Analyse probabiliste des systèmes temps réel

Dorin Maxim

I. Contexte scientifique et motivation de la thèse

Depuis l'article fondateur de Liu et Layland, publié en 1973 [4], la recherche dans le domaine des systèmes témps réel est avancé vers une importante maturité, ses applications étant utilisées dans différentes branches de l'industrie. De nos jours, c'est devenu habituelle de parler des différentes types de systèmes témps réel comme les "systèmes temps réel critiques", les "systèmes à criticalité mixte", etc. Néamoins, la recherche la plus importante a été dédiée à l'étude des systèmes témps réel critiques. Ses systèmes sont présents dans l'aéronautique, l'automobile, la santé et d'autres domaines d'applications. L'étude des systèmes critiques est faite non seulement d'un point de vue de leurs correctitude mais également en prennant en compte le temps.

L'analyse temporelle des systèmes critiques a été longtemps étudiée via des approches pire cas et ces approches deviennent trop pessimistes en présence d'un besoin croissant de performance; proposer des solutions dans ces conditions represente un vrai défi pour la communauté temps réel.

La grande variabilité des temps d'exécution d'un programme en présence des architectures, s'appuyant sur des composantes comme le cache, implique que les approches pire cas peuvent donner un système non-faisable même si en pratique la probabilité d'un défaillance soit vraiment réalisable pendant la durée de vie du système soit extrêment petite. De cette manière, des approches pire cas imposent un sur-dimensionnement important des architectures, en limitant l'introduction de nouvelles fonctionalités. D'ailleurs, les approches pire cas utilisent également des valeurs pire cas pour les paramètres des systèmes et obtenir ces valeurs peut s'avérrer difficile pour des architectures complexes.

Un autre exemple concernant les limites des analyses pire cas sont les systèmes événementiels pour lequels ce n'est pas toujours utile de proposer une borne pour l'arrivée des jobs générés par les interruptions venant des capteurs par exemple [1]. Le plus connu exemple d'un tel système sont les réseaux CAN (controller area networks) avec des fautes d'intérférence magnetiques (EMI) [1], [7].

Certaines industries proposent des solutions probabilistes en définissant des paramètres avec un comportament aléatoire. Par exemple dans l'industrie automobile on retrouve des capteurs pour le radar de récul qui ont les fréquences qui sont choisies aléatoirement afin d'éviter des situations de colision entre deux vehicules qui sont dos à dos (voir Figure 1) [2]. Dans le cas des fréquences identiques les radars peuvent être moins efficaces, voire inefficaces. Les capteurs générent dans ce cas des jobs qui ont des séquences d'arrivées aléatoires. Ces jobs appartiennnent à des tâches qui peuvent être vues comme sporadiques avec la période égale au temps minimum entre les arrivées de deux jobs consécutifs.

Figure 1. Deux voitures en réculant avec les radars avec la même fréquence peuvent ne pas se "voir"

Les approches probabilistes sont des alternatives aux approches pire cas. La feasabilité des systèmes est, en général, exprimée en terme de probabilités associés à différetes valeurs pour les paramètres d'un système. Dans

les dernières années l'analyse probabiliste des systèmes temps réel a pris de l'ampleur, différent travaux existent actuellement à ce sujet. Des avancées importantes ont été proposées dans l'analyse des systèmes qui ont au plus un paramètre décrit par une variable aléatoire. Ce n'est pas le cas des systèmes avec plusieurs paramètres décrits par des variables aléatoires, pour lesquels très peu de résultats existent.

Des nouveaux modèles pour les systèmes temps réel ont été proposés afin de relaxer les contraintes et mieux prendre en compte la variabilité du système. Le modèle des tâches implicites, aussi connu sous le nom de modèle de Liu et Layland, est le modèle basique des tâches temps réel avec deux paramètres le temps d'exécution et la période, l'échéance étant égale à la période. La contrainte de périodicité est relaxée dans le cadre des tâches sporadiques et ce modèle est, dans le pire cas d'un seul processor équivalent au modèle périodique. Dans les modèles cités plus haut, aucune variabilité n'est possible au niveau des temps d'exécution.

