Theoretical Computer Science 572 (2015) 58–65

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Online scheduling with equal processing times and machine eligibility constraints

Jia Xu, Zhaohui Liu *

Department of Mathematics, East China University of Science and Technology, Shanghai 200237, China

ARTICLE INFO

Article history:
Received 15 September 2014
Received in revised form 6 January 2015
Accepted 14 January 2015
Available online 19 January 2015
Communicated by D.-Z. Du

Keywords:
Scheduling
Parallel machine
Eligibility constraint
Online algorithm

ABSTRACT

We consider the online scheduling problem on m parallel machines with eligibility constraints. The jobs arrive over time and have equal processing times. The objective is to minimize the makespan. We develop optimal deterministic online algorithms for the nested processing set case and the inclusive processing set case with an arbitrary number of machines, as well as the tree-like processing set case with three machines.

1. Introduction

In this paper we consider the problem of scheduling n jobs on m parallel machines with eligibility constraints. Let $J = \{J_1, J_2, \ldots, J_n\}$ be the set of jobs and $M = \{M_1, M_2, \ldots, M_m\}$ be the set of machines. Every job J_j is associated with a release time r_j, a processing time p_j, and a processing set $\mathcal{M}_j \subseteq M$, which mean the job can only be processed at or after time r_j and on the machines in $\mathcal{M}_j$, and its processing takes p_j time units. We study this scheduling problem in the online setting. Then, the information of any job is available only after its being released, even about its existence. But when a job appears, we have the option of scheduling it immediately or postponing its scheduling till some later time. Our goal is to find a schedule that minimizes the makespan $C_{\max}$, i.e., the maximum completion time of the jobs. Using the 3-field notation of Graham et al. [3], we denote the problem as $P|\mathcal{M}_j, r_j, online|C_{\max}$. Specially, we concern ourselves with the problem with equal processing times, i.e., $P|\mathcal{M}_j, r_j, p_j = p, online|C_{\max}$.

We consider three types of eligibility constraints in this paper, namely the nested processing sets, the inclusive processing sets, and the tree-like processing sets. In the nested processing set case, for any two jobs J_i and J_j, it holds that either $\mathcal{M}_i \subseteq \mathcal{M}_j$ or $\mathcal{M}_i \supseteq \mathcal{M}_j$ or $\mathcal{M}_i \cap \mathcal{M}_j = \emptyset$. In the inclusive processing set case, for any J_i and J_j, it holds that either $\mathcal{M}_i \subseteq \mathcal{M}_j$ or $\mathcal{M}_i \supseteq \mathcal{M}_j$. Clearly, the inclusive processing set is a special case of the nested processing set. In the literature, the inclusive processing set is also called the GoS processing set. In the tree-like processing set case, we can construct a rooted tree such that each node represents a machine and the processing set of a job consists of the nodes/machines on the unique path from some node to the root of the tree. Fig. 1 shows an example of the tree-like processing sets. Note that each job can be associated with a node of the tree to indicate the processing set. In the example, J_1, J_2, J_3, J_4 are associated with the machine nodes M_2, M_2, M_3, M_6, respectively. Then, $\mathcal{M}_1 = \mathcal{M}_2 = \{M_1, M_2\}$, $\mathcal{M}_3 = \{M_1, M_2, M_3\}$, $\mathcal{M}_4 = \{M_1, M_5, M_6\}$.

* Corresponding author.
E-mail address: zhliu@ecust.edu.cn (Z. Liu).

http://dx.doi.org/10.1016/j.tcs.2015.01.015
0304-3975/

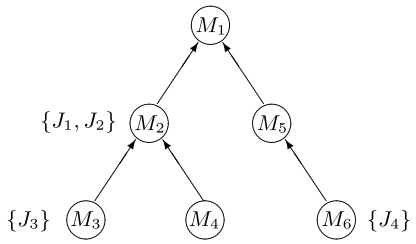

Fig. 1. Tree-like processing set.

We use $\mathcal{M}_j(nested)$, $\mathcal{M}_j(inclusive)$, $\mathcal{M}_j(tree)$ to specify these three eligibility constraints. The inclusive processing set is also the special case of the tree-like processing set.

Scheduling problems with eligibility constraints occur quite often in practice. Glass and Mills [2] gave an application of the nested processing set case for the steam-treatment process in a food processing plant. Ou et al. [7] described an application of the inclusive processing set case for scheduling cranes with different weight capacity limits.

To evaluate the performance of an algorithm, we use the worst-case performance ratio and the competitive ratio for the offline problem and online problem, respectively. Let σ^* denote the offline optimal schedule and σ denote the schedule generated by the algorithm in context. Let $C_{\max}(\sigma^*)$ and $C_{\max}(\sigma)$ denote the makespan of σ^* and σ, respectively. If $C_{\max}(\sigma) \leq \rho C_{\max}(\sigma^*)$, this algorithm is said to be a ρ-approximation algorithm for the offline problem, and a ρ-competitive algorithm for the online problem.

