Étude de la complexité de problèmes d'ordonnancement avec tâches-couplées sur monoprocesseur

Gilles Simonin

Étude de la complexité de problèmes d'ordonnancement avec tâches-couplées sur monoprocesseur

Gilles Simonin

Laboratoire du LIRMM (CNRS - UM2)
161 rue Ada
34392 Montpellier Cedex 5
gilles.simonin@lirmm.fr

RÉSUMÉ. *Nous présentons dans cet article un nouveau type de problème d'ordonnancement sur monoprocesseur avec tâches-couplées en présence d'un graphe de compatibilité. Ce type de problème est motivé par l'étude d'une torpille autonome sous-marine. Dans un premier temps, nous présentons et modélisons la problématique générale, puis nous étudions la complexité de ce type de problèmes en présence de contrainte de compatibilité. Des résultats de $\mathcal{NP}$-complétude sont obtenus et un algorithme de complexité polynomiale est donné pour résoudre deux problèmes en particulier. Enfin après une synthèse de ces résultats, nous présentons une vision globale de la complexité de cette classe de problème.*

ABSTRACT. *We present in this article a new type of scheduling problem on monoprocesssor with coupled-tasks and compatibility graph. This problem is motivated by the study of a submarine autonomous torpedo. In a first time, we present the general problem, then we study the complexity of differents problems with compatibility constraint. $\mathcal{NP}$-complete results are proven and a polynomial algorithm is given in order to solve particular problems. Finally, the results and a complexity overall view of this problem class are summarized.*

MOTS-CLÉS : *Ordonnancement, complexité, tâches-couplées, monoprocesseur, compatibilité.*

KEYWORDS: *Scheduling, complexity, coupled-tasks, mono processor, compatibility.*

1. Introduction

La théorie de l'ordonnancement est une branche de la recherche opérationnelle qui s'intéresse au calcul de dates optimales d'exécution de tâches. L'ordonnancement gère l'allocation de différentes tâches sur des ressources en fonction du temps. Les ressources et les tâches peuvent prendre différentes formes. Les ressources peuvent être des machines dans une usine, des pistes dans un aéroport, et bien d'autres encore. Les tâches quant à elles peuvent être des décollages ou atterrissages d'avion dans un aéroport, l'exécution de programmes sur un ordinateur, etc. Les objectifs peuvent aussi prendre plusieurs formes (minimisation de la longueur de l'ordonnancement, ou du nombre de tâches finies après un certain temps fixé). En trente ans, de nombreux travaux ont été réalisés pour des problèmes d'ordonnancement sur monoprocesseur, mais depuis quelques années, d'autre types de problèmes d'ordonnancement ont émergé provenant de domaines différents et de cas concrets. Ces problèmes se modélisent souvent par des structures de tâches particulières, des contraintes inhabituelles ou des objectifs très spécifiques.

1.1. *Présentation du problème*

Dans cet article, nous étudions le problème d'ordonnancer des tâches-couplées, introduites par Shapiro (Shapiro, 1980), soumises à des contraintes de compatibilité sur un monoprocesseur. Pour cela, un ensemble $\mathcal{A}=\{A_1,\ldots,A_n\}$ de tâches-couplées est donné. Chaque tâche $A_i \in \mathcal{A}$ consiste en deux sous-tâches a_i et b_i de durée d'exécution p_{a_i} et p_{b_i} séparées par un délai d'inactivité incompressible L_i. La seconde sous-tâche b_i doit démarrer son exécution exactement L_i unités de temps après l'exécution de la première sous-tâche a_i. Nous introduisons un graphe non orienté de compatibilité $G_c=(\mathcal{A}, E_c)$ où l'ensemble des tâches-couplées $\mathcal{A}$ définit les sommets du graphe et E_c représente l'ensemble des arêtes reliant deux tâches-couplées susceptibles de s'intersecter, c'est à dire de pouvoir exécuter au moins une sous-tâche d'une tâche A_i pendant le délai d'inactivité d'une autre tâche A_j (figure 1).