Le niveau suivant de généralisation est atteint par le modèle *multi-frame* (MF) task model, le modèle *multi-frame* généralisé (GMF) et le modèle *récurrent temps réel* (RRT). Ces modèles permettent aux tâches d'avoir des temps d'exécution différents, ainsi que des temps inter-arrivées et des échéances différents et cela d'une manière cyclique. Dans les modèles MF et GMF si une tâche a n types possibles de jobs, alors les paramètres d'un job i a les mêmes paramètres qu'un job $i + n, \forall i$.

L'hypothèse de cyclicité dans le modèle GMF implique que les instructions à l'intérieur des tâches sont toujours executées dans le même ordre et cela n'est pas nécessairement vrai.

L'hypothèse de cyclicité a été enlevée dans le modèle des tâches non-cyclique GMF, le modèle des tâches non-cyclique RRT et le modèle des graphes orientés temps réel.

Une hierarchie des modèles mentionés plus tôt est présentée dans la figure 2. Un modèle est d'autant plus général et plus expressif, plus il est haut placé dans la figure.

Nous avons placé le modèle probabiliste à un haut niveau d'expresivité et nous pensons qu'il a le potentiel d'être le modèle le plus expressif de nos jours. Cela est motivé par la possibilité dans le modèle probabiliste d'associer à une tâche une combinaison quelconque. Un modèle probabiliste qui permettrait de décrire des dépendences entre les jobs de la même tâche serait celui le plus proche du modèle RRT task model.

Le travail de cette thèse introduit des analyses nouvelles pour les systèmes temps réel probabilistes ainsi que des algorithmes optimaux d'ordonnancement pour ces systèmes et des techniques efficaces pour réduire la compléxité associée.

II. Contributions

Les principales contributions de cette thèse sont les suivantes

L'algorithmes optimaux pour les systèmes temps réel probabilistes : Pour les systèmes temps réel avec le temps d'exécution décrit par une variable aléatoire, nous avons identifié trois problèmes d'ordonnancement à priorités fixes pour lequels nous avons proposé des algorithmes d'ordonnancement optimaux. Nous soulignons que dans le cas où les jobs peuvent s'ordonnancer après leurs échéances respectives, l'ordre des tâches les plus prioritaires n'influencent pas la distribution des temps de réponse des tâches les moins prioritaires.

Nouveaux modèles pour les systèmes temps réel probabilistes : Nous avons proposé une généralisation du modèle des tâches implicites vers un modèle avec tous les paramètres décrits par des variables aléatoires.

Analyse nouvelle d'ordonnançabilité : Nous avons proposé des nouvelles analyses suivant deux directions :

- analyse pour un réseau CAN (controller area networks) pour des messages avec la gigue décrite par une variable aléatoire;
- analyse monoprocesseur des systèmes temps réel avec paramètres multiples décrits par des variables aléatoires .

Techniques pour la réduction de la complexité des analyses proposées précedement : Nous avons étudié le bénéfice des diverses techniques d'échantillonage sur la complexité de l'analyse en réduisant la taille ou le nombre de variables aléatoires, tout en introduisant un pessmisme minimale.

