Computers & Operations Research 60 (2015) 37–54

Contents lists available at ScienceDirect

Computers & Operations Research

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

Scatter search with path relinking for the job shop with time lags and setup times

Miguel A. González [a,*], Angelo Oddi [b], Riccardo Rasconi [b], Ramiro Varela [a]

[a] Department of Computing, University of Oviedo, Campus de Gijón, 33204 Gijón, Spain
[b] Institute of Cognitive Sciences and Technologies, ISTC-CNR, Via San Martino della Battaglia 44, 00185 Rome, Italy

ARTICLE INFO

Available online 20 February 2015

Keywords:
Job shop
Setup times
Time lags
Scatter search
Tabu search
Path relinking

ABSTRACT

This paper addresses the job shop scheduling problem with time lags and sequence-dependent setup times. This is an extension of the job shop scheduling problem with many applications in real production environments. We propose a scatter search algorithm which uses path relinking and tabu search in its core. We consider both feasible and unfeasible schedules in the execution, and we propose effective neighborhood structures with the objectives of reducing the makespan and regain feasibility. We also define new procedures for estimating the quality of the neighbors. We conducted an experimental study to compare the proposed algorithm with the state-of-the-art, in benchmarks both with and without setups. In this study, our algorithm has obtained very competitive results in a reduced run time.

1. Introduction

The job shop scheduling problem (JSP) is a simple model of many real production processes. It is one of the most classical and difficult scheduling problems and it has been studied for several decades, from the seminal work of Giffler and Thompson [1] to the most recent papers, for example [2].

However, in many environments the production model has to consider additional characteristics or complex constraints. For example, in automobile, printing, semiconductor, chemical or pharmaceutical industries, setup operations such as cleaning up or changing tools are required between two consecutive jobs on the same machine. These setup operations depend on both the outgoing and incoming jobs, so they cannot be considered as being part of any of these jobs. Additionally, time lag constraints arise in many real-life scheduling applications. For example, in the steel industry, the time lag between the heating of a piece of steel and its moulding should be small [3]. Similarly when scheduling chemical reactions, the reactive usually cannot be stored for a long time between two stages of a process to avoid interactions with external elements [4]. When these two factors are considered simultaneously, the problem is known as the job shop scheduling problem with time lags and sequence-dependent setup times (SDST–JSPTL).

The classical JSP with makespan minimization has been intensely studied, and many results have been established that have given rise to very efficient algorithms, for example the *i*-TSAB algorithm proposed by Nowicki and Smutnicki in [5].

Incorporating sequence-dependent setup times changes the nature of scheduling problems, and the well-known results and techniques for the JSP are not directly applicable anymore. The job shop with setups (SDST–JSP) is studied in [6], where Brucker and Thiele developed a branch and bound algorithm. Cheung and Zhou in [7] use a genetic algorithm combined with two dispatching rules. Also, in [8] Balas et al. propose a shifting bottleneck heuristic. One of the most efficient proposals is the branch and bound algorithm proposed by Artigues and Feillet in [9]. More recently, in [10] and [11] Vela et al. and Gonzalez et al. design two hybrid approaches that combine a genetic algorithm with local search procedures, taking some ideas proposed by Van Laarhoven et al. in [12] as a basis for new neighborhood structures. The maximum lateness minimization was also considered by some researchers [13,14].

The addition of time lag constraints between successive job operations makes difficult even the usually simple task of finding a feasible schedule. The job shop with time lags is a very challenging problem for local search metaheuristics, because the classical neighborhood structures for the standard job shop lead to unfeasible schedules most of the time. Very few articles tackle time-lag constraints. For example, Wikum et al. in [15] study problems with a single machine considering minimum and maximum distances between jobs and prove that these problems are NP-hard, although some particular cases are polynomially solvable. Brucker et al. in [16]

* Corresponding author. Tel.: +34 985182493.
E-mail addresses: mig@uniovi.es (M.A. González), angelo.oddi@istc.cnr.it (A. Oddi), riccardo.rasconi@istc.cnr.it (R. Rasconi), ramiro@uniovi.es (R. Varela).

http://dx.doi.org/10.1016/j.cor.2015.02.005
0305-0548/

show many examples of scheduling problems that can be modelled as single-machine problems with time-lags, like multi-processor tasks, multi-purpose machines or problems with changeover costs. They also propose a branch-and-bound method to solve the problem. Hurink and Keuchel, in [17], propose a local search approach for the single machine problem. Fondrevelle et al. in [18] and Dhouib et al. in [19] solve permutation flow-shop scheduling problems with maximal and minimal time lags. Botta-Genoulaz [20] tackles the maximum lateness minimization in the hybrid flow shop scheduling with precedence constraints and time lags. Another example is the thesis of Zhang [21], where several variants of online and offline problems with time-lags are studied.

The job shop with minimum 5and maximum time lags (JSPTL) was tackled by Caumond et al. [22], where they propose a list scheduling heuristic for generating initial solutions and a memetic algorithm based on a disjunctive graph model. Artigues et al. [23] study the JSPTL as well. They propose a job insertion heuristic for generating initial solutions and generalized resource constraint propagation mechanisms, which are embedded in a branch-and bound algorithm. Recently, Grimes and Hebrard [24] propose a constraint programming approach that uses a number of generic SAT and AI techniques such as weighted degree variable ordering, solution guided value ordering, geometric restarting and nogood recording from restarts. Their approach was adapted to solve several variants of the job shop scheduling problem, including the JSPTL. In [25] and [26], the authors propose constraint propagation methods for the job shop with generic time-lags, which is a generalized version of the JSPTL. Additionally, since the JSPTL is a particular case of the resource-constrained project scheduling problem with time lags (RCPSP/max), literature on this problem is also worth mentioning, for example [27–30].

