Applied Soft Computing 10 (2010) 703–710

Contents lists available at ScienceDirect

Applied Soft Computing

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

A high performing metaheuristic for job shop scheduling with sequence-dependent setup times

B. Naderi, S.M.T. Fatemi Ghomi *, M. Aminnayeri

Department of Industrial Engineering, Amirkabir University of Technology, 424 Hafez Avenue, Tehran, Iran

ARTICLE INFO

Article history:
Received 27 May 2008
Received in revised form 7 April 2009
Accepted 28 August 2009
Available online 6 September 2009

Keywords:
Scheduling
Job shop
Sequence-dependent setup times
Simulated annealing
Taguchi method

ABSTRACT

This paper investigates scheduling job shop problems with sequence-dependent setup times under minimization of makespan. We develop an effective metaheuristic, simulated annealing with novel operators, to potentially solve the problem. Simulated annealing is a well-recognized algorithm and historically classified as a local-search-based metaheuristic. The performance of simulated annealing critically depends on its operators and parameters, in particular, its neighborhood search structure. In this paper, we propose an effective neighborhood search structure based on insertion neighborhoods as well as analyzing the behavior of simulated annealing with different types of operators and parameters by the means of Taguchi method. An experiment based on Taillard benchmark is conducted to evaluate the proposed algorithm against some effective algorithms existing in the literature. The results show that the proposed algorithm outperforms the other algorithms.

1. Introduction

Job shop scheduling (or JSS) is one of the most complicated combinatorial optimizations. A JSS could be described as follows: we have a set of n jobs need to be operated on a set of m machines [1]. Each job has its own processing route; that is, jobs visit machines in different orders. Each job might need to be performed only on a fraction of m machines, not all of them. The following assumptions are additionally characterized. Each job can be processed by at most one machine at a time and each machine can process at most one job at a time. When the process of an operation starts, it cannot be interrupted before the completion; that is, the jobs are non-preemptive. There is no transportation time between machines; in other words, when an operation of a job finishes, its operation on subsequent machine can immediately begin. The jobs are independent; that is, there are no precedence constraints among the jobs and they can be operated in any sequence. The jobs are available for their process at time 0. There is unlimited buffer between machines for semi-finished jobs; meaning that if a job needs a machine that is occupied, it waits indefinitely until it becomes available. There is no machine breakdown (i.e. machines are continuously available). The objective function when solving or optimizing a JSS is to determine the processing order of all jobs on each machine that minimizes the makespan.

Numerous savings obtained by considering setup times in scheduling decisions prompted researchers to utilize this assumption in their studies [2]. Setup times are typically sequence-dependent (or SDST), that is, the magnitude of setup strongly depends on both current and immediately processed jobs on a given machine. For example, this may occur in a painting operation, where different initial paint colors require different levels of cleaning when being followed by other paint colors. We also assume that setup is non-anticipatory, meaning that the setup can only begin as soon as the job and the machine are both available. The sequence-dependent setup time job shop scheduling (SDST JSS) is defined as J/STsd/Cmax according to three-fold notation of Graham et al. [41].

The JSS is known to be an NP-hard optimization problem [3]. Therefore, effective metaheuristics for the JSS are necessary to find optimal or near optimal solutions in reasonable amount of time. This paper proposes such an algorithm, in the form of simulated annealing (or SA), for the problem under consideration. Many researchers in the field of scheduling conclude that SAs show inferior performance in comparison with other metaheuristics [4]; however, SAs have recently proved their efficiency and effectiveness in a wide variety of optimization problems [4–6]. It is known that the performance of SAs strongly depends on the choice of its operators and parameters. Hence, beside presenting our operators, we explore the impact of different operators and parameters on the performance of SA by means of Taguchi method. Taguchi method is an optimization technique that brings robustness into experimental designs as well as being a cost-effective and labor-saving method [7,8]. It can simultaneously investigate several factors and

* Corresponding author. Tel.: +98 21 66413034; fax: +98 21 66413025.
E-mail address: fatemi@aut.ac.ir (S.M.T. Fatemi Ghomi).