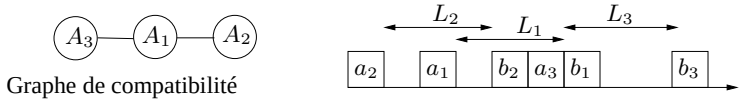

Figure 1. *Dans le graphe de compatibilité, A_2 et A_3 ne sont pas reliés par une arête, ainsi nous serons obligés d'exécuter A_2 suivi de A_3, ou l'inverse. Puisque A_1 est relié aux tâches A_2 et A_3, A_1 peut s'exécuter pendant le temps d'inactivité de A_2 et A_3.*

En utilisant la notation de Graham et al. (Graham *et al.,* 1979), les problèmes rencontrés seront définis par le triplet $\alpha|\beta|\gamma : 1|p_{a_i}, p_{b_i}, L_i, G_c|C_{max}$. 1 indique la présence d'un monoprocesseur, le triplet (p_{a_i}, p_{b_i}, L_i) définit la structure des tâches-couplées, G_c indique la présence d'un graphe de compatibilité quelconque, et enfin C_{max} désigne la fonction objectif, ici l'ordonnancement de longueur minimum.

Mesurons l'impact de l'introduction du graphe de compatibilité sur la complexité des problèmes d'ordonnancement de tâches-couplées sur un monoprocesseur. Rappelons qu'un problème de décision est dans la classe $\mathcal{P}$ (Polynomial) s'il peut être décidé en temps polynomial par rapport à la taille de la donnée. Un problème est dans la classe $\mathcal{NP}$ (Non-déterministe Polynomial) si le problème admet un algorithme polynomial capable de vérifier la validité d'une solution donnée. Ce type de problème a été étudié par Orman et Potts (Orman *et al.,* 1997) avec un graphe de compatibilité complet (où tous les sommets sont reliés entre eux, ainsi les tâches peuvent s'intersecter mutuellement). Dans notre problématique la contrainte de compatibilité sera relaxée en prenant un graphe quelconque. En comparant les résultats d'Orman et Potts et ceux en relaxant la contrainte de compatibilité, nous verrons l'impact de la contrainte de compatibilité sur ce type de problème.

1.2. *Motivation*

L'étude du problème d'ordonnancer des tâches-couplées soumises à des contraintes de compatibilité est motivée par le problème d'acquisition de données par une torpille en immersion : la torpille possède des capteurs qui collectent des informations. Une sonde émet une onde qui se propage sous l'eau pour recueillir des données, cette opération est appelée tâche d'acquisition. Il y aura deux sous-tâches, une qui envoie l'écho, l'autre qui le reçoit et un temps d'inactivité incompressible et indilatable entre les deux. Ces tâches d'acquisition peuvent être assimilées à des tâches-couplées. Pendant ce temps d'inactivité, d'autres échos peuvent être envoyés sur d'autres sondes afin d'employer le temps d'inactivité. Cependant, la localisation trop proche des ondes provoque des interférences. Sachant que nous souhaitons traiter des informations exemptes d'erreurs, un graphe de compatibilité est construit entre les tâches, décrivant celles pouvant potentiellement s'exécuter en même temps.

1.3. *État de l'art*

Le problème d'ordonnancer des tâches-couplées sans graphe de compatibilité a été étudié du point de vue de la complexité (Blazewicz *et al.*, 2001, Orman *et al.*, 1997). Dans cet article nous étudions un problème dans lequel les tâches-couplées (ou tâches d'acquisition) sont soumises à des contraintes de compatibilité. Les principaux résultats de complexité d'Orman et Potts sont donnés par le tableau 1 :

Problème	Complexité	Référence
$1\|p_{a_i} = L_i = p_{b_i}\|C_{max}$	$\mathcal{NP}$-complet	(Orman *et al.*, 1997)
$1\|p_{a_i} = a, L_i = L, p_{b_i} = b\|C_{max}$	Problème ouvert	(Orman *et al.*, 1997)
$1\|p_{a_i} = p_{b_i} = p, L_i = L\|C_{max}$	Polynomial	(Orman *et al.*, 1997)
$1\|p_{a_i}, L_i = p_{b_i} = p\|C_{max}$	Polynomial	(Orman *et al.*, 1997)