We have already seen that there are several papers dealing with each of the two factors separately (setups and time lags). However, very few papers have considered both minimum and maximum time lags and sequence-dependent setup times at the same time (SDST–JSPTL). As far as we know, the only approach is that reported in [31], where Oddi et al. solve this problem by means of a constraint-based iterative sampling procedure.

Hybrid metaheuristics usually perform very well at solving combinatorial optimization problems, since they allow algorithm designers to combine the advantages of different search techniques. In particular, they have a long track of success with scheduling problems. Even for the classical JSP researchers continue to propose hybrid metaheuristics that outperform previous ones. For example, in [32] a Hybrid Genetic Tabu Search for solving the JSP is proposed. Also, the algorithms proposed in [33] and in [34] are probably the most efficient approaches to the JSP with makespan minimization. These algorithms combine the *i*-TSAB algorithm proposed in [5] with a simulated annealing in the first case and with a solution-guided search method in the second case.

In this paper we propose to solve the SDST–JSPTL with a scatter search algorithm which uses path relinking and tabu search in its core. This hybrid metaheuristic can obtain the best results in some scheduling problems, e.g. [35]. We propose a novel neighborhood structure, establish feasibility and non-improving conditions, and give an algorithm for fast estimation of neighbors' quality. We consider both feasible and unfeasible solutions through the search and the neighborhood structure defines a different subset of moves depending on whether the current solution is feasible or unfeasible. We also extend the heuristic proposed in [23] for generating initial solutions and the longest path algorithm defined in [22]. We conducted an experimental study in which we first analyzed the proposed algorithm, and then we compared it with the state-of-the-art in both the SDST–JSPTL and the standard JSPTL, showing that our results are very competitive.

The remainder of the paper is organized as follows. In Section 2 we formulate the problem and describe the solution graph model. In Section 3 we describe the heuristic used to generate the initial schedules. Section 4 details the algorithm used to calculate a schedule from a processing order. In Section 5 we define the proposed neighborhoods. Section 6 details the metaheuristics used. In Section 7 we report the results of the experimental study, and finally Section 8 summarizes the main conclusions of this paper.

2. The job shop scheduling problem with time lags and setup times

In this section we introduce the SDST–JSPTL. Firstly, we give a formal definition of the problem as an extension of the classic JSP and then we propose a disjunctive model that allows us to represent problem instances and schedules, as well as to formalize the neighborhood structures.

2.1. Problem formulation

In the SDST–JSPTL, we are given a set of n jobs, $J=\{J_1,\ldots,J_n\}$ that must be processed on a set of m machines or resources, $M=\{M_1,\ldots,M_m\}$, subject to a set of constraints. There are *precedence constraints*, so each job J_i, $i=1,\ldots,n$, consists of N_i operations $O_i=\{o_{i1},\ldots,o_{iN_i}\}$ to be sequentially scheduled. Also, there are *capacity constraints*, upon which an operation o_{ij} requires the uninterrupted and exclusive use of given machine $m_{ij}\in M$ during p_{ij} time units.

We also consider the addition of sequence-dependent setup times which depend on both the outgoing and incoming operations. After an operation o_{ij} leaves the machine and before an operation o_{kl} enters the same machine, a setup operation is required to adjust the machine, with duration $s_{o_{ij}o_{kl}}$. We also define an initial setup time $s_{0o_{ij}}$ required when o_{ij} is the first operation scheduled on its machine. We consider that the setup times verify the triangle inequality, i.e., $s_{uv}+s_{vw}\geq s_{uw}$ holds for any operations u, v and w requiring the same machine. This condition is usually assumed in the literature as it usually happens in real scenarios.

Finally, we consider the addition of minimum and maximum time lags between consecutive operations of the same job. If we have the operation o_{ij} (which is not the last operation of its job), we denote by $\mathrm{TL}_{min}(o_{ij})$ and $\mathrm{TL}_{max}(o_{ij})$ the minimum and maximum difference between the ending time of o_{ij} and the starting time of o_{ij+1}.

A solution to this problem consists of an allocation of starting times for each operation in the set $O=\bigcup_{1\leq i\leq n}O_i$ which is *feasible* (i.e. all constraints hold). The objective is to find an *optimal* solution according to some criterion, most commonly that the *makespan*, which is the completion time of the last operation to finish, is minimal.

2.2. Disjunctive graph model

We propose a disjunctive model for the SDST–JSPTL which extends the model proposed in [22] to cope with setup times. In accordance with this model, we have a directed graph $G=(V,A\cup E\cup I_1\cup I_2\cup M)$ where

- Each node in set V represents an operation of the problem, with the exception of the dummy nodes *start* and *end*, which represent fictitious operations with processing time 0 that do not require any machine.
- A is the set of *conjunctive arcs* and represents precedence relations. It contains arcs of the form (v,w), where v is the predecessor of w in its job sequence, weighted with $p_v+\mathrm{TL}_{min}(v)$.