Information Processing Letters 113 (2013) 465–469

Contents lists available at SciVerse ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Inefficiency of Nash equilibria with parallel processing policy

Long Wan[a,*], Xiaofang Deng[b], Zhiyi Tan[c,1]

[a] *Department of Mathematics, Zhejiang University, Hangzhou, 310027, PR China*
[b] *Software School, Jiangxi Normal University, Nanchang, 330022, PR China*
[c] *Department of Mathematics, State Key Lab of CAD, Zhejiang University, Hangzhou, 310027, PR China*

ARTICLE INFO

Article history:
Received 26 September 2012
Received in revised form 6 February 2013
Accepted 14 March 2013
Available online 27 March 2013
Communicated by F.Y.L. Chin

Keywords:
Parallel processing
Nash equilibrium
Price of anarchy

ABSTRACT

In this paper, we revisit the coordination mechanism of parallel processing policy introduced in [L. Yu, K. She, H. Gong, C. Yu, Price of anarchy in parallel processing, Information Processing Letters 110 (8–9) (2010) 288–293]. For both the problem of minimizing makespan and the machine covering problem, we give the analysis of price of anarchy with this new mechanism. In the first problem, we point out an error in the original paper and provide a correct instance. Moreover, we show the exact PoA for identical and uniform machines. In the second problem, we obtain new results for several scheduling models.

1. Introduction

In a selfish job scheduling game, there are m machines and n jobs. The processing time of job i on machine j is p_{ij}. Each job belongs to a selfish agent who is concerned only with the cost which is defined as the completion time of his/her own job. We focus on the pure strategy case where each agent is entitled to select one specific machine to process his/her job. On the contrary, the centralized controller prefers a social objective to the system. There are two kinds of social objectives mostly studied: minimizing the makespan and maximizing the machine cover. In the first problem, it aims to minimize the maximum completion time of all machines, which is so-called makespan. In the second problem, the objective is to maximize the minimum completion time of all machines. The latter is usually referred to as the machine covering problem by many scheduling researchers [9].

To prevent too much degradation of the social performance to the system, the centralized controller must design appropriate mechanisms to affect the agents' selfish behavior. There are several approaches in mechanism design, such as using tolls [4,7,12], Stackelberg Strategy [3,18,21], coordination mechanism [2,6], etc. Among all these approaches, the coordination mechanism, which is first studied by Christodoulou, Koutsoupias and Nanavati [6], may be of the most interest in job scheduling games. This approach is to design local policies that can assign cost to each agent for each strategy combination, and the cost is a function of the agents who choose this strategy combination. In job scheduling games, coordination mechanism usually contains a set of scheduling policies, one for each machine. Let's first review some classical and well studied coordination mechanisms: Makespan, ShortestFirst, LongestFirst and Randomized. Makespan mechanism is the one to force jobs on the same machine completed at the same time [8,10]. In ShortestFirst and LongestFirst mechanisms, jobs on each machine are processed in the non-decreasing and non-increasing order of the processing times, respectively [6,17]. Randomized mechanism is to process jobs randomly [8,13,17]. In a recent research [24], Yu et al. formalized a new and natural coordination mechanism from the computer processing field, namely Parallel Processing Policy. In this mechanism, each machine

* Corresponding author.
E-mail address: cocu3328@163.com (L. Wan).

1 Supported by the National Natural Science Foundation of China (10971191, 60021201) and Fundamental Research Funds for the Central Universities (2010QNA3040).

Table 1
The price of anarchy for four different policies and scheduling problems.

	Makespan	ShortestFirst	LongestFirst	Randomized
$Pm\|C_{\max}$	$\frac{2m}{m+1}$ [22,11]	$\frac{2m-1}{m}$ [14]	$\frac{4}{3}-\frac{1}{3m}$ [15,6]	$\frac{2m}{m+1}$ [22,11]
$Qm\|C_{\max}$	$\Theta(\frac{\log m}{\log\log m})$ [8]	$\Theta(\log m)$ [1]	$1.54<\rho<1.5773$ [20]	$\Theta(\frac{\log m}{\log\log m})$ [8]
$Rm\|C_{\max}$	Unbounded [22]	m [16,5]	Unbounded [17]	$\Theta(m)$ [17]

processes all its currently unfinished jobs in parallel. More concretely, the machine divides the time into tiny time slots with equal length and assigns one time slot to each unfinished job in the round robin way.

