Computers & Operations Research 40 (2013) 180–191

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Online interval scheduling on a single machine with finite lookahead ☆

Feifeng Zheng [a,b], Yongxi Cheng [a,*], Ming Liu [c], Yinfeng Xu [a]

[a] School of Management, Xi'an Jiaotong University, Xi'an 710049, China
[b] Glorious Sun School of Business and Management, Donghua University, Shanghai 200051, China
[c] School of Economics & Management, Tongji University, Shanghai 200092, China

ARTICLE INFO

Available online 15 June 2012

Keywords:
Online interval scheduling
Lookahead
Preemption
Competitive ratio

ABSTRACT

We study an online weighted interval scheduling problem on a single machine, where all intervals have unit length and the objective is to maximize the total weight of all completed intervals. We investigate how the function of finite lookahead improves the competitivities of deterministic online heuristics, under both preemptive and non-preemptive models. The lookahead model studied in this paper is that an online heuristic is said to have a lookahead ability of *LD* if at any time point it is able to foresee all the intervals to be released within the next *LD* units of time. We investigate both competitive online heuristics and lower bounds on the competitive ratio, with lookahead $0 \le LD \le 1$ under the preemptive model, and lookahead $0 \le LD \le 2$ under the non-preemptive model. A method to transform a preemptive lookahead online algorithm to a non-preemptive online algorithm with enhanced lookahead ability is also given. Computational tests are performed to compare the practical competitivities of the online heuristics with different lookahead abilities.

1. Introduction

In the online single machine interval scheduling problem, there is one machine to schedule a set of weighted intervals with various arrival time, and at any time point at most one interval can be processed on the machine. The goal is to maximize the total weight of all completed intervals. The problem can be viewed as a special job scheduling problem in which each interval is considered to be a job, and each job has, besides its weight, an arrival time and a processing time. All the information of a job becomes known upon its arrival, that is at the release time of the job. If one does not start an interval immediately upon its arrival, or if one aborts an interval before its completion, that interval is lost. The interval scheduling problem arises naturally from various real-life applications, including the assignment of transports to loading/unloading terminals, work planning for personnel, bandwidth allocation of communication channels, etc. [1]. Refer to [1,2] for recent surveys on offline and online interval scheduling problems and their variants.

We use the concept of competitive ratio (see [3]) to measure the performance of an online algorithm $\mathcal{A}$, which is the worst case ratio between the weight obtained by an optimal offline algorithm OPT and the weight obtained by $\mathcal{A}$, over all possible input interval sequence I. More specifically, let $\mathcal{A}(I)$ and I^* denote the schedules produced by $\mathcal{A}$ and by an optimal offline algorithm OPT, on an input interval sequence I, respectively. Let $|\mathcal{A}(I)|$ and $|I^*|$ denote the total weight of all the intervals in $\mathcal{A}(I)$ and I^*, respectively. Then, the competitive ratio of $\mathcal{A}$ is defined as $c = \sup_I |I^*|/|\mathcal{A}(I)|$, where the supremum is taken over all possible input sequence I. If c is finite, then $\mathcal{A}$ is said to be competitive, or, to be more specific, c-competitive.

For the general online weighted interval scheduling problem, even on a single machine, Woeginger [4] showed that no deterministic algorithm has a finite competitive ratio. Later, Canetti and Irani [5] showed that the same also holds for randomized algorithms. On the other hand, randomized competitive algorithms do exist for special cases where there is a certain relation between the length of an interval and its weight [6,7].

1.1. Lookahead and preemption

Offline algorithms have all the information of all intervals from the very beginning, while online algorithms know nothing about future intervals. Somewhat in between are the algorithms with certain lookahead ability, which have knowledge of the intervals in the near future. Usually the model with a finite lookahead ability represents a more realistic situation. For example, a doctor who responds to patients' requests for office visits is unable to know all requests in the future, however, it is possible for him/her

☆ This work was partially supported by the National Natural Science Foundation of China under Grant nos. 71172189, 11101326, 71101106, and 71071123.

* Corresponding author.
E-mail addresses: zhengff@mail.xjtu.edu.cn (F. Zheng), chengyx@gmail.com, chengyx@mail.xjtu.edu.cn (Y. Cheng), minyivg@gmail.com (M. Liu), yfxu@mail.xjtu.edu.cn (Y. Xu).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.06.003

to know the requests in the near future, say up to the next week, via an appointment system. An online algorithm is said to have a lookahead ability of $LD \geq 0$, if at any time t the algorithm has information of all the intervals to be released in $[t, t+LD]$, including the release time and the weight of each interval.

An online interval scheduling problem is said to be under preemptive model if it is allowed to abort the interval being currently processed in order to start a new one, and in this case the aborted interval is lost; otherwise the problem is said to be under non-preemptive model, that is once the algorithm starts processing an interval it must complete that interval.