Tableau 1. *Principaux résultats de complexité d'Orman et Potts*

1.4. *Résultats*

Nous allons montrer la $\mathcal{NP}$-complétude de trois problèmes d'ordonnancement $\Pi_1 : 1|p_{a_i} = p_{b_i} = p, L_i = L, G_c|C_{max}$, $\Pi_2 : 1|p_{a_i} = p_{b_i} = L_i, G_c|C_{max}$ et enfin $\Pi_3 : 1|p_{a_i} = a, p_{b_i} = b, L_i = L, G_c|C_{max}$ avec $L, a, b, p \in \mathcal{N}$. Nous montrerons également la polynomialité de deux problèmes $\Pi_4 : 1|p_{a_i} = L_i = p, p_{b_i}, G_c|C_{max}$ et $\Pi_5 : 1|p_{b_i} = L_i = p, p_{a_i}, G_c|C_{max}$. La complexité de ces cinq problèmes sera étudiée de la même manière qu'Orman et Potts dans le but d'obtenir une visualisation globale de la complexité de ces problèmes. Un problème est dit $\mathcal{NP}$-complet s'il est au moins aussi difficile que n'importe quel problème de la classe $\mathcal{NP}$. S'il existe un algorithme polynomial qui donne une solution optimale à ce problème alors tous les problèmes de cette classe admettent un algorithme de complexité polynomiale.

2. Étude de la complexité

2.1. *Étude du problème* Π_1*, avec* $L, p \in \mathcal{N}$

D'après Orman et Potts, le problème $1|p_{a_i} = p_{b_i} = p, L_i = L, G_c\ complet|C_{max}$ est polynomial. Nous allons montrer que la relaxation des contraintes sur le graphe de compatibilité entraîne la $\mathcal{NP}$-complétude. Étudions le cas spécifique $1|p_{a_i} = p_{b_i} = p, L_i = 2p, G_c|C_{max}$. Si ce cas est $\mathcal{NP}$-complet alors le cas général Π_1 sera $\mathcal{NP}$-complet.

Théorème 2.1 *Le problème de décider si une instance de notre problème possède un ordonnancement de longueur* $\beta = \sum_{i=1}^{n}(p_{a_i} + p_{b_i}) = 2np$ *est* $\mathcal{NP}$*-complet.*

Preuve :

Afin de prouver que ce problème est $\mathcal{NP}$-complet, une réduction polynomiale va être réalisée entre le problème $\mathcal{NP}$-complet TRIANGLE PACKING et Π_1.

TRIANGLE PACKING :
Données : Soit $G = (V, E)$ un graphe avec $|V| = 3q$
Résultat : Recouvrement des sommets par des triangles dans le graphe G
Mesure : Cardinalité maximum du recouvrement

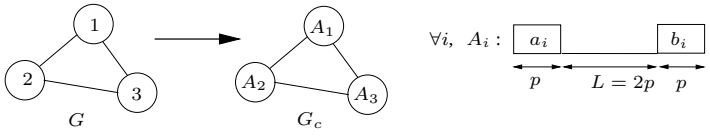

Figure 2. *Exemple de transformation polynomiale*

Soit une instance I^* de Triangle Packing, construisons une instance I de $1 \mid p_{a_i} = p_{b_i} = p$, $L_i = 2p$, $G_c \mid C_{max} = 2np$ de la manière suivante :

Soit $G = (V, E)$ le graphe de l'instance I^*. $G_c = (\mathcal{A}, E_c)$ est construit de la manière suivante (voir illustration figure 2) :

– $\forall i \in V$, où $|V| = 3q$, une tâche A_i est introduite dans $\mathcal{A}$, composée de deux sous-tâches a_i et b_i de durée d'exécution $p_{a_i} = p_{b_i} = p$ et d'un temps d'attente, appelé "slot", incompressible et indilatable entre ces deux sous-tâches de longueur $L_i = 2p$.

– Pour chaque arête $e = (i, j) \in E$ dans G, une arête $e_c = (A_i, A_j) \in E_c$ est ajoutée dans G_c, on a une relation de non exclusivité entre les deux tâches A_i et A_j.

Maintenant que l'instance I est construite, montrons que l'existence d'un recouvrement par triangle des sommets de G implique l'existence d'un ordonnancement optimal sans temps d'inactivité, et inversement :

$\Rightarrow$ Supposons qu'il existe un recouvrement en triangle des sommets du graphe G. Montrons alors qu'il existe un ordonnancement en $2np$ unités de temps. Pour cela, il suffit de regrouper les tâches d'acquisitions A_i trois par trois dans G_c selon le recouvrement en triangle trouvé dans G. L'exécution de ces groupes de tâches forme un ordonnancement sans temps d'inactivité (voir illustration figure 3) et donc en $2np$ unités de temps.