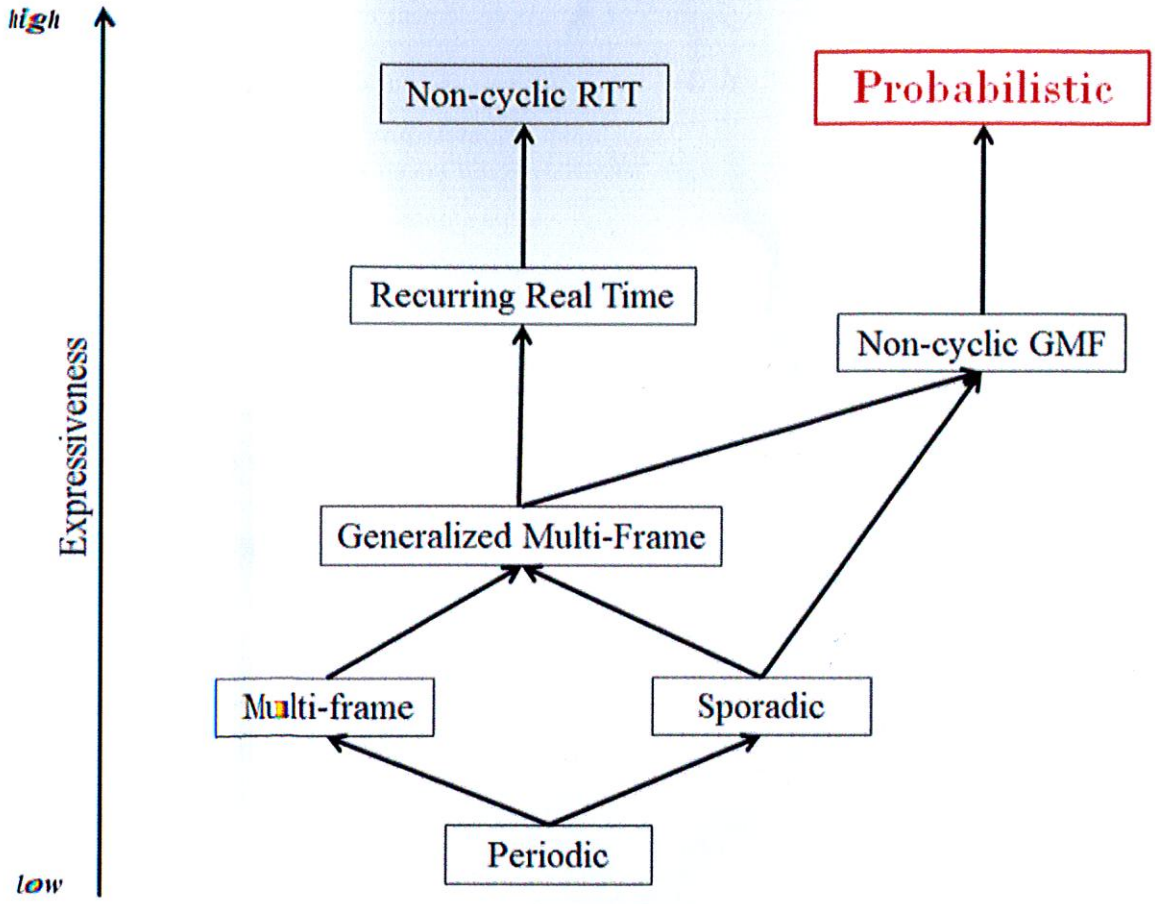

Figure 2. Hierarchie des modèles utilisés pour décrire les systèmes temps réel. Les fléches indiquent les relations entre les modèles

III. Organisation de la thèse

Nous présentons le modèle des systèmes considérés dans ce travai ainsi que les pas de la validation d'une analyse probabiliste pour un système temps réel. Nous présentons ensuite un nouveau modèle d'un système temps réel avec contraintes probabilistes multiples et son analyse. Nous présentons également des méthodes d'échantillonage pour les temps d'exécution et les temps inter-arrivées. Les conclusions de cette thèse ainsi que les perspectives ouvertes clotûrent ce résumé.

Nous présentons dans cette section le modèle des systémes temps réel probabilistes utilisés dans notre travail.

