European Journal of Operational Research 236 (2014) 433–444

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Discrete Optimization

A heuristic for scheduling in a two-stage hybrid flowshop with renewable resources shared among the stages

Ewa Figielska *

Warsaw School of Computer Science, Lewartowskiego 17, 00-169 Warsaw, Poland

ARTICLE INFO

Article history:
Received 22 December 2012
Accepted 3 December 2013
Available online 9 December 2013

Keywords:
Scheduling
Heuristic
Column generation
Tabu search
Flowshop
Resource constraints

ABSTRACT

In this paper we propose a heuristic for solving the problem of resource constrained preemptive scheduling in the two-stage flowshop with one machine at the first stage and parallel unrelated machines at the second stage, where renewable resources are shared among the stages, so some quantities of the same resource can be used at different stages at the same time. Availability of every resource at any moment is limited and resource requirements of jobs are arbitrary. The objective is minimization of makespan. The problem is NP-hard. The heuristic first sequences jobs on the machine at stage 1 and then solves the preemptive scheduling problem at stage 2. Priority rules which depend on processing times and resource requirements of jobs are proposed for sequencing jobs at stage 1. A column generation algorithm which involves linear programming, a tabu search algorithm and a greedy procedure is proposed to minimize the makespan at stage 2. A lower bound on the optimal makespan in which sharing of the resources between the stages is taken into account is also derived. The performance of the heuristic evaluated experimentally by comparing the solutions to the lower bound is satisfactory.

1. Introduction

A hybrid flowshop is a system which consists of a set of two or more processing centers (processing stages) with at least one center having two or more parallel machines. A job in such a system consists of a sequence of operations processed at successive stages, and all jobs pass through processing stages in the same order. At a stage with parallel machines a job can be processed on any of the parallel machines. The flowshops with parallel machines have received considerable attention from researchers during last years (e.g. Choong, Phon-Amnuaisuk, & Alias, 2011; Gupta & Stafford, 2006; Kis & Pesch, 2005; Linn & Zhang, 1999; Oguz, Ercan, Cheng, & Fung, 2003; Rashidi, Jahandar, & Zandieh, 2010; Ribas, Leisten, & Framinñan, 2010; Ruiz & Vazquez-Rodriguez, 2010; Yaurima, Burtseva, & Tchernykh, 2009), but scheduling problems with additional renewable resource constraints have been widely investigated only for the one stage systems (e.g. Błażewicz, Cellary, Słowiński, & Węglarz, 1987; De Werra, 1988; Edis & Ozkarahan, 2012; Różycki & Węglarz, 2012; Słowiński, 1980).

In Figielska (2008, 2009, 2010, 2011) the research on the flowshop has been extended by including resource constraints into the stages. In Figielska (2008), a new algorithm based on the linear programming was proposed for minimizing makespan in the two-stage flowshop with parallel unrelated machines and renewable resources at the first stage and a single machine at the second stage. The resource requirements were assumed to be of 0–1 type. The paper of Figielska (2010) also deals with the problem with 0–1 resource requirements, but extended to the case of parallel machines and renewable resource constraints at both the stages, where to each of the stages its own resources are allotted. In Figielska (2009) the problem similar to that in Figielska (2008) but with arbitrary resource requirements was considered. In that paper, a column generation (CG) algorithm was used for solving the resource constrained parallel machine scheduling problem with job ready times equal to zero and resource availability, which was constant throughout the scheduling period. The obtained first stage schedule was composed of partial schedules. Then a metaheuristic was applied to sequencing these partial schedules so as to minimize the makespan in the flowshop. In (Figielska, 2011), three metaheuristics: tabu search, simulated annealing and a genetic algorithm were proposed to improve the solutions provided by the LP-based algorithms developed in Figielska (2008) for scheduling in the flowshop with 0–1 resource requirements.

The present paper further extends hybrid flowshop scheduling research by considering the problem with renewable resources shared among the stages. Each resource is available at any moment in limited quantity and can be used at the same time at different stages, so, if a resource is used at one stage, its quantity available at the same time at the other stages lessens. Resource requirements of jobs are assumed to be arbitrary integers. We consider the flowshop with a single machine at the first stage and parallel

* Tel.: +48 505 236 606.
E-mail address: efigielska@poczta.wwsi.edu.pl

0377-2217/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2013.12.003

unrelated machines at the second stage. The objective is to minimize the makespan.

In this paper, we propose a CG algorithm which solves the second stage problem of resource constrained preemptive scheduling on parallel unrelated machines with non-zero ready times of jobs and with resource availability at any moment depending on the usage of the resources at the first stage. Both the resource availability and ready times at stage 2 depend on the order of jobs at stage 1. For ordering the jobs, 5 priority rules based on the Johnson's rule (Johnson, 1954) are proposed. These rules (except one) depend not only on processing times of jobs but also on their resource requirements.

