Comparaison de deux algorithmes pour l'ajustement des dates d'échéance pour un problème d'ordonnancement à m machines

1 Descriptif du sujet

On considère le problème d'ordonnancement classique $P|prec, r_i, d_i|\star$ défini par un ensemble de tâches de durée quelconque et des contraintes de précédence à effectuer sur m processeurs identiques. Chaque tâche possède un date d'échéance d_i et une date de disponibilité r_i. Le problème consiste à déterminer si il existe un ordonnancement réalisable, et à le construire le cas échéant.

Ce problème est $\mathcal{NP}$-difficile, y compris dans le cas où les tâches sont de durée 1. Deux classes d'algorithmes ont été développées dans ce cas particulier pour améliorer l'évaluation initiale des dates d'échéance en exprimant des conditions nécessaires sur celles-ci. Ils effecuent alors des modifications des dates d'échéances jusqu'à arriver à un point fixe.

1. L'algorithme de Garey-Johnson [2] (GJ) sélectionne des ensembles de tâches à effectuer obligatoirement dans des intervalles fixés. A partir du volume de ces tâches, il peut déterminer des conditions nécessaires sur les dates d'échéances et les modifier.
2. L'algorithme de Leung, Palem et Pnueli [3](LPP) exprime des conditions sur les dates d'échéance en fonction des contraintes de précédence et utilise l'algorithme de Jackson pour déterminer si ces dates sont cohérentes.

Hanen et Munier [1] ont montré que ces deux algorithmes aboutissent aux mêmes date d'échéances et une étude expérimentale a confirmé que l'algorithme LPP est plus rapide que GJ. Le but de ce stage est d'étudier des extensions de ces deux algorithmes pour le cas où les tâches sont de durée quelconque, préemptives ou non. Ce problème, bien que central en théorie de l'ordonnancement, a été peu étudié. La question est alors de comparer deux extensions possibles :

1. D'un point de vue expérimental, il s'agit de comparer la complexité de ces deux approches et déterminer si ces deux algorithmes convergent vers les mêmes valeurs. La réduction des intervalles d'exécution des tâches à un impact sur le parrallèlisme du problème. Cet impact peut-être mesuré en considérant le nombre maximum de tâches exécutables à un instant donné en fonction des intervalles obtenus.
2. D'un point de vue plus théorique, il s'agit de déterminer si les conditions nécessaires exprimées par ces deux approches restent équivalentes pour des tâches de durée différentes.

Dans un premier temps, il s'agit d'implémenter ces deux extensions et de tester leur performance sur des instances aléatoires dans le but de comparer leur vitesse de convergence et la qualité des dates d'échéances obtenues. Dans un second temps, nous étudierons des cas particuliers en fonction de la structure du graphe de précédence et de la durée des tâches pour comparer de manière théorique les deux approches testées.

Les résultats de cette étude pourront être utilisés dans plusieurs contexte. Tout dabord, l'expression des conditions nécessaires et des algorithmes pour les mettre en oeuvre peuvent être utilisés comme un pré-traitement pour des méthodes de résolution exactes ou des algorithmes paramétrés du problème d'ordonnancement. Ces conditions peuvent également servir de base pour l'étude théorique d'algorithmes approchés avec ratio d'approximation pour ce problème.

2 Conditions du stage

Le stage aura lieu en 2020 au LIP6 sur le campus Pierre et Marie Curie (4 place Jussieu, 75252 Paris) et sera co-encadré par Claire Hanen et Alix Munier-Kordon pour une durée de 6 mois. La gratification prévue est de 554.4 Euros par mois.

Le candidat recherché doit être en Master 2 de recherche opérationnelle, de mathématiques appliquées, d'informatique ou une école d'ingénieur avec un cursus en informatique. Il doit avoir de solides connaissances en programmation et en algorithmique.

Pour postuler, envoyez par mail aux deux encadrants Claire.Hanen@lip6.fr et Alix.Munier@lip6.fr un CV avec au minimum les notes de $M1$ et de $M2$.

Références

[1] Aurélien Carlier, Claire Hanen, and Alix Munier Kordon. The equivalence of two classical list scheduling algorithms for dependent typed tasks with release dates, due dates and precedence delays. *J. Scheduling*, 20(3) :303–311, 2017.

[2] M. R. Garey and D. S. Johnson. Two-processor scheduling with start-time and deadlines. *SIAM Journal on Computing*, 6 :416–426, 1977.

[3] Allen Leung, Krishna V. Palem, and Amir Pnueli. Scheduling time-constrained instructions on pipelined processors. *ACM Trans. Program. Lang. Syst.*, 23 :73–103, January 2001.