



HAL
open science

Complexité paramétrée pour le problème d'ordonnement sur une machine avec instants de début et fin interdits

Michaël Gabay, Nadia Brauner, Christophe Rapine

► **To cite this version:**

Michaël Gabay, Nadia Brauner, Christophe Rapine. Complexité paramétrée pour le problème d'ordonnement sur une machine avec instants de début et fin interdits. 14ème congrès annuel de la Société française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF), Feb 2013, Troyes, France. hal-00752640

HAL Id: hal-00752640

<https://hal.science/hal-00752640>

Submitted on 19 Nov 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Complexité paramétrée pour le problème d’ordonnancement sur une machine avec instants de début et fin interdits

Michaël Gabay¹, Nadia Brauner¹, Christophe Rapine²

¹ G-SCOP, Grenoble-INP, UJF, CNRS, University of Grenoble,
46 avenue Félix Viallet, 38031 Grenoble Cedex, France

{prénom.nom}@g-scop.grenoble-inp.fr

² Laboratoire LGIPM, Université de Lorraine, île du Sauley,
57045 Metz Cedex 01, France

christophe.rapine@univ-lorraine.fr

Mots-clés : *Ordonnancement, Complexité, High multiplicity, Contraintes de disponibilité*

Les problèmes d’ordonnancement avec indisponibilités des machines, liés en particulier aux problèmes de pannes et maintenances, ont été largement étudiés dans la littérature ; voir [4] pour un état de l’art. Toutefois, dans l’industrie, d’autres types d’indisponibilités peuvent apparaître. Nous considérons ici le problème où la machine est continuellement disponible mais requiert des ressources externes pour commencer et terminer chaque tâche. Ces ressources peuvent être indisponibles, par exemple parce qu’elles sont partagées. Ce problème apparaît notamment dans l’industrie chimique [3] où un chimiste doit préparer les expériences, les lancer et également les terminer (arrêter les réactions et analyser les résultats). Les réactions peuvent durer plusieurs jours ou semaines et l’intervention humaine n’est nécessaire qu’au lancement et à la terminaison. Les chimistes ont néanmoins des congés et des week-ends pendant lesquels ils ne peuvent ni lancer, ni terminer des expériences. L’objectif est d’ordonnancer les tâches (expériences) sur une machine pour minimiser la durée totale sans qu’aucune tâche ne commence, ni ne termine, lorsque l’opérateur est indisponible.

Brauner *et al.* [2] et Rapine *et al.* [7] ont étudié ce problème avec des périodes d’indisponibilités (aucune tâche ne peut commencer ni terminer durant ces périodes). Ils ont montré que la minimisation de la durée de l’ordonnancement est un problème NP-difficile, même en présence d’une seule période d’indisponibilité. Ils ont également proposé des résultats d’inapproximabilité.

Billaut et Sourd [1] ont étudié en premier le cas où les indisponibilités sont un ensemble fini d’instant, en considérant uniquement comme indisponibilité des instants de débuts interdits. Rapine et Brauner [6] ont généralisé leurs résultats à des instants interdits de début et de fin des tâches. Ils ont montré que le problème de minimisation de la durée de l’ordonnancement peut être résolu en temps polynomial lorsque le nombre d’instant interdits K est inférieur strictement au nombre N de types de tâches, un type représentant toutes les tâches de même durée. Une telle instance est dite de *forte diversité*.

Nous montrons ici que pour K fixé, si $K = N$, le problème peut être résolu en temps constant même en considérant un codage *high multiplicity* des instances. Un tel codage décrit chaque type de tâches, en donnant sa durée et sa multiplicité, c’est à dire le nombre de tâches de ce type. En particulier, pour un ordonnancement, donner les dates de début de toutes les tâches est alors exponentiel en la taille de l’instance. Des précautions particulières pour ne pas énumérer toutes les tâches doivent donc être prises afin de rester polynomial en la taille de l’instance. On cherche ici à savoir s’il existe un ordonnancement sans temps mort (OSM) lorsqu’on a autant d’instant interdits que de types de tâches.

Théorème 1 *Pour K fixé, lorsque $K = N$, s’il existe un ordonnancement sans temps mort, on peut le trouver en temps constant $O(1)$.*

La suite de ce résumé donne les idées de l'algorithme et de sa preuve. On suppose que toutes les opérations arithmétiques de base sont réalisées en temps constant.