IV. Notations et définitions

Definition 1. *Un* système temps réel probabiliste *est un système temsp réel avec au moins un paramétre défini par une variable aléatoire.*

Une variable aléatoire $\mathcal{X}$ a une function de distribution (PF) $f_{\mathcal{X}}(\cdot)$ avec $f_{\mathcal{X}}(x) = P(\mathcal{X} = x)$. Les valeurs possibles de $\mathcal{X}$ appartiennent à l'intervalle $[x^{\min}, x^{\max}]$. Dans ce travail nous associons des probabilités aux valeurs possibles d'une variable aléatoire $\mathcal{X}$ en utilisant la notation suivante :

$$\mathcal{X} = \begin{pmatrix} X^0 = X^{min} & X^1 & \cdots & X^k = X^{max} \\ f_{\mathcal{X}}(X^{min}) & f_{\mathcal{X}}(X^1) & \cdots & f_{\mathcal{X}}(X^{max}) \end{pmatrix} \tag{1}$$

où $\sum_{j=0}^{k_i} f_{\mathcal{X}}(X^j) = 1$. Une variable aléatoire peut être définié également en indiquant sa function de distribution cumulative (CDF) $F_{\mathcal{X}}(x) = \sum_{z=x^{min}}^{x} f_{\mathcal{X}}(z)$.

Tout au long de ce document nous utilisons des charactères cursifs pour désigner les variables aléatoires.

Definition 2. *Deux variables aléatoires $\mathcal{X}$ et $\mathcal{Y}$ sont* **indépendantes** *(probabilistiquement) si elles décrivent des événement de telle manière que le résultat d'un des événements n'a aucun effect sur l'autre événement.*

Definition 3. *La somme $\mathcal{Z}$ de deux variables aléatoires indépendantes $\mathcal{X}$ et $\mathcal{Y}$ est la* **convolution** *$\mathcal{X} \otimes \mathcal{Y}$ où $P\{\mathcal{Z} = z\} = \sum_{k=-\infty}^{k=+\infty} P\{\mathcal{X} = k\} P\{\mathcal{Y} = z - k\}$.*

Par exemple, la somme de $\mathcal{X} = \begin{pmatrix} 3 & 7 \\ 0.1 & 0.9 \end{pmatrix}$ et $\mathcal{Y} = \begin{pmatrix} 0 & 4 \\ 0.9 & 0.1 \end{pmatrix}$ est égale à

$$\mathcal{Z} = \begin{pmatrix} 3 & 7 \\ 0.1 & 0.9 \end{pmatrix} \otimes \begin{pmatrix} 0 & 4 \\ 0.9 & 0.1 \end{pmatrix} = \begin{pmatrix} 3 & 7 & 11 \\ 0.09 & 0.82 & 0.09 \end{pmatrix}$$

Definition 4. *[5] Soit $\mathcal{X}_1$ et $\mathcal{X}_2$ deux variables aléatoires. La variable $\mathcal{X}_2$ est plus grande que $\mathcal{X}_1$ si $F_{\mathcal{X}_1}(x) \leq F_{\mathcal{X}_2}(x)$, $\forall x$, et on note par $\mathcal{X}_2 \succeq \mathcal{X}_1$.*

Dans la figure 3 $F_{\mathcal{X}_1}(x)$ ne déscend jamais en dessus de $F_{\mathcal{X}_2}(x)$. On peut remarquer que $\mathcal{X}_2$ et $\mathcal{X}_3$ ne sont pas comparable.

