Computers & Industrial Engineering 78 (2014) 66–73

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Two-agent single-machine scheduling to minimize the weighted sum of the agents' objective functions ☆

T.C.E. Cheng [a], Chih-Yu Liu [b], Wen-Chiung Lee [b,*], M. Ji [c]

[a] Department of Logistics and Maritime Studies, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong
[b] Department of Statistics, Feng Chia University, Taichung, Taiwan
[c] School of Computer Science and Information Engineering, Contemporary Business and Trade Research Center, Zhejiang Gongshang University, Hangzhou 310018, PR China

ARTICLE INFO

Article history:
Received 26 March 2013
Received in revised form 25 September 2014
Accepted 28 September 2014
Available online 8 October 2014

Keywords:
Scheduling
Total completion time
Total tardiness
Two agents
Single machine

ABSTRACT

Scheduling with two competing agents on a single machine has become a popular research topic in recent years. Most research focuses on minimizing the objective function of one agent, subject to the objective function of the other agent does not exceed a given limit. In this paper we adopt a weighted combination approach to treat the two-agent single-machine scheduling problem. The objective that we seek to minimize is the weighted sum of the total completion time of the jobs of one agent and the total tardiness of the jobs of the other agent. We provide two branch-and-bound algorithms to solve the problem. In addition, we present a simulated annealing and two genetic algorithms to obtain near-optimal solutions. We report the results of the computational experiments conducted to test the performance of the proposed algorithms.

1. Introduction

Most scheduling problems require a common objective function to be minimized for all the jobs. Recently, a new research topic contemplates a situation where the jobs might come from several customers that have their own goals to pursue, which is known as multi-agent scheduling. Many real-life operational scenarios give rise to multi-agent scheduling. For instance, maintenance operations compete with job processing for machine time in production with planned maintenance. Various types of packets and services in a telecommunications system, such as voice, web browsing, and file transferring via ftp, compete for radio resource usage. In transportation, agents owning their trains or aircraft transportation resources compete for the usage of rail tracks or airport lanes. A company's manufacturing department might be concerned about finishing jobs before their deadlines, while its research and development department might have more concern for quick response time in a prototype shop. Please refer to Kubzin and Strusevich (2006), Soomer and Franx (2008), Meiners and Torng (2009).

Baker and Smith (2003), Agnetis, Mirchandani, Pacciarelli, and Pacifici (2004) were pioneers of multi-agent scheduling research. Many researchers have since expended an abundance of effort on this new topic of scheduling research, for example, Yuan, Shang, and Feng (2005), Cheng, Ng, and Yuan (2006, 2008), Ng, Cheng, and Yuan (2006), Agnetis, Pacciarelli, and Pacifici (2007, 2009), Liu and Tang (2008), Fan, Cheng, Li, and Feng (2013), Lee et al. (2013), Wu, Lee, and Liou (2013), Wu et al. (2013), Choi and Chung (2014), Elvikis and T'kindt (2014), Sadi, Soukhal, and Billaut (2014), Sadi et al. (2014), Xu, Liu, and Yang (2014), Xu et al. (2014), etc. Recently, Liu, Tang, and Zhou (2010a) introduced the concepts of group technology and deteriorating jobs to two-agent scheduling. Their objective is to minimize the total completion time of the jobs of one agent, given that the maximum cost of the jobs of the other agent cannot exceed a given upper bound. Lee, Wang, Shiau, and Wu (2010) studied single-machine two-agent scheduling with deteriorating jobs. They provide a branch-and-bound algorithm and three heuristic algorithms for the problem to minimize the total completion time of the jobs of one agent, given that no tardy jobs are allowed for the other agent. Leung, Pinedo, and Wan (2010) extended the problems studied by Agnetis et al. (2004) to the case with multiple identical parallel machines where job preemption is allowed. They also discussed some single-machine problems where the jobs have different release dates under the cases of preemption and non-preemption, respectively. Liu, Zhou, and Tang (2010b) considered the effects of aging and learning on two-agent single-machine scheduling. Their objective is to minimize the total completion time of the jobs of one agent with the restriction that the maximum cost of the other

☆ This manuscript was processed by Area Editor T.C. Edwin Cheng.

* Corresponding author. Tel.: +886 4 24517250x4016; fax: +886 4 24517092.
E-mail address: wclee@fcu.edu.tw (W.-C. Lee).

http://dx.doi.org/10.1016/j.cie.2014.09.028
0360-8352/

agent cannot exceed a given upper bound. Cheng, Cheng, Wu, Hsu, and Wu (2011) consider a single-machine problem with truncated learning effect. The objective is to minimize the total weighted completion time of the jobs of one agent, given that no tardy job is allowed for the other agent. Wu, Huang, and Lee (2011) study the two-agent scheduling problem with learning effects on a single machine to minimize the total tardiness of the jobs of one agent, given that no tardy job is allowed for the other agent. Lee, Chung, and Hu (2012) considered two-agent scheduling with job release times. Their objective is to minimize the total tardiness of the jobs of one agent, given that the maximum tardiness of the other agent cannot exceed a certain bound. Liu, Yi, and Zhou (2011) presented the optimal solutions for some two-agent single-machine problems with increasing linear job deterioration. Their goal is to minimize the objective function of one agent, given that the objective function of the other agent cannot exceed a certain bound. Cheng, Chung, Liao, and Lee (2013) studied a single-machine scheduling problem with release times where the objective is to minimize the total weighted completion time of jobs from the first agent with the constraint that the maximum lateness of jobs from the second agent does not exceed an upper bound. Yu, Zhang, Xu, and Yin (2013) considered two-agent scheduling with piece-rate maintenance. They showed some single-machine remains polynomially solvable.