With certain mechanism, it is interesting to know the properties of the solutions resulting from the agents' selfish behavior, in particular, to estimate the price of anarchy (PoA in short). The PoA, introduced by Koutsoupias and Papadimitriou [19], is defined as the performance ratio between the social objective of the worst Nash equilibrium and the social objective of an optimal solution. For convenience, we use P, Q and R to represent identical machines, related machines, unrelated machines, respectively. By three-field notation, we use $Pm\|C_{\max}$ to denote the scheduling problem on m identical machines with the objective to minimize the makespan, $Q_2\|C_{\min}$ to denote the scheduling problem on two uniform machines with the objective to maximize the machine cover, etc. For several classical coordination mechanisms, we summarize the known upper and lower bounds of PoA in Table 1 (see also [24,17]). For parallel processing policy, Yu et al. showed pure Nash equilibriums always exist. And they gave both upper and lower bounds of PoA in various scheduling models [24]. Note all these results are related to the makespan minimization problem, there are few results concerning the machine covering problem. To the authors' knowledge, the first research concerning the latter goal is due to Epstein [9], who gave an excellent analysis for the machine covering problems on uniform and identical machines with Makespan mechanism. Tan et al. [23] showed the exact PoA for the machine covering problems on two uniform machines with Makespan mechanism.

In this paper, we consider both the makespan minimization problem and the machine covering problem with parallel processing policy. For the first problem, we point out an error in [24] and provide a correct proof. Moreover, we show that the exact PoA is $2-\frac{1}{m}$ for identical machines and $\min\{\frac{1+s}{s}, \frac{1+2s}{1+s}\}$ for two uniform machines, where $s \geqslant 1$ is the ratio between the speeds of the two machines. For the second problem, exact PoA is obtained for several scheduling models.

The paper is organized as follows. In Section 2 we give some useful lemmas. Section 3 and Section 4 study the makespan minimization problem and the machine covering problem, respectively.

2. Preliminaries

In parallel processing, suppose there are k jobs scheduled on machine j with the processing time ordered by $p_{1j} \leqslant p_{2j} \leqslant \cdots \leqslant p_{kj}$, then the t-th job's completion time C_t can be calculated according to

$$C_t = (k-t)p_{tj} + \sum_{i=1}^{t} p_{ij}. \tag{1}$$

Clearly, $C_1 \leqslant C_2 \leqslant \cdots \leqslant C_k$ and $C_k = \sum_{i=1}^{k} p_{ij}$. We define the completion time of a machine as the total processing time of jobs on it. Thus we have

Lemma 2.1. *With parallel processing policy, the completion time of a machine is equal to the cost of the largest job on it.*

Let L_j be the completion time of machine j, and $[j]$ be the largest job on j, then we say an assignment satisfies property P if:

$$L_a + p_{[b]a} \geqslant L_b, \quad \text{for any machines } a \text{ and } b.$$

Since each job in a Nash equilibrium prefers to process on its original machine, therefore this property is valid in any Nash equilibrium.

Lemma 2.2. *With parallel processing policy, any Nash equilibrium satisfies property P.*

In the following discussion, we simply denote σ as a Nash equilibrium, and let L^{σ} and L^* be the social objective value generated by σ and the optimal solution, respectively.

3. Makespan minimization problem

3.1. PoA in $Pm\|C_{\max}$

In identical scheduling model, the processing times of a job on different machines are equal, i.e., $p_{ij} = p_i$ for all $j = 1, 2, \ldots, m$. In [24], Yu et al. have proved the PoA in $Pm\|C_{\max}$ is at most $2-\frac{1}{m}$. Below we close the analysis by constructing a simple instance that achieves the same bound. There are $m(m-1)$ small jobs with identical processing time of 1 and one large job with processing time m. In an optimal solution, we assign the single large job independently to one machine and all the other jobs to the rest machines as evenly as possible. Hence, the minimum makespan equals m. However, the worst Nash equilibrium might assign all the small jobs evenly to m machines and leave the large job to any of the machines. Consequently, the generated makespan is $2m-1$. It's clear that the large job will never benefit from selecting a different machine and each small job would be completed no earlier than it is on the original machine as well. In other words, the assignment forms a Nash equilibrium. Hence, the PoA in $Pm\|C_{\max}$ must be no smaller than $\frac{2m-1}{m}$. Together with the result given by Yu et al. [24], we have