Computers & Operations Research 40 (2013) 109–116

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

Scheduling problems with two competing agents to minimized weighted earliness–tardiness

Enrique Gerstl, Gur Mosheiov *

School of Business Administration, The Hebrew University, Jerusalem 91905, Israel

A R T I C L E I N F O

Available online 23 June 2012

Keywords:
Scheduling
Parallel machines
Two-agents
Earliness–tardiness
Common due-date

A B S T R A C T

We study scheduling problems with *two competing agents*, sharing the same machines. All the jobs of both agents have identical processing times and a common due date. Each agent needs to process a set of jobs, and has his own objective function. The objective of the first agent is total weighted earliness–tardiness, whereas the objective of the second agent is maximum weighted deviation from the common due date. Our goal is to minimize the objective of the first agent, subject to an upper bound on the objective value of the second agent. We consider a single machine, and parallel (both identical and uniform) machine settings. An optimal solution in all cases is shown to be obtained in polynomial time by solving a number of linear assignment problems. We show that the running times of the single and the parallel identical machine algorithms are $O(n^{m+3})$, where n is the number of jobs and m is the number of machines. The algorithm for solving the problem on parallel uniform machine requires $O(n^{m+3}m^3)$ time, and under very reasonable assumptions on the machine speeds, is reduced to $O(n^{m+3})$. Since the number of machines is given, these running times are polynomial in the number of jobs.

1. Introduction

In classical *Just-in-Time* (JIT) scheduling problems, the objective is minimum earliness–tardiness cost of the job completion times from their due dates. Most of the early studies (see e.g. Baker and Scudder [1]) focused on scheduling problems where all the jobs share a *common due date*. Among these studies, some considered *minsum* objectives (where the scheduler goal is to minimize the total cost incurred by all the jobs). Others researchers focused on *minmax* objectives (where the goal is to minimize the cost of the worst scheduled job). In their seminal paper, Hall and Posner [2] proved that the single machine problem to minimize the weighted deviation of the jobs completion times from a common due date is NP-hard. This minsum version is known as the Weighted Earliness–Tardiness (WET) problem. Cheng and Li [3] proved that the problem of minimizing the *maximum* weighted deviations among the job completion times from a common due date is NP-hard. This minmax version is known as the Minimum Weighted Absolute Lateness (MWAL) problem.

In recent years, scheduling researchers have focused on a setting of *two competing agents*. In this setting, two agents who need to process their own sets of jobs, compete on the use of a common resource. Each one of the two agents has his own objective function, and the goal is to find the joint schedule that minimizes the value of the objective function of one agent, subject to an upper bound on the value of the objective function of the second agent. Baker and Cole Smith [4] introduced the first scheduling paper dealing with two agents sharing a single processor. They focused on minimizing makespan, maximum lateness and total weighted completion time. Agnetis et al. [5] extended significantly the list of scheduling measures and the machine settings, and were followed by: Ng et al. [6], Cheng et al. [7], Lee et al. [8], Agnetis et al. [9], Gawiejnowicz et al. [10], Leung et al. [11], Mor and Mosheiov [12], Lee et al. [13], Li and Hsu [14], Li and Yuan [15], and Mor and Mosheiov [16], among others.

In this paper we study a two-agent scheduling JIT problem, where the objective is to minimize the minsum measure of the first agent (WET), subject to an upper bound on the minmax measure of the second agent (MWAL). This setting of objective functions is a generalization of some of the settings introduced in [5], since the weighted earliness–tardiness objective (of the first agent) is clearly an extension of weighted tardiness or of total completion times studied in Agnetis et al., and maximum weighted deviation (of the second agent) is an extension of maximum weighted completion time assumed in Agnetis et al. The general problem (i.e. assuming general job-dependent processing times) is NP-hard, since, as mentioned above, even

* Corresponding author. Tel.: +972 2 588 3108; fax: +972 2 588 1341.
E-mail address: msomer@mscc.huji.ac.il (G. Mosheiov).