Most research on two-agent scheduling focuses on minimizing the objective function of one agent, subject to the objective function of the other agent does not exceed a given limit. In this paper, however, we study a single-machine two-agent scheduling problem with the objective of minimizing the weighted sum of the total completion time of the jobs of one agent and the total tardiness of the jobs of the other agent. These two objective functions are conflicting because completion time and tardiness are proxies for internal and external efficiency, respectively. To the best of our knowledge, Baker and Smith (2003), Wu, Yin, Wu, Wu, and Hsu (2014) are the only researchers who have considered two-agent single-machine scheduling to minimize the weighted sum of the agents' objective functions in the literature.

We formulate the problem as follows: There is a set $\{1, 2, \ldots, n\}$ of n jobs that are simultaneously ready to be processed on a single machine. Each job belongs to either agent AG_1 or AG_2. Associated with each job j, there is a processing time p_j, a due date d_j, and an agent code I_j, where $I_j = 1$ ($I_j = 2$) if job j belongs to AG_1 (AG_2). Under a schedule S, let $C_j(S)$ be the completion time of job j and $T_j(S) = \max\{0, C_j(S) - d_j\}$ be the tardiness of job j. Using the conventional three-field notation for describing scheduling problems, we denote our problem as $1||\theta\sum_{j\in AG_1} C_j + (1-\theta)\sum_{j\in AG_2} T_j$, where $0 \leqslant \theta \leqslant 1$ is a weighting factor for first agent's objective function.

The rest of the paper is organized as follows: In the next section we provide two branch-and-bound algorithms to solve the problem. In Section 3 we propose a simulated annealing and two genetic algorithms to obtain near-optimal solutions for the problem. In Section 4 we present the results of computational experiments conducted to evaluate the efficiency of the branch-and-bound algorithms and the performance of the heuristic algorithms. We conclude the paper and suggest topics for future research in the final section.

2. Branch-and-bound algorithms

It is noted that the problem under consideration is computationally intractable because when all the jobs are from agent AG_2, the problem reduces to the NP-hard classical single-machine total tardiness problem (Du & Leung, 1990). In this section we first provide several dominance properties, followed by a lower bound and descriptions of two branch-and-bound algorithms.

2.1. Dominance properties

In this subsection we derive two non-adjacent and several adjacent dominance properties of the optimal solution. These properties help reduce the search space and speed up the searching process for the optimal solution for our problem.

Theorem 1. *If jobs i and $j \in AG_1$, and $p_i < p_j$, then job i is scheduled before job j in an optimal sequence.*

Theorem 2. *If jobs i and $j \in AG_2$, $p_i < p_j$, and $d_i < d_j$, then job i is scheduled before job j in an optimal sequence.*

Let S and S' be two schedules of the jobs. The difference between them is a pairwise interchange of two adjacent jobs, i.e., $S = (\pi, i, j, \pi')$ and $S' = (\pi, j, i, \pi')$, where π and π' each denote a partial sequence. In addition, let t denote the completion time of the last job in π.

Property 1. *If jobs i and $j \in AG_2$, and $d_i < t + p_i + p_j \leqslant d_j$, then S dominates S'.*

Proof. The completion times of jobs i and j in S are

$$C_i(S) = t + p_i \tag{1}$$

and

$$C_j(S) = t + p_i + p_j. \tag{2}$$

Similarly, the completion times of jobs j and i in S' are

$$C_j(S') = t + p_j \tag{3}$$

and

$$C_i(S') = t + p_j + p_i. \tag{4}$$

Since $d_i < t + p_i + p_j \leqslant d_j$, we have

$$T_i(S) = \max\{t + p_i - d_i, 0\} \tag{5}$$
$$T_j(S) = 0 \tag{6}$$
$$T_j(S') = 0 \tag{7}$$

and

$$T_i(S') = t + p_j + p_i - d_i. \tag{8}$$

To show S dominates S', it suffices to show that $T_i(S) + T_j(S) < T_j(S') + T_i(S')$. Suppose that $T_i(S)$ is not zero. Note that this is the more restrictive case since it comprises the case that $T_i(S)$ is zero. From (5) and (8), we have

$$T_j(S') + T_i(S') - T_i(S) - T_j(S) = p_j > 0.$$

Thus, S dominates S'. □

Property 2. *If jobs i and $j \in AG_2$, $t + p_i \geqslant d_i$, $t + p_j \geqslant d_j$, and $p_i < p_j$, then S dominates S'.*

Property 3. *If jobs i and $j \in AG_2$, $t + p_i \geqslant d_i$, $t + p_i + p_j \geqslant d_j \geqslant t + p_j$, and $t + p_i < d_j$, then S dominates S'.*

Property 4. *If jobs i and $j \in AG_2$, $t + p_i \leqslant d_i \leqslant t + p_j$, and $t + p_j > d_j$, then S dominates S'.*

Property 5. *If jobs i and $j \in AG_2$, $t + p_i \leqslant d_i \leqslant t + p_j + p_i$, $t + p_j \leqslant d_j \leqslant t + p_i + p_j$, and $d_i < d_j$, then S dominates S'.*

Property 6. *If job $i \in AG_1$, job $j \in AG_2$, $t + p_j \geqslant d_j$, and $(1-\theta)p_i < \theta p_j$, then S dominates S'.*