Placement de tâches par des agents individualistes sur des machines distribuées

Johanne Cohen[1], Fanny Pascual[2]

[1] PRiSM-CNRS ; Université de Versailles Saint Quentin ; johanne.cohen@prism.uvsq.fr
[2] LIP6-CNRS ; Université Pierre et Marie Curie (Paris 6) ; fanny.pascual@lip6.fr

Mots-Clés : *Ordonnancement, théorie des jeux algorithmique, équilibres de Nash approchés*

Introduction. On s'intéresse à l'ordonnancement de tâches sur des machines partagées entre plusieurs personnes individualistes, que nous nommerons agents. Nous considérons en particulier le cas où il y a deux agents, chaque agent possédant un ensemble de tâches de différentes longueurs (la longueur d'une tâche étant la durée dont elle a besoin pour être exécutée). Ces agents ont à leur disposition un ensemble de m machines identiques, sur lesquelles ils exécutent leurs tâches. Le but de chaque agent est de minimiser la date de fin moyenne de ses tâches. Chaque agent choisit de placer chacune de ses tâches sur la machine de son choix. La machine a alors une politique qui lui indique dans quel ordre exécuter les tâches qui lui sont affectées. Les machines étant distribuées, ces politiques sont locales : l'ordonnancement des tâches d'une machine ne peut dépendre que des tâches affectées à cette machine, et non des tâches présentes sur les autres machines. Connaissant les tâches de l'autre agent et la politique des machines, chaque agent place donc ses tâches de façon à maximiser la date de fin moyenne de ses propres tâches.
Notre but est d'étudier, selon la politique des machines, la situation obtenue lors de l'affectation des tâches par les agents, dans le pire des cas : existe-t'il toujours un équilibre de Nash pur, c'est-à-dire une affectation des tâches aux machines pour laquelle chaque agent n'a pas intérêt à changer ses tâches de machine ? Si ce n'est pas le cas, existe-t'il toujours des équilibres de Nash α-approchés, c'est-à-dire une situation dans laquelle aucun agent ne diminuera la date de fin moyenne de ses tâches d'un facteur supérieur à α ? Ici α représente la stabilité des solutions obtenues : plus α sera petit, moins les agents auront intérêt à changer leurs tâches de machine.

Ce travail est une extension des travaux de Christodoulou et al. [1] qui introduisent la notion de mécanisme de coordination. Un mécanisme de coordination est un ensemble de politiques qui indiquent à chaque machine, étant donné les tâches qu'elle a à exécuter, l'ordre d'exécution des tâches. Ces politiques peuvent aussi éventuellement introduire des temps d'inactivités entre les tâches. Dans leurs travaux, Christodoulou et al., puis Immorlica et al. [2] considèrent un ensemble de n agents, chaque agent possédant une tâche, et souhaitant que cette tâche soit exécutée le plus tôt possible. Les auteurs étudient les performances de plusieurs politiques classiques :

- la politique SPT (pour *Shortest Processing Time first*), dans laquelle les tâches sont ordonnancées de la plus petite à la plus grande,
- la politique LPT (pour *Largest Processing Time first*), dans laquelle les tâches sont ordonnancées de la plus grande à la plus petite
- la politique Random, dans laquelle les tâches sont ordonnancées dans un ordre aléatoire.

Nous étudions également les performances de ces politiques classiques pour notre problème.