0305-0548/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cor.2012.05.019

the single agent WET problem is NP-hard. (Note that WET is NP-hard even for symmetric earliness–tardiness cost, and for a non-restrictive, i.e. sufficiently large, common due date, see Hall and Posner [2]). We focus here on the important and extensively studied special case of *identical* jobs. Beyond its theoretical importance, this setting is known to have numerous applications, including the wide range of common production lines focusing on identical items. Scheduling with identical jobs has been studied extensively, under various objective functions and machine settings. The list of papers dealing with JIT scheduling with both identical jobs and a common due date contains (among others): Hall and Posner [2], Cheng and Chen [17], Mosheiov and Shadmon [18], Cheng et al. [19], Mosheiov and Yovel [20], Li et al. [21], Tuong and Soukhal [22], Mosheiov and Sarig [23,24], Tuong and Soukhal [25], and Drobouchevitch and Sidney [26].

The input for the problems studied here consists of (*i*) a list of earliness and tardiness weights for the first agent, (*ii*) a list of earliness and tardiness weights for the second agent, (*iii*) an upper bound on the maximum permitted earliness/tardiness value of the second agent, and (*iv*) the common due date (which may be either large or small, i.e. *non-restrictive* or *restrictive*). First we solve the single machine problem. We introduce an $O(n^4)$ solution procedure (where n is the number of jobs of both agents). Then we extend this solution procedure to the setting of parallel machines. We consider first parallel *identical* machines, and introduce an $O(n^{m+3})$ solution algorithm (where m is the number of machines). Then we study parallel *uniform* machines, and an $O(n^{m+3}m^3)$ time algorithm is introduced for the general case. An improved $O(n^{m+3})$ algorithm is proposed under a very reasonable assumption on the machine speed factors. Since m is given (i.e. m is not part of the input), the above running times for parallel machines settings are *polynomial* in n.

The paper is organized as follows. In Section 2 we introduce the notation and the formulation of the problem. Section 3 focuses on the single machine case. Sections 4 and 5 present the solutions for parallel identical and uniform machines, respectively.

2. Formulation

Two agents, denoted by X and Y, respectively, need to process their jobs on a single processor. Agent X processes n^X jobs, and agent Y processes n^Y jobs. $n=n^X+n^Y$ denote the total number of jobs. All the jobs are available at time zero, and preemption is not allowed. p_j^Z denotes the processing time of job j of agent Z, $j=1,\dots,n^Z$, $Z=X,Y$. We assume that the job processing times are identical, and after appropriate scaling (i.e. without loss of generality) they are assumed to be *unit time* jobs, i.e. $p_j^Z=1$. The earliness unit cost of job j of agent Z is denoted by α_j^Z, $j=1,\dots,n^Z$, $Z=X,Y$. Similarly, the tardiness unit cost of job j of agent Z is denoted by β_j^Z, $j=1,\dots,n^Z$, $Z=X,Y$. All the jobs share a common due date denoted by d.

For a given schedule, C_j^Z denotes the completion time of job j of agent Z, $E_j^Z=\max\{0,d-C_j^Z\}$ denotes the earliness of job j of agent Z, and $T_j^Z=\max\{0,C_j^Z-d\}$, denotes the tardiness of job j of agent Z, $j=1,\dots,n^Z$, $Z=X,Y$. The total Weighted Earliness–tardiness (*WET*) cost of agent X is given by $WET=\sum_{j=1}^{n^X}(\alpha_j^X E_j^X+\beta_j^X T_j^X)$. The Maximum Earliness–tardiness (*MET*) cost of agent Y is given by $MET=\max_{j=1,\dots,n^Y}\{\alpha_j^Y E_j^Y+\beta_j^Y T_j^Y\}$. (Note that for a given j, either $E_j^Z=0$ or $T_j^Z=0$, $Z=X, Y$). Finally, let Q denote the upper bound on the maximum earliness–tardiness cost of agent Y.

