Applied Mathematics and Computation 219 (2013) 10783–10805

Contents lists available at SciVerse ScienceDirect

Applied Mathematics and Computation

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

Lagrangian relaxation and column generation-based lower bounds for the $Pm, h_{j1}\|\sum w_iC_i$ scheduling problem

Racem Mellouli [a,*], Imed Kacem [b,c], Chérif Sadfi [c], Chengbin Chu [c,d]

[a] Université de Sfax, ESC Unité LOgistique, Gestion Industrielle et Qualité, Sfax, Tunisia
[b] Université de Lorraine, LCOMS, Metz, France
[c] Université de Technologie de Troyes, ICD CNRS FRE 2848, Laboratoire d'Optimisation des Systèmes Industriels, Troyes, France
[d] École Centrale Paris, Laboratoire de Génie Industriel, Malabry, France

ARTICLE INFO

Keywords:
Scheduling
Availability constraint
Parallel machines
Total weighted completion time

ABSTRACT

We consider an identical parallel-machine scheduling problem to minimize the sum of weighted completion times of jobs. However, instead of allowing machines to be continuously available as it is generally assumed, we consider that each machine is subject to a deterministic and finite maintenance period. This condition increases the problem complexity. We provide for the problem an adapted heuristic, lower bounds based on Lagrangian relaxation and column generation methods. Computational study shows satisfactory results.

1. Introduction

We consider an identical parallel-machine scheduling problem to minimize the sum of weighted completion times of jobs. We assume that each machine is subject to a deterministic maintenance period. This problem has a strong practical relevance which constitutes a substantial motivation for this research. Indeed, the machine availability constraints situate the problem in a particular case of joint management of production and maintenance activities. We note that this question is playing a more and more crucial role in the industrial management field. For our problem, we consider preventive maintenance as already planned tasks. Hence, the remaining optimization consists in adapting production schedules with machine availability considerations. Concerning the choice of the optimization criterion, the total weighted completion time is one of most studied performance measures in production scheduling. Its importance has been grown since the recognition of the zero-inventory philosophy, which is based on minimizing additional costs implied by waiting times of jobs in storage facilities.

The minimization of total weighted completion time on identical parallel-machine, with one unavailability period on each machine, is denoted by $Pm, h_{j1}\|\sum w_iC_i$ according to the conventional notation. Without availability constraints, Bruno et al. [3] showed that the problem $P\|\sum w_iC_i$ is NP-hard even for two identical parallel machines. With one availability constraint, the single machine problem $1, h_1\|\sum C_i$ is NP-hard ([1,18]). Several papers have been published for the single machine case. Lee and Liman [18] showed a tight worst case error bound of 2/7 for the SPT heuristic. Sadfi et al. [25] proposed the MSPT heuristic (Modified Shortest Processing Time) showing a tight error bound of 3/17. They also provided an efficient dynamic programming method using the variable fixing technique. For the case of several maintenance periods on a single machine, they proved that there is no approximation scheme with a finite error bound unless $P = NP$. Lee [16]

* Corresponding author.
E-mail addresses: racem.mellouli@yahoo.fr (R. Mellouli), kacem@univ-metz.fr (I. Kacem), cherif.sadfi@utt.fr (C. Sadfi), chengbin.chu@ecp.fr (C. Chu).

0096-3003/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.amc.2013.05.004

proved that the resumable problem $1|r-a|\sum w_iC_i$ (with one unavailable time period) is NP-hard even if $w_i = p_i$, and proposed a dynamic programming method and a heuristic with worst-case analysis. Wang et al. [34] showed that this resumable problem, with several maintenance periods, is NP-hard in the strong sense. For the single maintenance period, they provided two heuristics with worst-case analysis. Kacem and Chu [9] studied the problem $1,h_1\|\sum w_iC_i$ and analyzed the worst case of two heuristics (WSPT and MWSPT - Modified Weighted Shortest Processing Time). For the same problem, Kacem et al. [8] compared three exact methods respectively based on branch-and-bound, integer linear programming and dynamic programming. Kacem and Chu [10] further improved this branch-and-bound algorithm using new heuristics and lower bounds. The resumable case was also studied in Kacem and Chu [11]. In Kacem and Houari [12] and Kacem and Kellerer [13], fast approximation algorithms are provided for the same problem where the second research considers different jobs releases dates. In Kacem et al. [8], the problem with multiple unavailability periods was also studied. A branch-and-bound and a dynamic programming methods were proposed.