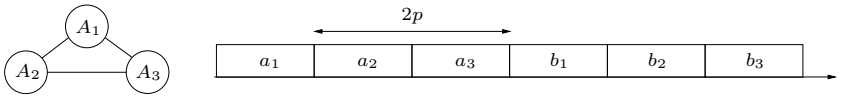

Figure 3. *Illustration de trois tâches d'acquisition formant un bloc*

$\Leftarrow$ Réciproquement, s'il existe un ordonnancement en $2np$ unités de temps, montrons alors que les sommets de G peuvent être recouverts par des triangles.

Il est clair que si $C_{max} = 2np$, il n'y a aucun temps d'inactivité sur le monoprocesseur. Ceci entraîne que chaque slot d'inactivité de longueur $L_i = 2p$ sera forcément rempli. Or il faut trois tâches d'acquisition exécutées les unes dans les autres pour avoir un bloc de trois tâches sans temps d'inactivité. Donc en ayant exactement q blocs, nous avons bien un ordonnancement sans temps d'inactivité. Et puisque trois tâches d'acquisition exécutées les unes dans les autres sont forcément compatibles dans G_c, il existe un recouvrement par triangles des sommets de G_c et des sommets de G par construction. Ce problème est donc bien $\mathcal{NP}$-complet. □

2.2. *Étude des deux problèmes* Π_2 *et* Π_3

D'après (Orman *et al.*, 1997), le problème $1|p_{a_i} = p_{b_i} = L_i, G_c\ complet|C_{max}$ est $\mathcal{NP}$-complet et le problème $1|p_{a_i} = a, p_{b_i} = b, L_i = L, G_c\ complet|C_{max}$ reste ouvert. Il est évident de voir que Π_2 est une généralisation de $1|p_{a_i} = p_{b_i} = L_i, G_c\ complet|C_{max}$ lorsque la contrainte de compatibilité est relaxé. Il est également évident de voir que le problème Π_3 est une généralisation du problème Π_1. Donc les problèmes Π_2 et Π_3 sont $\mathcal{NP}$-complets.

En résumé, nous avons montré que ces trois problèmes étaient $\mathcal{NP}$-complets, et que la relaxation de la contrainte de compatibilité faisait passer un problème ouvert et un autre polynomial à la $\mathcal{NP}$-complétude.

3. Problèmes polynomiaux

3.1. *Étude du problème* Π_4

Nous allons maintenant démontrer que le problème Π_4 est polynomial. Dans ce problème, le temps d'exécution de la première sous-tâche et le temps d'inactivité sont fixés à la même constante p. Nous allons étudier les différentes valeurs de p_{b_i} en pondérant les sommets du graphe $G_c = (\mathcal{A}, E)$ par les valeurs de p_{b_i}. Il y a trois cas possibles :

1) Pour $e = (A_k, A_l) \in E$, si $p_{b_k} > p$ et $p_{b_l} > p$, la deuxième sous-tâche de chaque tâche d'acquisition est supérieure au temps d'inactivité fixé $L_i = p$. De ce fait, A_k ne peut s'exécuter durant le temps d'inactivité de A_l. L'arête e est enlevée.

2) Pour $e = (A_k, A_l) \in E$, si $p_{b_k} \leq p$ et $p_{b_l} > p$ (ou inversement), la sous-tâche b_k est exécutée dans le temps d'inactivité de A_l (ou inversement). Un poids $max\{p_{b_k}, p_{b_l}\}$ est ajouté à l'arête e.

3) Pour $e = (A_k, A_l) \in E$, si $p_{b_k} \leq p$ et $p_{b_l} \leq p$, nous prenons la tâche de poids p_{b_i} minimum et exécutons sa sous-tâche b_i dans dans le temps d'inactivité de l'autre tâche. Un poids $min\{p_{b_k}, p_{b_l}\}$ est ajouté à l'arête e.