Notre algorithme a besoin de résoudre de nombreuses instances de SUBSET-SUM avec multiplicités d'objets et tolérance. Dans ce problème il faut décider l'existence d'un sous-ensemble d'objets tels que leur somme est dans un intervalle donné. Pour résoudre ce problème en *high multiplicity* lorsque le nombre M d'objets différents est fixé, on le transforme en sac à dos avec multiplicités et on utilise l'algorithme de Pisinger [5] dont la complexité est dominée par $O(M + 2^M)$. Pour M fixé, la complexité est donc en temps constant $O(1)$.

Le schéma de la preuve est le suivant : on suppose qu'il existe un OSM ayant une propriété \mathcal{P}_1 . Soit on trouve un tel ordonnancement, soit alors il n'existe pas d'OSM satisfaisant la propriété \mathcal{P}_1 . Dans le second cas on s'intéresse ensuite aux ordonnancements satisfaisant $\overline{\mathcal{P}}_1$ et \mathcal{P}_2 , puis $\overline{\mathcal{P}}_1$, $\overline{\mathcal{P}}_2$ et \mathcal{P}_3 , et ainsi de suite jusqu'à trouver un OSM, ou arriver au point où si aucune des propriétés n'est satisfaite, alors il n'existe pas d'OSM.

On s'intéresse en premier à la propriété \mathcal{P}_1 : "il existe un OSM dans lequel une tâche recouvre tous les instants interdits". Si il existe, on trouve un tel ordonnancement en temps constant en résolvant un problème de SUBSET-SUM avec tolérance. Sinon, dans tout OSM, une des tâches termine entre deux instants interdits. On considère alors la propriété \mathcal{P}_2 : "il existe une solution décomposable en 2 instances de forte diversité". D'après [6], chaque instance peut être résolue en temps constant. On résout ce problème et on s'intéresse ensuite à deux propriétés supplémentaires pour conclure.

L'algorithme utilise plusieurs étapes d'énumération sur les types de tâches pour résoudre les sous-problèmes (il faut remarquer que $N! = K! = O(1)$). La complexité totale de l'algorithme est $O(1)$ et jamais les tâches d'un même type ne sont distinguées ou énumérées.

On a discuté ici des ordonnancements sans temps mort pour $K = N$. Si on cherche à minimiser la durée totale, on résout le problème pour $K = N$ puis, si l'on ne trouve pas de solution, on ajoute des temps morts en ajoutant des tâches au problème qu'on résout de nouveau, jusqu'à obtenir un OSM. Le nombre de temps morts à ajouter est borné par $N + 1$ ([6]). La résolution du problème d'optimisation peut donc également être réalisée en temps constant. Les recherches futures devront nous permettre de généraliser cette approche pour $K > N$.

Références

- [1] J.C. Billaut and F. Sourd. Single machine scheduling with forbidden start times. *4OR : A Quarterly Journal of Operations Research*, 7(1) : 37–50, 2009.
- [2] N. Brauner, G. Finke, V. Lehoux-Lebacque, C. Rapine, H. Kellerer, C. Potts, and V. Strusevich. Operator non-availability periods. *4OR : A Quarterly Journal of Operations Research*, 7(3) : 239–253, 2009.
- [3] V. Lebacque, N. Brauner, B. Celse, G. Finke, C. Rapine Planification d'expériences dans l'industrie chimique, Chap. 1, pp 21-32. Dans : *Les systèmes de production : applications interdisciplinaires et mutations*, sous la direction de J-F. Boujut, D. Llerena, D. Brissaud, Hermès-Lavoisier, 2007.
- [4] C-Y. Lee, Machine scheduling with availability constraints. In : *Leung JYT (ed) Handbook of scheduling : algorithms, models and performance analysis*, Chap 22. Chapman & Hall/CRC, 2004.
- [5] D. Pisinger. A minimal algorithm for the bounded knapsack problem. *INFORMS Journal on Computing*, 12(1) : 75–82, 2000.
- [6] C. Rapine and N. Brauner. Polynomial time algorithms for makespan minimization on one machine with forbidden start and completion times. *Les cahiers Leibniz*, n°181, 2010.
- [7] C. Rapine, N. Brauner, G. Finke, and V. Lebacque. Single machine scheduling with small operator-non-availability periods. *Journal of Scheduling*, pages 1–13, 2012.