The function of lookahead has applications in various scheduling problems such as job-shop scheduling [8], transportation [9], page caching [10], etc. There are generally two types of lookahead models in the literature. The first lookahead model considers the number of future jobs to be foreseen at any time. Mao and Kincaid [11] studied a special lookahead model such that an online scheduler foresees the next $k=1$ released jobs at any time. In the single machine environment with the objective to minimize total completion time, they presented an online lookahead algorithm and proved that it outperforms most online and offline heuristics. Mandelbaum and Shabtay [12] considered the more general lookahead model such that at any time the online scheduler can foresee the next $k \geq 1$ jobs, while they considered the scenario where jobs are released over list but not over time. They assumed that jobs have different subsets of suitable machines. In the multiple machine environment to minimize the makespan, they showed that if there are only two types of jobs then there exists an online algorithm producing the optimal schedule, otherwise no such online algorithm exists. Coleman and Mao [13] studied a scheduling problem on unrelated machines with the objective of minimizing the average wait time. They compared a non-lookahead algorithm with a lookahead algorithm, and showed by simulation that the lookahead algorithm saves up to 35% of the average wait time.

The second lookahead model considers a limited length of time to be foreseen. Zheng et al. [14] investigated a single machine scheduling problem with unit length jobs, such that at any time the online algorithm can foresee the jobs to be released in the next $LD \geq 0$ units of time. For the objective to maximize the total number of completed jobs, they investigated both preemptive and non-preemptive models and presented some upper and lower bounds on the competitive ratio. When $1 \leq LD < 2$, they gave an optimal 3/2-competitive online algorithm for the non-preemptive model. Li et al. [15] studied the problem of scheduling unit length jobs on a parallel batching machine, aiming at maximizing the number of early jobs. They proved that a lookahead ability with $0 \leq LD < 1$ is useless, in the sense that it does not improve the competitive ratio of an optimal online algorithm. For $1 \leq LD < 2$, they presented an online algorithm that is 4 and 5-competitive, and proved lower bounds of 100/39 and 3/2 on the competitive ratio, for unbounded and bounded batching models, respectively. Woeginger [4] studied a single machine scheduling problem without lookahead under the preemptive model, to maximize the total weight of all completed jobs. In particular for the case where all jobs have unit length, he proved that any online deterministic algorithm cannot be better than 4-competitive. Moreover, an optimal 4-competitive online algorithm is presented, which preempts any currently processed job to start a newly arrived one, provided that the new job has a weight of at least twice larger.

1.2. Our results

In this paper we study the online weighted interval scheduling problem with the objective to maximize the total weight of all completed intervals, and explore how limited lookahead improves the performances of deterministic online heuristics. We mainly focus on the second lookahead model introduced above, and on the case where the processing time of all intervals is of unit length.

Given an input interval sequence, let Γ be the set of all intervals completed by an online algorithm. Adopting the three field notation proposed by Graham et al. [16], we use $1|r_j,online,LD|\sum_{J_j \in \Gamma} w_j$ to denote the online interval scheduling problem with unit length intervals under the non-preemptive model, where the online algorithm has the ability of lookahead with LD units of time, and r_j denotes that the intervals have various release time. We use $1|r_j,online,pmtn,LD|\sum_{J_j \in \Gamma} w_j$ to denote the same problem under the preemptive model.

We investigate both competitive online algorithms and lower bounds on the competitive ratio, with lookahead $0 \leq LD \leq 1$ under the preemptive model, and lookahead $0 \leq LD \leq 2$ under the non-preemptive model. Experimental tests are performed to compare the practical competitivities of the online algorithms with different lookahead abilities.

The rest of the paper is organized as follows. In Section 2, we study the preemptive model. We first give a tight lower bound of 4 on the competitive ratio when $0 \leq LD < 1$. For $LD=1$, we present a 3-competitive online algorithm and a lower bound of $\sqrt{2}$ on the competitive ratio. In Section 3, we study the non-preemptive model. We first observe a relation between the ability of lookahead and the ability of preemption, for deterministic online algorithms solving the interval scheduling problem. By applying this observation we transform the algorithmic results in Section 2 and in [4] under the preemptive model into new online algorithms under the non-preemptive model with enhanced lookahead ability. We also discuss lower bounds on the competitive ratio under these models. In particular, by applying a variant and extension of the idea used in [4], we prove a tight lower bound of 4 on the competitive ratio for problem $1|r_j,online,LD=1|\sum_{J_j \in \Gamma} w_j$. In Section 4, computational experiments are performed to compare the practical performances of the online algorithms with different lookahead abilities. We conclude our paper in Section 5.

2. The preemptive model with lookahead

In this section, we investigate two cases under the preemptive model: $0 \leq LD < 1$ and $LD=1$.

2.1. The case where $0 \leq LD < 1$

For the case where the time length of lookahead is strictly less than one, we have the following negative result.

Theorem 2.1. *For problem* $1|r_j,online,pmtn,LD|\sum_{J_j \in \Gamma} w_j$, *if the length of lookahead is strictly less than one, i.e.,* $0 \leq LD < 1$, *then no deterministic online algorithm has a worst case competitive ratio better than* 4.

The theorem can be proved in almost the same way as the proof for the lower bound in Theorem 4.4 in Woeginger [4]. The only difference is that for the set of intervals $S_1 = \{J_{1,1}, J_{1,2}, \dots J_{1,n_1}\}$ presented to the online algorithm in the first step, we have an extra requirement $r_{1,n_1} - r_{1,1} < 1 - LD$, where $r_{1,1}$ and r_{1,n_1} are the release time of $J_{1,1}$ and J_{1,n_1}, respectively. This requirement insures that when an online algorithm decides whether to start processing an interval in S_1, it cannot foresee the release of any interval in set S_2, which is to be released in the second step. It can be verified that similarly, when an online algorithm decides whether to start processing an interval in S_i (to be released in Step i), it cannot foresee the release of any