Le graphe de compatibilité G_c a maintenant des poids sur chaque sommet et sur chaque arête. La solution optimale consiste à trouver un couplage minimisant le poids des arêtes du couplage et des sommets isolés. Un couplage est un ensemble d'arêtes disjointes recouvrant les sommets d'un graphe. Pour trouver cette solution, construisons une structure particulière depuis G_c :

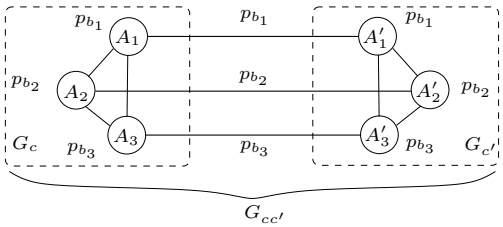

Figure 4. *Illustration de la transformation*

Soit $G_{c'}$ une copie du graphe de compatibilité G_c. Relions chaque sommet A_i de G_c vers son sommet associé A'_i dans $G_{c'}$. Nous obtenons un nouveau graphe de taille $2n$ que nous notons $G_{cc'}$. Trouver une solution à Π_4 revient à trouver un couplage

parfait (c'est à dire un couplage recouvrant tous les sommets du graphe) minimisant le poids des arêtes du couplage dans $G_{cc'}$. L'algorithme d'Edmonds permet de trouver un couplage parfait de poids minimum en un temps de complexité de $O(n^2m)$ (Edmonds, 1965). A partir de ce couplage nous ordonnançons les tâches deux à deux, puis les tâches isolées en dernier. Cet algorithme donne une solution optimale pour Π_4.

3.2. *Étude du problème* Π_5

Nous allons maintenant démontrer que le problème Π_4 est polynomial. Orman et Potts ont montré (Orman *et al.*, 1997) que $1|p_{b_i} = L_i = p, p_{a_i}|C_{max}$ et $1|p_{a_i} = L_i = p, p_{b_i}|C_{max}$ étaient symétriques. En relaxant la contrainte de compatibilité, les deux problèmes restent symétriques, donc le problème Π_5 est polynomial. Nous avons un ordonnancement optimal avec l'algorithme précédent en changeant p_{b_i} par p_{a_i}.

4. Résultats et conséquences

Nous avons montré la $\mathcal{NP}$-complétude des trois problèmes Π_1, Π_2 et Π_3, et la polynomialité des deux problèmes Π_4 et Π_5. À partir de ces résultats, nous obtenons trivialement la $\mathcal{NP}$-complétude de tous les problèmes plus généraux et la polynomialité de tous les problèmes plus spécifiques. En effet, un cas plus général qu'un problème $\mathcal{NP}$-complet est forcément $\mathcal{NP}$-complet, et inversement un cas plus spécifique qu'un problème polynomial est forcément polynomial. Une visualisation globale de la complexité de tous les résultats obtenus est présentée sur la figure 5. Pour une meilleure visibilité des résultats nous noterons nos problèmes seulement avec les paramètres des tâches d'acquisition (p_{a_i}, p_{b_i}, L_i) et le graphe de compatibilité G_c. Les graphes sont décrits de la manière suivante :

– $p_{a_i}, L_i, p_{b_i}, G_c$ indique le type de problème d'ordonnancement avec tâches d'acquisition et graphe de compatibilité

– $p_{a_i} = L_i, p_{b_i}, G_c$ et les formes similaires indiquent les problèmes où p_{b_i} est non restreint et où $\forall i\ p_{a_i} = L_i$

– $p_{a_i} = a, L_i, p_{b_i}, G_c$ et les formes similaires indiquent les problèmes où p_{b_i} et L_i sont non restreints et où $\forall i\ p_{a_i} = a$

– Enfin dans les graphes les arcs sont orientés d'un cas spécifique vers un problème plus général, et les arêtes relient deux problèmes symétriques.

