

Performance des algorithmes à véracité garantie pour l'ordonnement de tâches individualistes

G. Christodoulou¹, L. Gourvès², et F. Pascual³

¹ National and Kapodistrian University of Athens,
Dept. of Informatics and Telecommunications,
Panepistimiopolis, Ilissia, Athens 15784
gchristo@di.uoa.gr

² LAMSADE - Université Paris Dauphine, UMR 7024
Place du Maréchal de Lattre de Tassigny 75775 Paris Cedex 16
laurent.gourves@lamsade.dauphine.fr

³ Équipe MOAIS (CNRS-INRIA-INPG-UJF),
Laboratoire d'Informatique de Grenoble, 38330 Montbonnot Saint Martin
fanny.pascual@imag.fr

1 Introduction

Le problème que nous considérons se place dans le domaine de l'ordonnement d'agents individualistes (*selfish scheduling*). C'est une variante du problème $P||C_{max}$ qui peut être énoncé comme suit : un protocole est en charge d'ordonner n tâches sur m machines identiques sans connaître directement les longueurs (ou durées d'exécution) des tâches. Le protocole obtient ces informations d'agents qui contrôlent les tâches et sont les seuls à connaître les véritables valeurs (une tâche i a pour véritable longueur l_i). Le but de chaque agent est de minimiser la date de fin d'exécution de sa tâche tandis que le protocole cherche à minimiser la plus grande date de fin d'exécution (le *makespan*).

Dans ce contexte, un agent ayant connaissance de la politique d'ordonnement du protocole ainsi que des valeurs déclarées par les autres peut délibérément communiquer une longueur erronée si ceci lui permet d'optimiser son propre objectif. Muni d'informations pouvant être fausses, le protocole ne peut alors concrètement pas atteindre son objectif. On dit d'un algorithme (ou protocole) qu'il est à *véracité garantie* s'il met les agents dans une situation où mentir n'a pas d'intérêt (ils ne peuvent pas diminuer la date à laquelle leur tâche sera terminée en mentant sur la longueur de celle-ci). Nous abordons la question suivante : Quelle performance un algorithme à véracité garantie peut-il atteindre ? Cette question a été posée à la fois dans le cadre d'un système centralisé (le protocole collecte les valeurs puis définit un ordonnancement sur l'ensemble des m machines), mais aussi pour un système distribué [2] (chaque machine est munie d'une politique d'ordonnement qui ne dépend que des tâches qui lui sont allouées, et les agents décident eux-mêmes sur quelle machine ils souhaitent que leur tâche soit exécutée). On parle d'algorithme dans un système centralisé et de *mécanisme de coordination* dans un système distribué.

Pour les deux systèmes (centralisé ou distribué), les agents sont perçus comme les joueurs d'un jeu non-coopératif dont le protocole dicte les règles. On se propose d'évaluer à l'aide d'une analyse de type pire cas les performances des protocoles à véracité garantie. Plus précisément, on souhaite borner inférieurement et supérieurement le rapport entre le makespan dans un *équilibre de Nash* (situation dans laquelle aucun agent ne peut unilatéralement changer de stratégie – e.g. déclarer une longueur différente – et améliorer son objectif) et le makespan optimal. On parle alors de rapport d'approximation pour le système centralisé et de *prix de l'anarchie* pour le cas distribué [4].

Le domaine qui consiste à concevoir des algorithmes à véracité garantie (*mechanism design*) a mené à de nombreux travaux, en économie principalement et plus récemment en informatique, à la suite notamment de [6]. Il est courant dans ce domaine de considérer des mécanismes de paiement qui incitent les agents à déclarer la vérité. Nous nous restreignons ici aux techniques algorithmiques ne faisant pas appel à ce type d'outil.

