Ordonnancement sur une machine avec instants interdits de début et de fin des tâches

Christophe Rapine[1], Nadia Brauner[1]

G-SCOP, 46 avenue Félix Viallet, 38031 Grenoble Cedex, France
{christophe.rapine, nadia.brauner}@g-scop.fr

Mots-Clés : ordonnancement, instants d'indisponibilité, makespan

1 Instants interdits

Nous considérons un problème d'ordonnancement avec des instants d'indisponibilités particuliers, que nous appelons Fse, pour Forbidden Start & End. Pendant ces instants, il est interdit de commencer ou d'achever une tâche. Ces instants interdits diffèrent de la notion classique d'instants d'indisponibilité machine, pendant lesquels aucune opération ne peut avoir lieu sur la ressource : pendant un instant Fse il est en effet possible de continuer à exécuter la tâche courante. Ces instants apparaissent naturellement lorsqu'une ressource additionnelle (par exemple un opérateur) est requise lors du lancement et de la terminaison des tâches sur une machine. Alors que la machine est disponible en continu, l'opérateur est typiquement absent pendant la nuit, les week-ends, ou déjà pris par d'autres activités planifiées. Nous avons rencontré cette situation dans l'industrie chimique, dans le cadre d'une collaboration avec l'Institut Français du Pétrole (IFP). Les tâches correspondaient à des expériences chimiques dont la durée peut s'étaler de 3 jours à 3 semaines. Ces expériences sont menées sur un système automatisé qui contrôle les conditions expérimentales. Cependant l'intervention d'un chimiste est requise au début et à la fin de chaque expérience : au début pour constituer le mélange et lancer l'expérience, à la fin pour stopper la réaction et effectuer l'analyse des résultats, qui doit être faite sans attente.

2 Le problème étudié

Nous nous intéressons à l'ordonnancement d'un ensemble de n tâches indépendantes sur une ressource en présence de k instants interdits Fse de début et de fin. Notre but est de minimiser la date de fin de la dernière tâche, ou makepsan de l'ordonnancement. La préemption des tâches n'est pas autorisée. Ce type d'indisponibilité, liée à une ressource additionnelle requise lors du commencement et de la terminaison des tâches, a été assez peu étudiée dans la littérature. Dans [2, 3] les auteurs considèrent des périodes de temps, appelée périodes ONA, pendant lesquelles aucune tâche ne peut ni commencer ni finir. Ils montrent que minimiser le makespan est un problème NP-difficile pour plus de 2 périodes, même si leur longueur est petite devant la durée des tâches. Le travail le plus proche du notre est celui de Billaut et Sourd [1] : ils considèrent également la minimisation du makespan sur une ressource, en présence de k instants où il est interdit de commencer une tâche. Ils montrent que le problème est NP-difficile et qu'un ordonnancement sans temps mort existe toujours

si au moins $2k(k+1)$ tâches de durées distinctes apparaissent dans l'instance. Il est à noter que dans le cas particulier avec seulement 2 instants interdits, ils établissent que 3 tâches de durée distinctes suffisent à garantir l'existence d'un tel ordonnancement.

3 Existence d'un ordonnancement sans temps mort

Nous généralisons les résultats de Billaut et Sourd en montrant qu'en présence de k instants Fse, il suffit d'avoir $k+1$ durées d'exécution distinctes pour garantir l'existence d'un ordonnancement sans temps mort. Nous appelons une telle instance une instance de forte diversité, de faible diversité dans le cas contraire. Ce résultat est le plus fort possible dans le sens où il est facile de construire des instances avec seulement k durées d'exécution distinctes pour lesquelles un temps mort apparaît nécessairement dans tout ordonnancement valide.

Nous montrons ensuite comment trouver un tel ordonnancement sans temps mort en temps polynomial pour les instance de forte diversité. Notre approche est basée sur une hybridation entre algorithme de liste et échanges locaux. En corollaire, le problème avec un nombre constant d'instants interdits peut être résolu en temps polynomial $\mathcal{O}(n^k)$, en utilisant la même approche par programmation dynamique que dans [1] pour les instances de faible diversité.

4 Version High Multiplicity du problème

Finalement nous nous intéressons à un codage compact des instances, où les tâches ne sont pas codées une à une mais où, pour chaque durée d'exécution, est codée son nombre d'occurrences. Nous montrons que le problème reste polynomial pour les instances de forte diversité, et dérivons un algorithme (fortement) polynomial, qui de plus améliore notre approche pour un codage classique de l'instance.

Références

[1] J.-C. Billaut and F. Sourd. Single machine scheduling with forbidden start times. 4OR - Quarterly Journal of the Belgian, French and Italian Operations Research Societies, 7:37–50, 2009.

[2] N. Brauner, G. Finke, V. Lehoux-Lebacque, C. Rapine, H. Kellerer, C. Potts, and V. Strusevich. Operator non-availability periods. 4OR - Quarterly Journal of the Belgian, French and Italian Operations Research Societies, 7:237–253, 2009.

[3] C. Rapine, N. Brauner, G. Finke, and V. Lehoux-Lebacque. Single machine scheduling with small operator non-availability periods. submitted.