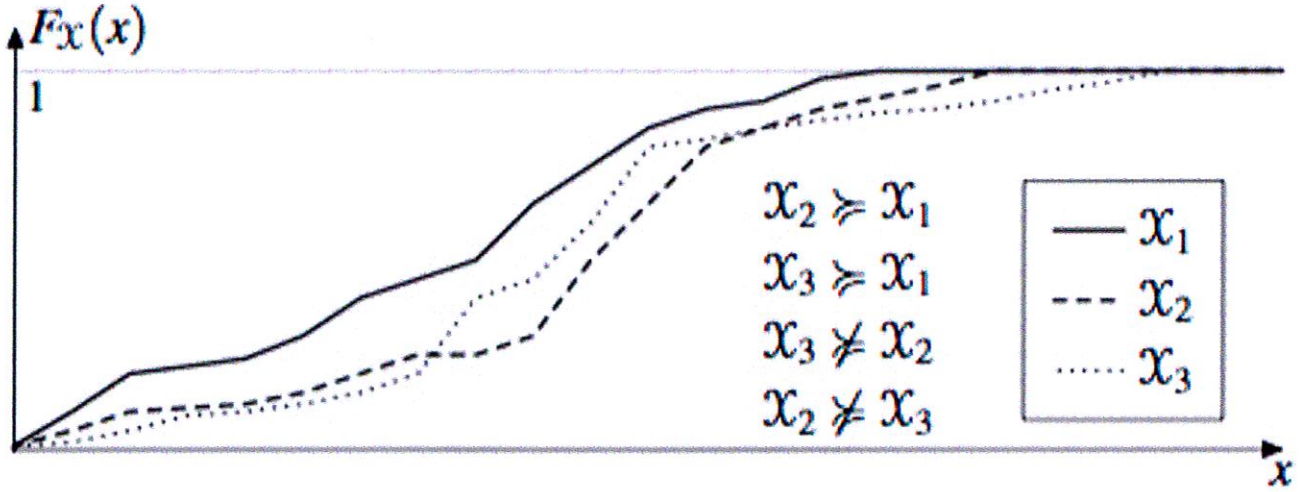

Figure 3. Relations possibles entre les CDFs des différentes variables aléatoires

V. Modèle d'un système temps réel probabiliste

On considère un système de n tâches $\{\tau_1, \tau_2, \ldots, \tau_n\}$ à départ simultanés qui sont à ordonnancer sur un processeur en utilisant une politique d'ordonnancement à priorités fixes au niveau des tâches. Sans perdre la généralité, on considère que τ_i a plus grande priorité que τ_j pour $i < j$. On note par $hp(i)$ l'ensemble des tâches avec un indice plus grand que τ_i.

Chaque tâche τ_i génère un nombre infini de jobs successives $\tau_{i,j}$, avec $j = 1, \ldots, \infty$. Tous les jobs sont indépendants d'autres jobs de la même tâche et de ceux des autres tâches.

Definition 5. *Le temps d'exécution probabiliste (pET) d'un job d'une tâche décrit la probabilité que le temps d'exécution est égal à une valeur donnée.*

Chaque tâche τ_i est une tâche sporadique généralisée [6] et elle est définie par un temps d'exécution pire cas probabiliste (pWCET) noté par $\mathcal{C}_i$ et par un temps probabiliste minimal inter-arrivées (pMIT). Ces notions sont définies plus bas.

Definition 6. *Le temps d'exécution pire cas probabiliste (pWCET) d'une tâche décrit la probabilité que le pire temps d'exécution de cette tâche soit égale à une valeur donnée.*

Un pWCET sure $\mathcal{C}_i$ est une borne sur tous les pETs $\mathcal{C}_i^j$, $\forall j$ et il peut être décrit par la relation $\succeq$ avec $\mathcal{C}_i \succeq \mathcal{C}_i^j$, $\forall j$. Graphiquement cela implique que le CDF de $\mathcal{C}_i$ reste en dessus du CDF de $\mathcal{C}_i^j$, $\forall j$.

Le temps d'exécution pire cas probabiliste $\mathcal{C}_i$ peut être écrit :

$$\mathcal{C}_i = \begin{pmatrix} C_i^0 = C_i^{\min} & C_i^1 & \cdots & C_i^{k_i} = C_i^{\max} \\ f_{\mathcal{C}_i}(C_i^{\min}) & f_{\mathcal{C}_i}(C_i^1) & \cdots & f_{\mathcal{C}_i}(C_i^{\max}) \end{pmatrix}, \tag{2}$$

où $\sum_{j=0}^{k_i} f_{\mathcal{C}_i}(C_i^j) = 1$.

Par exemple pour une tâche τ_i on peut avoir un pWCET $\mathcal{C}_i = \begin{pmatrix} 2 & 3 & 25 \\ 0.5 & 0.45 & 0.05 \end{pmatrix}$; alors $f_{C_i}(2) = 0.5$, $f_{C_i}(3) = 0.45$ et $f_{C_i}(25) = 0.05$.