Proposition 1 *Si toutes les machines ont la même politique déterministe, et si cette politique n'introduit pas de temps mort entre les tâches, alors il existe toujours une instance pour laquelle il n'existe pas d'équilibre de Nash.*

Preuve : Notons $i \prec j$ si la tâche i est placée avant la tâche j quand la machine doit ordonnancer ces deux seules tâches i et j. On considère m tâches $\{a_1, \ldots, a_m\}$ appartenant à l'agent A et une tâche b appartenant à l'agent B. Ces tâches sont telles que $a_1 \prec b$ et $b \prec a_2$. Ainsi l'agent B fera en sorte de placer b sur la même machine que a_2 (ou tout autre tâche a_i telle que $b \prec a_i$), afin que sa tâche soit en première position, alors que l'agent A choisira de faire le contraire : il placera ses tâches de façons à ce que a_1 (ou une autre tâche précédent b) soit avec la tâche b et les autres tâches seules sur leur machine. Il n'existe donc pas d'équilibre de Nash pur pour cette instance. □

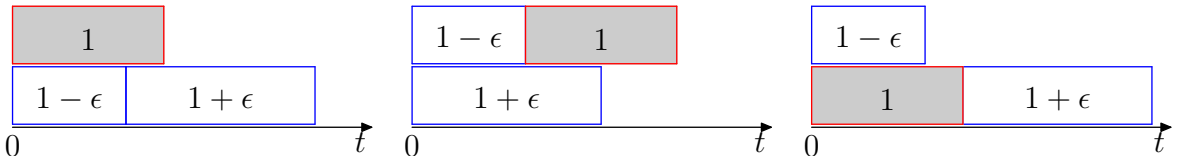

FIG. 1 – Exemple dans lequel les machines utilisent la politique SPT et où il n'existe pas d'équilibre de Nash pur : quelque soit l'ordonnancement proposé, un agent peut toujours diminuer la date de fin moyenne de ses tâches en les plaçant différemment. L'agent A a deux tâches de longueur $1-\epsilon$ et $1+\epsilon$ et l'agent B a une tâche de longueur 1.

Les politiques SPT et LPT rentrant dans le cadre de cette proposition, on en déduit qu'elles ne peuvent pas toujours engendrer des équilibres de Nash. La figure 1 illustre cette proposition quand deux machines ont pour politique SPT. La question est alors de savoir si certaines politiques peuvent tout de même engendrer des solutions assez stables, i.e. des équilibres de Nash α-approchés, avec α aussi petit que possible. Pour la politiques LPT, ce n'est pas le cas :

Proposition 2 *Si toutes les machines utilisent la politique LPT, alors il n'existe pas toujours d'équilibre de Nash α-approché, pour tout α fixé.*

Preuve : Il n'existe pas d'équilibre de Nash α-approché pour l'instance de la figure 2. □

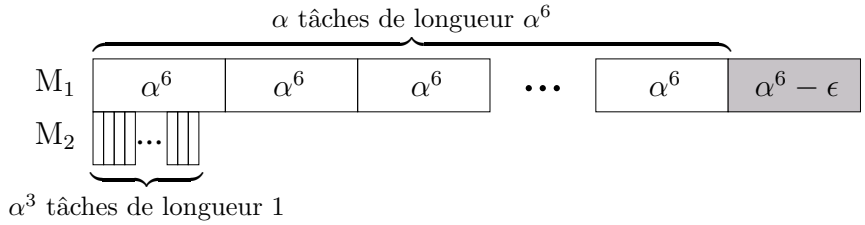

FIG. 2 – L'agent A a la tâche de longueur $\alpha^6 - \epsilon$, et l'agent B les autres tâches.

Nous obtenons le même résultat avec la politique Random : pour tout α il existe une instance pour laquelle il n'existe pas d'équilibre de Nash α-approché. Quand la politique des machines est SPT nous avons montré qu'il existe toujours des équilibres de Nash 3-approché. Une question intéressante serait de savoir s'il existe une politique des machines qui pourrait toujours engendrer des équilibres de Nash. Nous pourrions en particulier étudier le cas où les machines ont des politiques différentes.

Références

[1] G. Christodoulou, E. Koutsoupias, A. Nanavati. Coordination mechanisms. *Theoretical Computer Science*, 410(36):3327–3336, 2009.

[2] N. Immorlica, L. Li, V. Mirrokni, A. Schulz. Coordination mechanisms for selfish scheduling. *Theoretical Computer Science*, 410(17):1589–1598, 2009.