Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

An approximation algorithm based on game theory for scheduling simple linear deteriorating jobs

Kenli Li [a,*], Chubo Liu [a], Keqin Li [a,b]

[a] College of Information Science and Engineering, Hunan University, National Supercomputing Center in Changsha, Changsha, 410082, China
[b] Department of Computer Science, State University of New York, New Paltz, NY 12561, USA

ARTICLE INFO

Article history:
Received 5 August 2013
Received in revised form 2 May 2014
Accepted 28 May 2014
Available online xxxx
Communicated by D.-Z. Du

Keywords:
Approximation algorithm
Non-cooperative game theory
Price of anarchy
Simple linear deterioration

ABSTRACT

We consider the scheduling of simple linear deteriorating jobs on parallel machines from a new perspective based on game theory. In scheduling, jobs are often controlled by independent and selfish agents, in which each agent tries to select a machine for processing that optimizes its own payoff while ignoring the others. We formalize this situation as a game in which the players are job owners, the strategies are machines, and a player's utility is inversely proportional to the total completion time of the machine selected by the agent. The price of anarchy is the ratio between the worst-case equilibrium makespan and the optimal makespan. In this paper, we design a game theoretic approximation algorithm ***A*** and prove that it converges to a pure-strategy Nash equilibrium in a linear number of rounds. We also derive the upper bound on the price of anarchy of ***A*** and further show that the ratio obtained by ***A*** is tight. Finally, we analyze the time complexity of the proposed algorithm.

1. Introduction

It is a classical problem to schedule jobs with fixed processing times. However, there are many situations in which the processing time of a job increases with the delay of the starting time. Examples can be found in fire fighting, steel production, financial management [1,2], where delay in dealing with a job results in an increasing effort to complete the job. Such problems are generally known as scheduling with deterioration effect. The reader is referred to Kunnathur and Gupta [1], and Mosheiov [2,3] for motivations to model job deterioration in such a manner.

Scheduling of deteriorating jobs was initiated by Browne and Yechiali [4], Gupta and Gupta [5]. They defined a linear deteriorating job as a job whose actual processing time linearly increases when its starting time postpones. More precisely, the processing time of job J_i is expressed as $p_i = a_i + b_i t_i$, where a_i is the basic processing time, b_i (> 0) the deteriorating rate, and t_i the starting time. They showed that scheduling jobs in a non-decreasing order of a_i/b_i minimizes the makespan. Mosheiov [2] further introduced the concept of simple linear deteriorating job in which $p_i = b_i t_i$. Several polynomial time algorithms were proposed for the objective to minimize makespan, flow time, total completion time, total general completion time, and so on. Yu and Wong [6] proposed an algorithm DSDR (Delayed Smallest Deteriorating Rate) to minimize the total general completion time. The above research focused on a single machine for linear deteriorating jobs scheduling.

With respect to simple linear deteriorating jobs scheduling on m parallel machines, the problem becomes more complicated. Mosheiov [3] proved that the makespan minimization problem is strongly NP-hard even for a two-machine case. An

* Corresponding author.
E-mail addresses: lkl@hnu.edu.cn (K. Li), liuchubo@hnu.edu.cn (C. Liu), lik@newpaltz.edu (K. Li).

http://dx.doi.org/10.1016/j.tcs.2014.05.023
0304-3975/

asymptotically optimal heuristic algorithm was given in that paper. Ji and Cheng studied the problem in [7] to minimize the total completion time and they further [8] showed that the problems to minimize makespan, total machine load, and total completion time are strongly NP-hard with an arbitrary number of machines and NP-hard in the ordinary sense with a fixed number of machines. They proved the non-existence of polynomial time approximation algorithm with a constant ratio when the number of machines is arbitrary for the former two problems, and then proposed two similar fully polynomial time approximation schemes (FPTAS). Miao, Zhang and Cao [9] considered the makespan minimization problem with simple linear deteriorating function (i.e., $p_i = b_i t_i$), and also proposed an FPTAS. For more results on the scheduling of linear deteriorating jobs, the reader is referred to [10–14].

In this paper, we consider the scheduling of simple linear deteriorating jobs on m parallel machines from a new perspective based on game theory. Each job is regarded as a player and we propose an algorithm $\boldsymbol{A}$. We prove that the proposed algorithm is a potential game and converges to a *Nash equilibrium* in a linear number of rounds. The price of anarchy of the algorithm is $(1+b_{\max})^{\frac{m-1}{m}}$, where $b_{\max}$ is the maximum deteriorating rate of all jobs. We further show that the price of anarchy obtained by $\boldsymbol{A}$ is tight. Finally, we analyze the time complexity of the algorithm, which is $\Theta(n\log(n))$ where n is the number of jobs to be scheduled.

