Computers & Industrial Engineering 119 (2018) 146–156

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Efficient heuristics for minimizing weighted sum of squared tardiness on identical parallel machines

Jeffrey Schaller[a,*], Jorge M.S. Valente[b]

[a] Department of Business Administration, Eastern Connecticut State University, 83 Windham St., Willimantic, CT 06226-2295, United States
[b] LIAAD – INESCTEC LA, Faculdade de Economia, Universidade do Porto, Rua Dr. Roberto Frias, 4200-464 Porto, Portugal

ARTICLE INFO

Keywords:
Scheduling
Sequencing
Heuristics
Integer programming

ABSTRACT

Scheduling jobs on a set of identical parallel machines using efficient heuristics when the objective is to minimize total weighted squared tardiness is considered. Two efficient heuristics and an improvement procedure are presented for the problem. These heuristics and other heuristics are tested using problem sets that represent a variety of conditions. The results show that one of the heuristics consistently performs better than the other heuristics tested. It is also shown how these heuristics can be incorporated into other procedures such as the existing Lagrangian relaxation procedure or meta-heuristics to obtain improved solutions for medium sized problems.

1. Introduction and problem description

This research investigates the use of efficient heuristics when scheduling on identical parallel machines with an objective of minimizing total weighted squared tardiness. Let M be the number of machines and n the number of jobs to be scheduled. Let p_j, w_j, and d_j represent the processing time, weight and due date for each job j (j = 1, …, n) respectively, and C_j is the time that job j is completed, j = 1, …, n. The weights (w_j) for each job would usually be determined by the scheduler or other management individuals in the organization and would reflect things such as product margins or importance of the customer to the organization's success. The tardiness of job j, T_j, is defined as: $T_j = \max\{0, C_j - d_j\}$, j = 1, …, n. The objective function, Z, can be expressed as: $Z = \sum_{j=1}^{n} w_j * T_j^2$. If the job to be sequenced in position j of machine m (m = 1, …, M) is denoted as [j]m, then $C_{[j]m} = C_{[j-1]m} + p_{[j]}$, with $C_{[0]m} = 0$ for m = 1, …, M.

The problem described in the previous paragraph was originally motivated by studies of scheduling in an Aerospace and Defense manufacturing facility (Hoitomt, Luh, Max, & Pattipati, 1990; Luh & Hoitomt, 1993). A distinguishing feature of the problem is that the objective is a function of the square of tardiness. Manufacturers, as well as service organizations, operate as part of supply chains in which timely delivery to customers is crucial, and the cost of tardy deliveries can be very high. Taguchi (1986) proposed a quadratic penalty to measure quality costs incurred by customers. Timely delivery to meet requested customer due dates is a dimension of quality that is very important. Quadratic tardiness can be used as part of a measure of on time delivery to represent the increased cost as tardiness increases, as suggested by Taguchi (1986). Traditionally, linear functions of tardiness have been used to evaluate schedules. The sum of weighted tardiness would be the linear equivalent of the objective considered in this paper. Frequently, the solution can be different depending on whether weighted tardiness or weighted squared tardiness is used as the objective. For example suppose two jobs are to be sequenced first and second on the same machine and have these processing times, weights and due dates: $p_1 = 6$, $p_2 = 3$, $d_1 = 4$, $d_2 = 0$, $w_1 = 3$, $w_1 = 1$. If weighted tardiness is used as the objective then scheduling job 1 before job 2 would result in a total weighted tardiness of 15 for the two jobs. If job 2 is instead scheduled before job 1, the total weighted tardiness of the two jobs would be 18, so scheduling job 1 before job 2 would be better. If weighted squared tardiness is used as the objective then scheduling job 1 before job 2 would result in a total weighted squared tardiness of 93 for the two jobs. If job 2 is instead scheduled before job 1, the total weighted squared tardiness of the two jobs would be 84, so scheduling job 2 before job 1 would be better and therefore the solutions under the two objectives would be different. All customers are considered with either of these objectives. Another objective, maximum tardiness, has also been frequently used. This is a simple measure and implicitly recognizes that customer dissatisfaction with tardiness does not increase in a linear fashion, as a schedule with one job that is 10 units tardy is worse than a schedule that has two tardy jobs, each five units tardy. In fact, if squared tardiness was used, the first schedule in the preceding example would be twice as costly as the second (100 versus 50). A problem with using maximum tardiness is that it focuses on just the one

* Corresponding author.
E-mail addresses: schallerj@ecsu.ctstateu.edu (J. Schaller), jvalente@fep.up.pt (J.M.S. Valente).

https://doi.org/10.1016/j.cie.2018.03.036
Received 29 August 2017; Received in revised form 20 March 2018; Accepted 21 March 2018
Available online 26 March 2018
0360-8352/

job or customer that has the maximum tardiness, whereas squared tardiness considers all of the jobs or customers. Sun, Noble, and Klein (1999) provide additional examples contrasting squared tardiness with linear tardiness and maximum tardiness. In situations where demand is higher than the capacity of the machines or processors, the costs of tardy deliveries can become very high. Sometimes it is possible to increase capacity by using subcontractors, so the total weighted squared tardiness can be reduced. It is important to be able to generate timely schedules that would indicate the problem and allowing for searching for alternatives for meeting customer demand. If subcontractors are not available, then these schedules would allow for the evaluation of adding additional machines or processors.