2 Deux modèles d'exécution

Lorsque l'agent qui contrôle la tâche i déclare b_i , il est considéré comme possédant une tâche de longueur b_i par le protocole. On peut supposer qu'aucun agent n'a intérêt à déclarer $b_i < l_i$ sinon une durée trop petite pour que sa tâche soit complètement exécutée lui est allouée. Un agent ne ment alors que dans une seule direction en augmentant artificiellement la durée de sa tâche (i.e. $b_i \geq l_i$). Nous considérons par la suite les deux *modèles d'exécution* suivants :

- **modèle fort** : si l'agent qui contrôle la tâche i déclare $b_i \geq l_i$ alors il sera considéré comme ayant une tâche de longueur b_i par le protocole mais il obtiendra son résultat l_i unités de temps après le début de l'exécution de sa tâche.
- **modèle souple** : si l'agent qui contrôle la tâche i déclare $b_i \geq l_i$ alors il sera considéré comme ayant une tâche de longueur b_i par le protocole et obtiendra son résultat b_i unités de temps après le début de l'exécution de sa tâche.

3 Système centralisé

Pour les deux modèles d'exécution énoncés ci-dessus, l'algorithme qui exécute de manière gloutonne les tâches de la plus petite à la plus grande, est à véricité garantie : aucun agent ne peut en effet espérer avancer le début d'exécution de sa tâche en augmentant artificiellement sa longueur. Cet algorithme déterministe a un rapport d'approximation égal à $2 - \frac{1}{m}$ [3]. Existe-t'il un meilleur algorithme à véricité garantie ?

Pour le modèle d'exécution fort, il existe également un algorithme probabiliste à véricité garantie $2 - \frac{1}{m+1}(\frac{5}{3} + \frac{1}{3m})$ -approché [1]. Nous apportons la preuve que le rapport $2 - \frac{1}{m}$ ne peut être amélioré dans le cas d'un algorithme déterministe, tandis qu'une borne inférieure de $\frac{3}{2} - \frac{1}{2m}$ sur le rapport d'approximation d'un algorithme probabiliste à véricité garantie existe.

Pour le modèle d'exécution souple, il existe un algorithme probabiliste à véricité garantie qui est optimal (néanmoins exponentiel) [5]. Notre contribution consiste à améliorer le rapport pour le cas déterministe (on obtient un rapport $\frac{4}{3} - \frac{1}{3m}$) et à proposer des bornes inférieures : $1 + \frac{\sqrt{105}-9}{12}$ si $m = 2$ et $\frac{7}{6}$ si $m \geq 3$.

4 Système distribué

D'un point de vue positif, il existe un mécanisme de coordination déterministe dont le prix de l'anarchie est égal à $2 - \frac{1}{m}$ pour les deux modèles d'exécution [2]. On peut observer que les bornes inférieures que nous avons obtenues pour le modèle d'exécution fort dans le système d'exécution centralisé peuvent être réutilisées pour le système d'exécution distribué.

Pour le modèle d'exécution souple, nous proposons des bornes inférieures à l'aide de techniques différentes où seuls les équilibres de Nash purs⁴ sont considérés, à savoir $\frac{1+\sqrt{17}}{4}$ (resp. $1 + \frac{\sqrt{13}-3}{4}$) pour un mécanisme de coordination déterministe (resp. probabiliste).

Références

1. E. Angel, E. Bampis and F. Pascual. *Truthful Algorithms for Scheduling Selfish Tasks on Parallel Machines*. In Proc. of WINE 2005, LNCS 3828, pp. 698-707, 2005.
2. G. Christodoulou, E. Koutsoupias and A. Nanavati. *Coordination Mechanisms*. In Proc. of ICALP 2004, LNCS 3142, pp. 345-357, 2004.
3. R. Graham. *Bounds on multiprocessor timing anomalies*. In SIAM Jr. on Appl. Math. 17(2), pp. 416-429, 1969.
4. E. Koutsoupias and C. Papadimitriou. *Worst Case Equilibria*. In Proc. of STACS 1999, LNCS 1563, pp. 404-413, 1999.
5. F. Pascual. *Optimisation dans les réseaux : de l'approximation polynomiale à la théorie des jeux*, Thèse de doctorat, Université d'Évry, 2006.
6. N. Nisan, A. Ronen. *Algorithmic mechanism design*. In Proc. of STOC 1999, 129-140.

⁴ Tout agent décide de manière déterministe sur quelle machine il souhaite que sa tâche soit exécutée.