HAL open science

Ordonnancement Deux-Agents avec Augmentation de Ressources

Vincent Fagnon, Giorgio Lucarelli, Clement Mommessin, Denis Trystram

To cite this version:

Vincent Fagnon, Giorgio Lucarelli, Clement Mommessin, Denis Trystram. Ordonnancement Deux-Agents avec Augmentation de Ressources. 23ème congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision, INSA Lyon, Feb 2022, Villeurbanne - Lyon, France. hal-03595369

HAL Id: hal-03595369

https://hal.science/hal-03595369v1

Submitted on 3 Mar 2022

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.

Distributed under a Creative Commons Attribution 4.0 International License

Ordonnancement Deux-Agents avec Augmentation de Ressources

Vincent Fagnon[1], Giorgio Lucarelli[2], Clément Mommessin[3], Denis Trystram[1]

[1] Univ. Grenoble Alpes, CNRS, INRIA, Grenoble INP, LIG, Grenoble, France
{vincent.fagnon,denis.trystram}@inria.fr
[2] LCOMS, University de Lorraine, Metz
giorgio.lucarelli@univ-lorraine.fr
[3] University of Leeds, Leeds, UK
c.mommessin@leeds.ac.uk

Mots-clés : *Optimisation, Dual Fitting, Analyse Compétitive, Augmentation de ressources*

1 Introduction

La plupart des données produites par les capteurs de l'Internet des Objets n'ont d'intérêt que pour leur environnement local et ont une durée de vie courte. Dans ce travail, nous ciblons un système informatique composé de plusieurs unités de calcul et connecté à certains capteurs. Lorsqu'aucune donnée n'est analysée sur ce système, les unités de calcul peuvent être utilisées par d'autres calculs utiles, de la même manière qu'avec le volonteer computing [4]. De manière informelle, il existe deux types de tâches à exécuter, et chaque type est associé à un objectif distinct.

1.1 Description du problème

Les caractéristiques de ces deux types de tâches (appelons-les agents) sont les suivantes :

— Le premier agent gère les tâches ("locales") générées par les capteurs, qui doivent être traitées le plus rapidement possible. Leur temps de traitement est connu au moment de leur release date. Mais les release date sont inconnues. En raison de leur localité et de leur taille (petite en pratique), la préemption de ces tâches est autorisée. L'objectif cible est de minimiser leur Flowtime total.

— Le second agent ("global") correspond aux tâches externes soumises par batch. Leur temps de traitement est connu. Ces tâches ne peuvent pas être préemptées. Cependant, le rejet de certaines de ces tâches lors de leur exécution peut être autorisé si cela est nécessaire pour conserver une bonne performance du système. Intuitivement, une tâche globale rejetée sera soumise à nouveau dans un batch ultérieur dans le même ou dans un autre système ciblé. L'objectif est de réaliser ces tâches dans un délai raisonnable, avant une échéance commune qui est fixée à la charge totale de calcul pendant le batch.

1.2 Contributions

Le problème de l'agent local seul peut être résolu de manière optimale sur une seule machine (SRPT), mais il est difficile à approximer avec plusieurs machines [9]. Cela conduit à analyser le problème dans le contexte d'*augmentation de ressources*, où plus de puissance est donnée à l'algorithme pour obtenir des garanties de performance à facteur constant [7, 8, 12]. Avec l'introduction des tâches de l'agent global, nous montrons qu'il est difficile d'approcher l'objectif du FlowTime pour les tâches locales, même en autorisant l'augmentation des ressources. Plus précisément, tout algorithme online qui utilise une $(1 + \epsilon_s)$-speed devrait rejeter au plus k

tâches globales pour atteindre un ratio compétitif en $O\left((1-\frac{2k}{n^G})\mathcal{W}\right)$, où $\mathcal{W}$ est le ratio de la charges de travail globales sur la charge de travail local, et n^G le nombre de tâches globales.

D'autre part, nous proposons un algorithme pour résoudre le problème d'ordonnancement à deux agents, en considérant à la fois rejection et augmentation de vitesse. Nous analysons son ratio de compétitivité en utilisant l'approche du Dual-Fitting [3]. En particulier nous montrons que, dans le cas général, l'algorithme est $(1+\epsilon_s)$-speed et $O(\frac{\mathcal{W}}{\epsilon_r \cdot \epsilon_s})$-compétitif, où ϵ_s et ϵ_r sont les paramètres de l'augmentation des ressources (respectivement pour la vitesse et le rejet).

2 Related Work

Agnetis et al. [2] présentent un problème deux-agents où chaque agent vise à minimiser son propre objectif ; en maintenant le second objectif sous un seuil ; ou en trouvant l'ensemble des couples non dominés sur le front de Pareto. Ce travail a été étendu [1] à d'autres problèmes multi-agent sur plusieurs machines. De nombreuses variantes existent notamment des dates d'échéance [6, 13], des latences à chaque *"changement"* d'agent [10], plusieurs agents sur une seule machine en minimisant des combinaisons linéaires des objectifs de chaque agents [5].
Le problème étudié ici peut être considéré comme une extension du problème de minimisation du Flowtime total de tâches online sur des machines parallèles pour l'agent local [11], en ajoutant la contrainte supplémentaire du respect de la deadline pour l'agent global.

Références

[1] Alessandro Agnetis, Jean-Charles Billaut, Stanislaw Gawiejnowicz, Dario Pacciarelli, and Ameur Soukhal. *Multiagent Scheduling - Models and Algorithms*. Springer, 2014.

[2] Alessandro Agnetis, Pitu B. Mirchandani, Dario Pacciarelli, and Andrea Pacifici. Scheduling problems with two competing agents. *Operations Research*, 52(2) :229–242, 2004.

[3] S Anand, Naveen Garg, and Amit Kumar. Resource augmentation for weighted flow-time explained by dual fitting. In *SODA*. SIAM, 2012.

[4] David P Anderson. Boinc : A system for public-resource computing and storage. In *proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, 2004.

[5] Kenneth R. Baker and J. Cole Smith. A multiple-criterion model for machine scheduling. *J. Scheduling*, 6(1) :7–16, 2003.

[6] C. Cheng, S. Li, K. Ying, and Y. Liu. Scheduling jobs of two competing agents on a single machine. *IEEE Access*, 7, 2019.

[7] Anamitra Roy Choudhury, Syamantak Das, Naveen Garg, and Amit Kumar. Rejecting jobs to minimize load and maximum flow-time. *JCSS*, 2018.

[8] Bala Kalyanasundaram and Kirk Pruhs. Speed is as powerful as clairvoyance. *Journal of the ACM (JACM)*, 47(4) :617–643, 2000.

[9] Stefano Leonardi and Danny Raz. Approximating total flow time on parallel machines. *Journal of Computer and System Sciences*, 73(6) :875–891, 2007.

[10] Shi-Sheng Li, Ren-Xia Chen, and Qi Feng. Scheduling two job families on a single machine with two competitive agents. *J. Comb. Optim.*, 32(3) :784–799, 2016.

[11] Giorgio Lucarelli, Nguyen Kim Thang, Abhinav Srivastav, and Denis Trystram. Online non-preemptive scheduling in a resource augmentation model based on duality. *ESA*, 2016.

[12] Cynthia A Phillips, Cliff Stein, Eric Torng, and Joel Wein. Optimal time-critical scheduling via resource augmentation. In *STOC*, 1997.

[13] Yunqiang Yin, Dujuan Wang, and T.C.Edwin Cheng. *Due Date-Related Scheduling with Two Agents - Models and Algorithms*. Springer, 2020.