For the parallel-machine case, few results appeared in the literature. Kaspi and Montreuil [14] and Lee [15] studied the parallel-machine scheduling problem with non-simultaneous initial availability times. They showed that the SPT algorithm minimizes the total completion times for the problem. Lee and Liman [17] studied the two capacitated parallel-machine problem to minimize the total completion times. In this problem, one machine is continuously available and the second is available until a certain date. They proved the computational complexity of the problem, proposed a dynamic programming method and studied the worst-case performance of their SPT-based heuristic. Liao et al. [20] studied the same problem. They proposed an optimal branch and bound algorithm which employs three powerful elements, including an algorithm for computing the upper bound, a lower bound algorithm, and a fathoming condition. In Mellouli et al. [22], we studied the $Pm,h_{j1}\|\sum C_i$ problem. We proposed four mixed integer linear programming methods, a dynamic programming method, several SPT-based heuristics, a constructive lower bound, some dominance properties and two branching schemes. The latter were integrated in two branch-and-bound methods. Ma et al. [21] provided a survey of different problems with machine availability constraints. Some recent works considered worst-case performances of approximation algorithms. A sample of them includes Fu et al. [6,7] in both non-resumable and resumable cases and with different models of availability constraints, Sun and Li [28] in the case of scheduling with multiple maintenance activities on two identical parallel machines, Zhao et al. [37] where one machine among two is not available in a specified time period, Tan et al. [29] where only the k-first machines are subject to unavailable periods. More recently, Shen et al. [26] presented polynomial time algorithm to solve the problem of parallel-machine scheduling with non-simultaneous machine available time.

In this paper, we propose new methods to solve both $Pm,h_{j1}\|\sum C_i$ and $Pm,h_{j1}\|\sum w_iC_i$ scheduling problems. More precisely, the contribution of this paper is the improvement of some of the latest results published in literature about these problems. These are given by Mellouli et al. [22] for the particular case of the unweighted problem in terms of lower bounds and some exact approaches. This contribution will be validated at the experimental part with comparisons between previous and recent results. The paper is organized as follows. In Section 2, we describe the studied problem. In Section 3, we propose an adapted representation of the well known WSPT policy for this problem. In Section 4, we provide Lagrangian relaxation-based lower bounds. In Section 5, we define a column generation-based lower bound. The final section contains different computational experiments to evaluate and compare the different lower bounding methods. All these bounds were included in a branch-and-bound algorithm to solve the unweighted problem.

2. Problem definition and preliminary

The problem can be stated as follows: There are n independent jobs ready to be processed on m identical parallel machines. For each job i, p_i denotes its processing time, w_i denotes its weight and C_i denotes its completion time for a given schedule. For the sake of simplicity, we note by $< p_i, w_i >$ the job i for all i. For each machine j, a maintenance period is scheduled at the time window $[T_{j,1}, T_{j,2}]$ and divides its scheduling horizon into two available time windows. This problem is equivalent to a $2m$-parallel machine problem where the first m machines are capacitated, and the second m machines are non-simultaneously available. This means that machines $j = 1,\ldots,m$ are available until time $T_{j,1}$ and machines $j = m+1,\ldots,2m$ are available since time $T_{j-m,2}$ (see Fig. 1). Smith [27] showed that the WSPT (Weighted Shortest Processing Times first) policy is optimal for the problem $1\|\sum w_iC_i$. Then for the problem $Pm,h_{j1}\|\sum w_iC_i$, we can deduce the dominance property that in each available time window, jobs are sequenced according to the WSPT rule. To facilite the formulation and without loss of generality, we assume that jobs are indexed in non-decreasing order of processing time-weight ratios $(\frac{p_i}{w_i})$ by breaking ties with the non-decreasing order of p_i. This priority rule is noted WSPT/W_{min}.

Example 1. We consider the following example where $n = 9$ and $m = 3$. Jobs are described in Table 1. These jobs are ordered according to the WSPT/W_{min} rule. Maintenance periods occur during time windows $[2,4]$, $[4,6]$ and $[5,6]$, each one on one machine. These periods can be considered as fixed jobs. We numerate them respectively as jobs 10, 11 and 12. This example is introduced to illustrate the proposed bounds construction.

Remark 1. *The equivalent problem:* Since every maintenance period divides the availability interval of each machine in two separate intervals, as they belong to two different machines, our studied problem is equivalent to a problem with $2m$ parallel machines (see Fig. 1), knowing that in the resulting problem each machine $j = 1,\ldots,2m$ is available in a single time interval