Lenstra et al. [6] provided a 2-approximation algorithm for the offline problem $P|\mathcal{M}_j|C_{\max}$ in which all jobs are available at time zero. After a long time, Shchepin and Vakhania [8] gave a slightly improved $(2-\frac{1}{m})$-approximation algorithm for it. Gabriella et al. [1] considered the problem $P|\mathcal{M}_j(nested)|C_{\max}$ and obtained a PTAS. Leah and Asaf [4] presented PTAS for both $P|\mathcal{M}_j(nested)|C_{\max}$ and $P|\mathcal{M}_j(tree)|C_{\max}$. Since the inclusive processing set is a special case of the nested processing set and the tree-like processing set, $P|\mathcal{M}_j(inclusive)|C_{\max}$ has a PTAS.

Shmoys et al. [9] showed that if there is a ρ-approximation algorithm for some scheduling problem in which all jobs are available at time zero, then there exists a 2ρ-competitive algorithm for the corresponding problem in which the jobs are released online over time. Therefore, $P|\mathcal{M}_j, r_j, online|C_{\max}$ has a $(4-\frac{2}{m})$-competitive algorithm, and $P|\mathcal{M}_j(inclusive), r_j, online|C_{\max}$, $P|\mathcal{M}_j(nested), r_j, online|C_{\max}$ and $P|\mathcal{M}_j(tree), r_j, online|C_{\max}$ have $(2+\epsilon)$-competitive algorithms, where ϵ is a sufficiently small positive number. Lee et al. [5] considered the problems with equal processing times, and gave a 1.618-competitive algorithm for $P2|\mathcal{M}_j(nested), r_j, p_j = p, online|C_{\max}$ and a $\sqrt{2}$-competitive algorithm for $P2|\mathcal{M}_j(inclusive), r_j, p_j = p, online|C_{\max}$.

In this paper, we consider the online problem with equal processing times, i.e., $P|\mathcal{M}_j, r_j, p_j = p, online|C_{\max}$. In Sections 2 and 3, we present optimal algorithms for the nested processing set case and the inclusive processing set case with an arbitrary number of machines, which generalize the algorithms in Lee et al. [5]. In Section 4, we design an optimal algorithm for the tree-like processing set case with three machines.

2. Nested processing sets

In this section, we consider the nested processing set problem $P|\mathcal{M}_j(nested), r_j, p_j = p, online|C_{\max}$. For its special case with two machines, Lee et al. [5] have proposed an optimal algorithm with a competitive ratio of $1+\alpha$, where $\alpha = \frac{\sqrt{5}-1}{2}$. In their algorithm, the release times of the jobs with $\mathcal{M}_j = \{M_1, M_2\}$ are uniformly delayed by αp time units, and the jobs with less flexibility, i.e., $\mathcal{M}_j = \{M_1\}$ or $\{M_2\}$, have priority for processing whenever a machine becomes idle. When the number of machines is arbitrary, we also let the jobs with the least flexibility, i.e., the smallest $|\mathcal{M}_j|$, have the highest priority. However, we do not modify the release times of some specified jobs by a fixed amount but schedule all jobs at the specified times $\alpha p + kp$, where $k = 0, 1, 2, \ldots$. So, our schedule has a more regular structure. Our algorithm is called H1, where $\tilde{J}_t$ denotes the set of unscheduled available jobs at time t.

We now prove that H1 is a $(1+\alpha)$-competitive algorithm. Let I be the original instance solved by Algorithm H1. We need to consider the auxiliary instance $\hat{I}$ obtained from I by delaying the release time of each job to the nearest time in $\{\alpha p + kp \,|\, k = 0, 1, 2, \ldots\}$. Let $\hat{r}_j$ be the release time of J_j in $\hat{I}$. Then, $\hat{r}_j = \min\{\alpha p + kp \,|\, \alpha p + kp \geq r_j\}$. Noticing that $\hat{r}_j - r_j < p$ holds for any j, we have the following lemma.

Lemma 1. *$C_{\max}(\sigma^*) > C_{\max}(\hat{\sigma}^*) - p$, where σ^* and $\hat{\sigma}^*$ denote the offline optimal schedules of I and $\hat{I}$, respectively.*

In addition, as all jobs are released at the specified times $\alpha p + kp$ $(k = 0, 1, 2, \ldots)$ in $\hat{I}$, we can also get Lemma 2.

Lemma 2. *There must be an offline optimal schedule for $\hat{I}$ such that all jobs are scheduled at times $\alpha p + kp$ $(k = 0, 1, 2, \ldots)$.*