The proposed CG algorithm in successive iterations minimizes the makespan by using linear programming (LP), generates n (n being the number of jobs) new columns (a column determines the assignment of jobs to machines for simultaneous processing) by means of a tabu search (TS) algorithm so as to improve the solution (makespan), and uses a greedy procedure to provide an initial solution to the TS algorithm. The CG algorithm provides a schedule which is composed of n sets of partial schedules (in a partial schedule jobs are assigned to machines for simultaneous processing during some period of time so that resource constraints are satisfied at every moment) belonging to successive time intervals which correspond to processing times of the consecutive jobs executed at stage 1. The availability of the resources and the number of jobs which are ready for processing at stage 2 are different in the partial schedules from the different sets.

The performance of the heuristic is experimentally evaluated by comparing the solutions with a lower bound on the optimal makespan. Two lower bounds are considered. A lower bound is proposed in which sharing of the resources between the stages is taken into account. The influence of the choice of the TS initial solution on the convergence speed of the CG algorithm is also discussed.

The problem under consideration arises in real-life systems that are encountered in a variety of industries, e.g. in chemical, food, cosmetics and textile industries. These systems usually consist of a number of production centers which can have parallel machines. They are often subjected to some resource constraints associated with limited availability of the resources such as skilled labor, tools, power. The resources may be capable of migrating to processing centers as needed, for example workers which are cross-trained to develop the skills required to perform different tasks associated with multiple processing centers (Daniels & Mazzola, 1994). Another example of shared resources can be equipment, personnel and information technology which are used in both of the two stages of hospital operations: to generate medical record to track treatments, tests, drug dosages and costs in the first stage and to generate the second stage patient services (Chen, Du, Sherman, & Zhu, 2010).

2. Problem formulation

The problem being considered can be formally described as follows. There are n jobs to be processed at two stages in the same technological order, first at stage 1 then at stage 2. Stage 1 has one machine, at stage 2 there are m parallel unrelated machines. A job upon finishing its processing at stage 1 is ready to be processed at stage 2; it may be processed at stage 2 when a machine is available there, or it may reside in a buffer space of unlimited capacity following stage 1 until one of the machines at stage 2 becomes available. At stage 2, a job can be processed on any of the parallel machines, and its processing times may be different on different machines. The processing time of job j ($j = 1, \ldots, n$) is equal to s_j time units if it is executed at stage 1, and p_{ij} time units if it is executed on machine i ($i = 1, \ldots, m$) at stage 2. At stage 1, each job is processed without interruptions. Preemptions of jobs are allowed at stage 2, where the processing of a job on a machine may be interrupted at any moment and resumed later on the same or another machine without incurring any additional cost or loss of time. No setups or repetitions of parts of jobs are considered. Jobs for their processing at both the stages require, besides machines, additional resources. There are l types of additional resources. The resources are shared among the stages, i.e. at the same time some quantities of a resource of a given type can be used by both the stages. A resource of type r ($r = 1, \ldots, l$) is available in a quantity limited to W_r units at a time. Job j uses α_{jr} units of resource r when it is processed at stage 1 and β_{ijr} units of resource r while it is processed on machine i at stage 2. The total usage of resource r at any moment by jobs simultaneously executed at both the stages cannot exceed the availability of this resource. All required resources are granted to a job before its processing begins or resumes and they are returned by the job after finishing its processing at a stage or in the case of its preemption. The objective is to find a feasible schedule which minimizes makespan, $C_{\max}$, which is equal to the maximum job completion time at stage 2 of the flowshop.

The considered problem is NP-hard in the strong sense since the problem of preemptive scheduling in the two-stage flowshop with two identical parallel machines at one stage and one machine at another is NP-hard in the strong sense (Hoogeveen, Lenstra, & Veltman, 1996).

3. Heuristic

The heuristic first sequences jobs on the machine at stage 1 according to a priority rule. Given the order of jobs at stage 1, a CG algorithm solves the minimum makespan problem at stage 2.

For solving the problem at stage 2, the scheduling period is divided into n time intervals with lengths (apart from the last one) equal to the processing times of consecutive jobs executed at stage 1 (starting from the second job), so interval k begins at the time of the completion of the kth job at stage 1 and has the length equal to the processing time of $(k + 1)$th job at stage 1. The schedule at stage 2 begins when the first job completes its processing at stage 1. The last time interval (of index n) begins when the last job leaves stage 1, and its length is minimized.

The CG algorithm minimizes the length of the last time interval by creating partial schedules in all the time intervals (let us recall that in a partial schedule jobs are assigned to machines for simultaneous processing during some period of time so that resource constraints are satisfied at every moment). The CG algorithm is an iterative procedure which starts from a number of randomly generated columns (a column determines an assignment of jobs to machines in a partial schedule), and, in every iteration, in turn minimizes the makespan by solving an LP problem using columns obtained in previous iterations, and generates n (n is the number of jobs) new columns (one column for each time interval) improving the solution. These new columns are created by means of a TS algorithm which uses a dual improve condition as an objective function and starts from an initial solution generated by a greedy procedure. Two greedy rules are considered, both take into account the values of dual variables, but one additionally includes resource requirements of jobs.

Six priority rules used in the heuristic for sequencing jobs at stage 1 are presented in Table 1.

Rule P1 creates a random sequence of jobs at stage 1 and is used to show the usefulness of the next rules. Priority rule P2 adapts the Johnson's rule to the problem with parallel unrelated machines at the second stage. Rules P3–P6 extend rule P2 by introducing the dependence on the resource requirements of jobs. The values of