En utilisant le même raisonement on définit le temps probabiliste minimal inter-arrivées (pMIT) noté par $\mathcal{T}_i$.

Definition 7. *Le temps probabiliste inter-arrivées (pIT) d'un job d'une tâche décrit la probabilité qu'un job va arriver à un instant donné.*

Definition 8. *Le temps probabiliste minimal inter-arrivées (pMIT) d'une tâche décrit la probabilité que le temps minimal inter-arrivées est égal à une valeur donnée.*

Un pMIT sure $\mathcal{T}_i$ est une borne pour les pITs $\mathcal{T}_i^j$, $\forall j$ et il peut être décrit par la relation $\succeq$ avec $\mathcal{T}_i^j \succeq \mathcal{T}_i$, $\forall j$. Graphiquement cela implique que le CDF de $\mathcal{T}_i$ reste en dessus du CDF de $\mathcal{T}_i^j$, $\forall j$.

Une tâche τ_i est donc définie par un tuple $(\mathcal{C}_i, \mathcal{T}_i)$. Un job d'une tâche doit finir son exécution avant l'arrivée du job suivant de la même tâche, i.e., task must finish its execution before the arrival of the next job of the same task, i.e., the arrival of a new job represents the deadline of the current job[1]. Thus, the task's deadline may also be represented by a random variable $\mathcal{D}_i$ which has the same distribution as its pMIT, $\mathcal{T}_i$.

Comme cela est expliqué dans [3] quand on considère the valeurs pire case probabilistes (pMIT ou pWCET), alors les variables aléatoires sont indépendantes probabilistiquement.

VI. Les propriétés imposées des systèmes

Nous avons parlé jusqu'à maintenant des propriétés implicites des systèmes, nous allons passer aux propriétés imposées par le processus de certification nécessaire aux systèmes temps réel. Une serie de contraintes découlent des fonctionnalités attendues de système et la mesure la plus importante qu'on utilise dans notre travail est le *temps de réponse*.

Le temps de réponse d'une tâche est le temps entre son activation et la fin de son exécution. Nous considérons des jobs avec des temps d'exécution probabilistes et donc le temps de réponse est également décri par une variable aléatoire.

Les constraintes sont mises sous la forme d'une *probabilité de rater son échéance*, e.g., la probabilité de rater son échéance devrait être plus petite ou égale à 10^{-9}, ou comme une fonction d'optimisation, e.g., minimisant la probabilité moyenne de rater son échéance pour toutes les tâches du système.

Definition 9 (Probabilité d'échec pour un job). *Pour un job $\tau_{i,j}$ la probabilité d'échec $DMP_{i,j}$ est la probabilité que le job j^{th} d'une tâche τ_i rate son échéance et elle est égale à:*

$$DMP_{i,j} = P(\mathcal{R}_{i,j} > D_i). \tag{3}$$

où $\mathcal{R}_{i,j}$ est la distribution du temps de réponse du job j^{th} de la tâche τ_i.

Si la tâche qu'on étudie est périodique, i.e., la distribution pMIT a seulement une valeur, alors la probabilité d'échec d'une tâche peut être calculée comme étant la moyenne des probabilités d'échec des jobs activés pendant

[1] In the analysis of GMF tasks this is known as the *frame separation* constraint.

l'intervalle $[a, b]$, qui pourrait être par exemple l'hyper-période d'un ordonnancement. Un raisonnement pire cas pourrait prendre en compte seulement le plus grand DMP parmi les DMPs de tous les jobs activés dans l'intervalle $[a, b]$.

Definition 10 (Probabilité d'échec d'une tâche). *Pour une tâche τ_i et un intervalle de temps $[a, b]$, la probabilité d'échec d'une tâche est calculé :*

$$DMR_i(a,b) = \frac{P(\mathcal{R}_i^{[a,b]} > D_i)}{n_{[a,b]}} = \frac{1}{n_{[a,b]}} \sum_{j=1}^{n_{[a,b]}} DMP_{i,j}, \tag{4}$$

où $n_{[a,b]} = \lceil \frac{b-a}{T_i} \rceil$ est le nombre de jobs d'une tâche τ_i activés dans l'intervalle $[a, b]$.

