q. J. (1997). Exact penalization and necessary optimality conditions for generalized bilevel programming problems. *SIAM J. Optim.*, 7(2):481–507.

[106] Yu, B. (2011). *A Branch and cut approach to linear programs with linear complementarity constraints*. PhD thesis, Rensselaer Polytechnic Institute.