Using the standard notation of scheduling problems, the single machine problem is:

$\mathbf{P1}: 1/p_j^Z=1,d_j^Z=d/WET:MET\le Q.$

The extensions to settings of m parallel identical machines and parallel uniform machines, respectively, are:

$\mathbf{P2}: Pm/p_j^Z=1,d_j^Z=d/WET:MET\le Q$,and

$\mathbf{P3}: Qm/p_j^Z=1,d_j^Z=d/WET:MET\le Q.$

3. The single machine case

In this section we introduce a polynomial time solution for problem **P1**, where both agents share a single machine. We first claim that an optimal schedule exists with no idle time between consecutive jobs. [If an idle time exists between two early jobs, close this idle time by delaying the early jobs. The resulting schedule is clearly feasible since the cost of the early Y-jobs is reduced, and the objective function value (based on the X-jobs) is smaller. Similarly, if an idle time exists between two tardy jobs, a better schedule is obtained by starting the tardy jobs earlier.]

Next we prove the following property:

Property 1. *An optimal schedule exists such that either (i) the first scheduled job starts at time zero, or (ii) at least one Y-job has cost value of Q, or (iii) an X-job is completed exactly at the due date.*

Proof. Consider an optimal schedule which does not satisfy this property, i.e. it starts at some positive time, the cost of all the Y-jobs is strictly smaller than Q, and no X-job is completed at time d. Let S_E^X and S_T^X denote the sets of early and tardy X-jobs, respectively. The proof is based on small shifts of the entire schedule to the left (to start earlier), and to the right (to start later). When starting the schedule ε units of time earlier, the change of the total earliness–tardiness cost is given by $\Delta=\sum_{j\in S_E^X}\alpha_j^X E_j^X-\sum_{j\in S_T^X}\beta_j^X T_j^X$. On the other hand, when starting the schedule ε units of time later, the change of the total earliness–tardiness cost is $-\Delta$. Hence, shifting the entire sequence either to the left or to the right will result in a schedule with no higher cost, implying that the new schedule is optimal as well. This shift (with no increasing the cost) is feasible until either (*i*) the starting time of the first job becomes zero, or (*ii*) one of the Y-jobs reaches its maximum cost (Q), or (*iii*) one of the X-jobs is completed at time d. This completes the proof. □

Based on Property 1, we introduce a solution procedure for problem **P1**. Each of the above cases ((*i*), (*ii*) and (*iii*)) is treated separately. Cases (*i*) and (*iii*) are reduced to a single linear assignment problem (LAP). Case (*ii*) is reduced to a series of LAPs. Each LAP represents a candidate for optimality, and the global optimum is determined by the minimal-cost LAP.

Case (*i*): The schedule starts at time zero.

Let k denote the last job (either an X-job or a Y-job) completed prior to or at time d. We define $\delta=d-C_k^Z$. The assignment matrix consists of n rows reflecting the n jobs, and n columns reflecting the n positions. (Since the schedule starts at time zero and contains no idle time, the matrix dimensions are $n\times n$.) The jobs can be assigned to $\lfloor d\rfloor$ early potential positions, and $n-\lfloor d\rfloor$ tardy potential positions. Let $cost_{jr}$ denote the cost of assigning job j to position r. For convenience, let f_j^Y be the *first* position where a Y-job j obeys the earliness constraint, i.e.: $\alpha_j^Y(d-r)>Q$, $r=1,\dots,f_j^Y-1$, and $\alpha_j^Y(d-f_j^Y)\le Q$. Similarly, let l_j^Y be the *last* tardy position where a Y-job j obeys the tardiness constraint, i.e.: $\beta_j^Y(r-d)>Q$, $r=l_j^Y+1,\dots,n$, and $\beta_j^Y(l_j^Y-d)\le Q$. The assignment cost matrix consists of two blocks: one for positions of the X-jobs and one for the positions of the Y-jobs. The first block contains: (*i*) the early job-positions of agent X: $cost_{jr}=\alpha_j^X(\lfloor d\rfloor-r+\delta)$, $j=1,\dots,n^X$, $r=1,\dots,\lfloor d\rfloor$, where $\lfloor d\rfloor$ denotes the largest integer less or equal to d, and (*ii*) the tardy job-positions of