To the best of the authors' knowledge, the complexity of the weighted squared tardiness on parallel machines is open. Other complexity results suggest that this problem is likely hard. Lawler (1977) and Lenstra, Rinnooy Kan, and Brucker (1977), show that the single machine weighted tardiness problem is strongly NP-hard. Garey and Johnson (1978), show that minimizing makespan on parallel machines is strongly NP-hard. Schaller and Valente (2013), developed exact algorithms for the problem, but were only able to solve small scale sized problems with up to 15 jobs and four machines in a reasonable amount of time. For these reasons we focus on efficient heuristics that can solve larger sized problems. Our objective is to see if efficient heuristics that have been used for the single-machine problem with the same objective can be adapted to the parallel machines environment, in order to generate good solutions for the problem quickly. The next section will review literature for problems with the objective of minimizing weighted squared tardiness. Efficient heuristics are presented for the problem in section three, and section four describes an improvement procedure that can be applied to solutions generated by the heuristics. Computational tests are described and results presented in section five. It is shown how the procedures can be incorporated into other procedures, in order to improve performance, in section six. Finally, the last section concludes the paper.

2. Literature review

Past research that is related to this paper includes research with a tardiness objective and the parallel machines environment. Sen, Sulek, and Dileepan (2003), provide a survey of research for minimizing weighted and unweighted tardiness. Mokotoff (2001), surveyed research for parallel machine scheduling problems. The focus of this literature review will be research with the objective of minimizing the weighted sum of squared tardiness. Previous work on problems involving an objective of minimizing the sum of weighted squared tardiness includes single-machine problems (Goncalves et al., 2016; Schaller & Valente, 2012; Sun et al., 1999; Valente & Schaller, 2012) parallel machine problems (Hoitomt et al., 1990; Luh & Hoitomt, 1993; Schaller & Valente, 2013) and job shop problems (Sun & Noble, 1999; Thomalla, 2001). Several papers, Hoitomt et al., 1990; Luh & Hoitomt, 1993; Sun et al., 1999; Sun & Noble, 1999; Thomalla, 2001, use a Lagrangian relaxation approach based on the procedure that Fisher (1973a, 1973b) used for other problems.

Hoitomt et al. (1990), studied a parallel machines shop environment at a Pratt and Whitney plant where the jobs have precedence constraints. A Lagrangian relaxation procedure was developed and demonstrated using three examples from the plant. Luh and Hoitomt (1993), also developed a Lagrangian relaxation procedure for a shop environment at a Pratt and Whitney plant that consisted of identical parallel machines. This procedure was also tested using data from the plant.

Sun et al. (1999), studied a single machine problem that included sequence-dependent setup times and release dates. They also developed a lagrangian relaxation procedure and compared their procedure against simple dispatching rules as well as more computationally intensive algorithms, namely tabu search and simulated annealing.

A job shop environment was studied by Sun and Noble (1999) and Thomalla (2001). In both papers lagrangian relaxation procedures were developed. The problem Sun and Noble (1999), studied included sequence-dependent setup times and Thomalla (2001)'s, included alternative processing plans.

Schaller and Valente (2012) and Goncalves et al. (2016), all considered the single machine problem. Schaller and Valente (2012), proposed several dominance rules, as well as branch-and-bound algorithms, which incorporate these rules, and found that problems with up to 50 jobs could be solved within a reasonable amount of time. Valente and Schaller (2012), focused on developing dispatching heuristics for the problem in order to solve large scale instances. They found that a rule that sequences jobs from the end of the schedule and works toward the beginning of the schedule worked the best. Goncalves et al. (2016), developed meta-heuristics for the problem that included an iterated local search, a variable greedy algorithm and a genetic algorithm. In computational tests the iterated local search generated solutions with lower objective values.

Schaller and Valente (2013) developed branch-and-bound algorithms for the identical parallel machines problem. These branch-and-bound algorithms utilized results from the single machine scheduling problem, but it was found that only small scale problems of up to 15 jobs could be solved in a reasonable amount of time.

3. Heuristics

In this section two heuristics are described and proposed for the problem. These two procedures are referred to as the QB and QBP procedures. Both of these procedures are adaptations to parallel machines of the single machine heuristic QBackv6 developed by Valente and Schaller (2012). The QBackv6 heuristic was found by Valente and Schaller (2012) to be the best performing heuristic for the single machine problem. The QBackv6 heuristic works from the end of a sequence to the beginning. If n is the number of jobs to be sequenced then, at each iteration k of the procedure, the jobs in positions n–k to n are sequenced and the jobs not yet sequenced need to be assigned to the first n–k–1 positions. This allows for the jobs that will be at the end of the sequence, and completed the latest, to be examined first and hopefully avoid placing jobs with a high cost in these positions. In the single machine problem this approach is straightforward because we know a job's completion time when it is sequenced, even though we do not know the exact order of the jobs to be sequenced before it on the machine. In the parallel machines case this is not so; therefore, the two approaches described in this section attempt to estimate a job's completion time when determining the sequence. The two procedures differ in the way that they estimate completion times. Once a sequence is determined, each of the procedures can create a schedule and calculate the objective value based on the actual completion times of the jobs.

Each of these two heuristics can be combined with the improvement procedure described in the next section. Also, a preprocessing problem reduction step is first performed before starting any of the procedures. This step is described first.

3.1. Preprocessing problem reduction

This step attempts to find jobs that will not be tardy no matter where they are scheduled, and hence will not contribute to the objective. These jobs are removed from the problem when the sequencing heuristics are applied. After the other jobs have been sequenced and scheduled, the jobs that were removed in this step will be placed at the end of the sequence and scheduled.

Let $P = \sum_{j=1}^{n} p_j$, let $A_j = ((P - p_j)/M) + p_j$. Azizoglu and Kirca (1998), show that a job j's maximum completion time will be A_j, and hence that job will not be tardy if $d_j \geq A_j$. This result is used in this step to identify jobs that will not be tardy, and therefore can be scheduled at the end. In the following procedure EDD[k] is the kth job if the jobs are