The rest of the paper is organized as follows. In Section 2, we formally define the problem and give some necessary notations for the discussion. In Section 3, we introduce the concept of non-cooperative game theory and consider the scheduling problem in a game way. In Section 4, we present an approximation algorithm and analyze its properties. We conclude the paper and suggest some interesting topics for future research in the last section.

2. Problem statement and notations

There are a set of n simple linear deteriorating jobs $\ell = \{J_1, J_2, \ldots, J_n\}$, which are simultaneously available at time t_0, to be scheduled on m (>0) identical parallel machines. Once a job has been processed, it cannot be interrupted by any other job until it is finished. For each job J_i, its actual processing time is $p_i = b_i t_i$, where b_i (>0) and t_i are its deteriorating rate and starting time respectively. We assume that $t_0 > 0$, since otherwise if $t_0 = 0$, it is trivial that the makespan is equal to 0 due to $p_i = 0$ ($i \in \{1, \ldots, n\}$). The objective is to minimize the makespan, i.e., the completion time of the last finished job. We denote the problem as $P_m|p_i = b_i t_i|C_{\max}$.

Denote by M_j ($j \in \{1, \ldots, m\}$) the j-th machine in the system. Let $C_{[i]}$ be the completion time of the i-th job on a machine. It is noted that $C_{[i+1]} = t_{[i+1]} + p_{[i+1]} = t_{[i+1]}(1+b_{[i+1]}) = C_{[i]}(1+b_{[i+1]})$. Thus, by induction, we have

$$C_{[i]} = t_0 \prod_{j=1}^{i}(1+b_{[j]}),$$

for every $i \geq 1$ on this machine.

Given a schedule, let n_j ($j \in \{1, \ldots, m\}$) be the number of jobs scheduled on machine M_j. Then $n_j \in \{1, \ldots, n\}$ for $j \in \{1, \ldots, m\}$ and $\sum n_j = n$. Denote C_j the completion time of the last finished job on M_j. We obtain

$$C_j = t_0 \prod_{i=1}^{n_j}(1+b_{[i]}).$$

Our goal is to minimize the makespan, i.e., $C_{\max} = \max_{j=1,\ldots,m}(C_j)$. Since $t_0 > 0$ is an extraneously given constant, we assume without loss of generality that $t_0 = 1$.

3. Non-cooperative game theory

Game theory studies the problems in which players try to maximize their returns. In this section, we formulate the problem $P_m|p_i = b_i t_i|C_{\max}$ as a non-cooperative game among the players. As described in [15], a non-cooperative game consists of a set of players, a set of strategies, and preferences over the set of strategies. In this paper, each job in ℓ is regarded as a player, i.e., the set of players is the n non-preemptive jobs. The strategy set S_i of player i ($i \in \{1, \ldots, n\}$) is the m identical parallel machines, i.e., $S_i = \{1, \ldots, m\}$. We define $S = \bigtimes_{i=1}^{n} S_i$.

The strategy of player i is represented by s_i ($s_i \in \{1, \ldots, m\}$). Thus, for all players we obtain a strategy vector $\boldsymbol{s} = (s_1, \ldots, s_n)$ ($\boldsymbol{s} \in S$). $\boldsymbol{s}$ is called a job scheduling strategy vector. Each player's preference is represented by its utility u_i ($i \in \{1, \ldots, n\}$) and the player tries to maximize it. We denote $u_i = u_i(\boldsymbol{s})$ which means the utility of player i when the strategy vector of all players is $\boldsymbol{s}$. A player i prefers the strategy s_i^* to the strategy s_i' if and only if $u_i(s_i^*, \boldsymbol{s}_{-i}) > u_i(s_i', \boldsymbol{s}_{-i})$, where $\boldsymbol{s}_{-i}$ is used to denote what remains from $\boldsymbol{s}$ when its i-th element s_i is dropped. $(s_i', \boldsymbol{s}_{-i})$ denotes the strategy vector after replacing s_i by s_i'.

Let $C_j(\boldsymbol{s})$ ($j \in \{1, \ldots, m\}$) be the completion time of last finished job on M_j when the strategy vector is $\boldsymbol{s}$. $C_{\max}(\boldsymbol{s})$ denotes the maximal completion time of all machines under $\boldsymbol{s}$. Similarly, let $C_{\min}(\boldsymbol{s})$ be the minimal completion time. Then the scheduling problem stated in Section 2 is to find a strategy vector $\boldsymbol{s}$ such that $C_{\max}(\boldsymbol{s})$ is minimized, i.e.,

$$\textit{minimize} \quad \max_{j=1,\ldots,m} C_j(\boldsymbol{s}), \quad \boldsymbol{s} \in S \tag{OPT}$$

Please cite this article in press as: K. Li et al., An approximation algorithm based on game theory for scheduling simple linear deteriorating jobs, Theor. Comput. Sci. (2014), http://dx.doi.org/10.1016/j.tcs.2014.05.023