5. Conclusion

Dans cet article, un problème d'ordonnancement avec tâches-couplées et relaxation des contraintes de compatibilité a été présenté et modélisé. Ce problème est motivé dans la pratique par la gestion des acquisitions d'une torpille sous-marine autonome travaillant sur un monoprocesseur. En se basant sur les travaux d'Orman et Potts (Orman *et al.*, 1997) pour ce type de tâche, une étude de la complexité en relaxant la contrainte de compatibilité a été faite. Dans un premier temps, la $\mathcal{NP}$-complétude

de trois problèmes en particulier a été montrée, puis la polynomialité de deux autres problèmes en donnant un algorithme polynomial. À partir de ces résultats, nous avons une vision générale de la complexité des différents cas que nous pouvons rencontrer sur ce genre de problématique. Il est également possible d'observer que la contrainte de compatibilité amène la $\mathcal{NP}$-complétude sur un cas particulier qui était polynomial. Il pourrait être intéressant d'étendre ces résultats dans le cas de systèmes multiprocesseurs, ou encore de les étendre pour des problèmes stochastiques et on-lines.

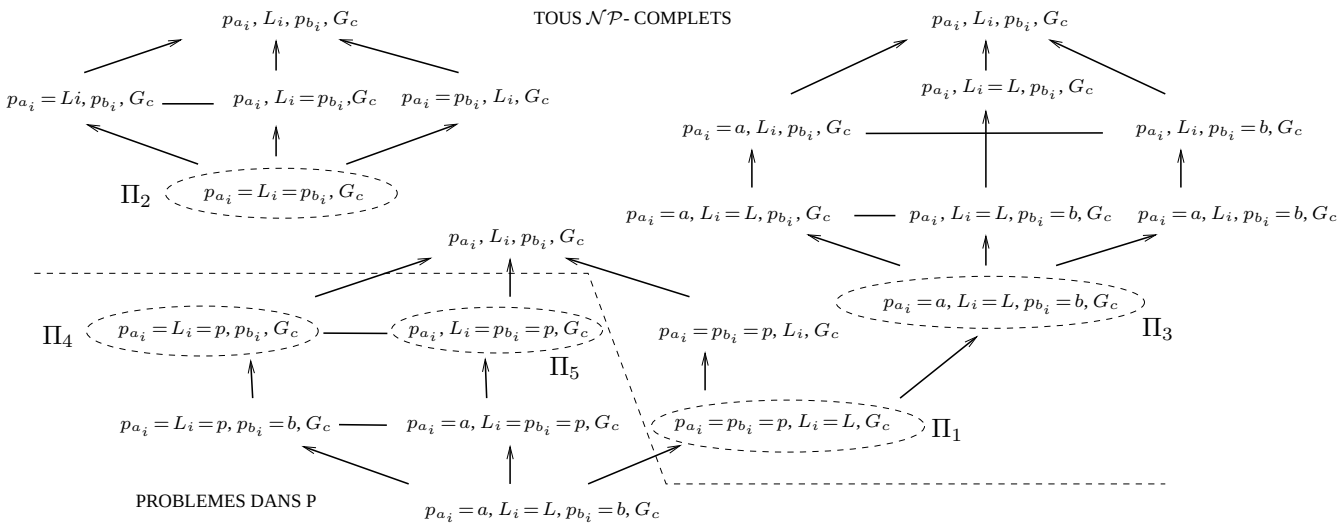

Figure 5. *Visualisation globale de la complexité des problèmes d'ordonnancement avec tâches-couplées en présence de contrainte de compatibilité*

6. Bibliographie

Blazewicz J., Ecker K., Kis T., Tanas M., « A note on the complexity of scheduling coupled-tasks on a single processor », *Journal of the Brazilian Computer Society*, vol. 7, n° 3, p. 23-26, 2001.

Edmonds J., « Maximum Matching and a Polyhedron with $0, 1$ Vertices », *Journal of Research of the National Bureau of Standards*, vol. 69 B, p. 125-130, 1965.

Garey M., Johnson D., *Computers and Intractability ; A Guide to the Theory of NP-Completeness*, W. H. Freeman & Co., New York, NY, USA, 1990.

Graham R., Lawler E., Lenstra J., Kan A. R., « Optimization and Approximation in Deterministic Sequencing and Scheduling : a Survey », *Annals of Discrete Mathematics*, vol. 5, p. 287-326, 1979.

Orman A., Potts C., « On the Complexity of Coupled-Task Scheduling », *Discrete Applied Mathematics*, vol. 72, p. 141-154, 1997.

Shapiro R., « Scheduling coupled tasks », *Naval Research Logistics Quarterly*, vol. 27, p. 477-481, 1980.