VII. Conclusions et perspectives

Le principal objectif de notre travail concerne la proposition des nouvelles techniques d'analyse pour les systèmes temps réel probabilistes. La difficulté de l'objectif vient de la relative "jeunesse" de ce domaine de recherche. Les avancées de l'analyse des systèmes probabilistes temps réel consistent principalement en généralisations des résultats pour les systèmes déterministes avec un seul paramétre probabiliste. Avant cette thèse, peu de progress avait été fait dans la direction des systèmes avec multiples paramétres probabilistes. Une autre restriction des résultats existant concerne le type de distribution accepté par les analyse qui pouvait être très limitatifs.

Cet état de l'art nous a motivé à considerer des analyses pour des systèmes temps réel avec multiples paramétres probabilistes qui généralisent non seulement le modèle, mais également l'étendue des résultats en prennant en compte des distributions quelconques. Nous avons reussi à produire ces analyses et on considère cette contribution comme la plus importante de cette thèse. Nous avons montré que la prise en compte des contraintes multiples pour les tâches temps réel peut décroitre d'une manière importante le pessimisme introduit par les analyses faites en faisant des raisonnements pire cas.

Comme travail futur, nous souhaitons d'étendre ces analyses pour prendre en compte d'autres paramètres probabilistes comme les délais d'activations, etc.

Les techniques proposées dans cette thèse ouvrent la porte à d'autres avancées dans l'état de l'art avec par exemple la définition de l'optimalité pour les systèmes probabilistes. La proposition des algorithmes optimaux dans le cadre de cette thèse a été une autre direction de recherche attaquée. Nous avons identifié trois problèmes importants d'optimisation et nous proposé pour chacun de ses problèmes un algorithme optimal. Encore plus important, pour certaines configurations d'un système le temps de réponse d'une tâche n'est pas influencée par l'ordre relatif parmi les tâches les plus prioritaires ou par celles d'une priorité moins importante que la tâche étudiée. Ce résultat permet d'appliquer un raisonnement de type Audsley pour obtenir des algorithmes optimaux c'est à dire, capable de minimiser la probabilité d'échec d'une tâche.

Nos résultats préliminaires montrent que dans le cas avec des paramètres multiples probabilistes l'ordre relatif des tâches plus prioritaires n'a aucune influence sur la distribution de temps de réponse d'une tâche et donc des algorithmes s'appuyant sur le principe d'Audsley peuvent être proposés. Néanmoins, prouver la validité de ce raisonnement est difficile à cause des intervalles d'études qui devront être établis par rapport aux temps d'inter-arrivés probabilistes.

References

[1] I. Broster and A. Burns. Applying random arrival models to fixed priority analysis. In *the Proceedings of the Work-In-Progress of the 25th IEEE Real-Time Systems Symposium (RTSS04)*, 2004.

[2] Darren Buttle. Real-time in the prime-time. In *Keynote talk at the 24th Euromicro Conference on Real-Time Systems (ECRTS)*, 2012.

[3] L Cucu-Grosjean. Independence - a missunderstood property of and for real-time systems. In *the 60th anniversary of A. Burns*, 2013.

[4] C. L. Liu and James W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *J. ACM*, 20(1):46–61, January 1973.

[5] J.M. Lopez, J. L. Díaz, J. E., and D. García. Stochastic analysis of real-time systems under preemptive priority-driven scheduling. *Real-time Systems*, 40(2):180–207, 2008.

[6] A.K. Mok. *Fundamental Design Problems of Distributed Systems for the Hard-Real-Time Environment*. PhD thesis, Laboratory for Computer Science, Massachusetts Institute of Technology, 1983.

[7] N. Navet, Y.-Q. Song, and F. Simonot. Worst-case deadline failure probability in real-time applications distributed over controller area network. *J. Syst. Archit.*, 46(7):607–617, April 2000.