1568-4946/$ – see front matter
doi:10.1016/j.asoc.2009.08.039

define quickly those having significant effects on response variables by conducting a minimal number of possible experiments [9]. There are many successful applications of this approach at the parameter design stage in many fields [10].

The reminder of the paper is organized as follows. Section 2 reviews the literature of SDST JSS. Section 3 describes the proposed simulated annealing. The calibration of the proposed algorithm is presented in Section 4. In Section 5, experimental design and comparison of the proposed algorithm with existing methods are reported. Section 6 concludes the paper and provides some directions for future studies.

2. Literature review

The consideration of sequence-dependent setup times in scheduling problems has developed into a very active field of research in the recent years. Many review papers list the researches that are conducted on SDST scheduling problems. Among all, we can point out to survey papers of [2,11–13]. In [11–13], papers published before 2000 are cited while in [2] papers after 2000 are covered. The importance of this consideration is comprehensively explored in [14]. Coleman [15] proposes an integer programming model for minimizing earliness/tardiness in a single machine with sequence-dependent setups. He also shows that SDST single machine is strongly NP-hard. Most of researches in production scheduling with SDST are restricted to the flowshop and its extensions, such as the hybrid flowshop scheduling [16,17].

With regard to SDST JSS, Kim and Bobrowski [18] group and evaluate scheduling rules in dynamic scheduling environments defined by due date tightness, setup times and cost structure. They use a simulation model of a nine machine job shop for the experiment. Choi and Korkmaz [19] explore JSS with anticipatory SDST, i.e. to begin the setup, only machine need to be idle. They formulate the problem as a mixed integer program and propose a heuristic based on consecutively recognizing a pair of operations which gives a minimum lower bound on the makespan of the associated two-job/m-machine problem with release times. They also show that the proposed heuristic is more effective than one proposed by [20]. Regarding enumeration algorithms, three different branch and bound methods are presented in [21–23].

Schutten [24] addresses the job shop with some practical aspects, such as release and due date, setup times and transportation times. He then proposes an extension of the shifting bottleneck procedure for the problem. Sun and Noble [25] consider JSS with release dates, due dates, and sequence-dependent setup times to minimize the weighted sum of squared tardiness. They decompose the problem into a series of single-machine scheduling problems within a shifting bottleneck framework. The single-machine scheduling problem is solved using a lagrangian relaxation-based approach. They compare the proposed algorithm against some dispatching rules including "Earliest Due Date", "Apparent Tardiness Cost" and "Similar Setup Times".

Cheung and Zhou [26] develop a hybrid algorithm based on a genetic algorithm and a well-known dispatching rule for SDST JSS where the setup times are anticipatory. The first operation for each of the m machines is obtained by the genetic algorithm while the subsequent operations on each machine are scheduled according to the shortest processing time (or SPT) rule. Choi and Choi [27] study JSS with alternative operations and SDST. They provide a mixed integer program as well as a local-search scheme which incorporates a speed-up feature. Artigues and Roubellat [28] provide a polynomial insertion algorithm for multi-resource SDST JSS under minimization of maximum lateness. They first describe the algorithm for pure JSS, and then introduce multi-resource requirements for the operations. Finally, SDSTs are integrated in the multi-resource context. Using some dominance properties, they show that the proposed insertion algorithm outperforms the alternative enumeration algorithms.

Sun and Yee [29] address JSS with reentrant work flows and SDST to minimize makespan. They use the disjunctive graph representation to study interactions between machines. For this representation, four two-phase heuristics are proposed. They also introduce a genetic algorithm employing an efficient local improvement procedure. Artigues et al. [30] study SDST JSS with concentration on formal definition of schedule generation schemes (SGSs) based on the semi-active, active, and non-delay schedule categories. They also review some priority rules and present a comparative computational analysis of the different SGSs on sets of instances taken from the literature. Zhou et al. [31] address SDST JSS and propose an immune algorithm which certifies the diversity of the antibody.

Manikas and Chang [32] consider SDST JSS and present a scatter search combining the mechanisms of diversification and intensification. To evaluate the proposed algorithm, they compare it with a simple tabu search, simulated annealing, and a genetic algorithm. Naderi et al. [33] propose a hybrid algorithm that shows high performance in comparison with other algorithms in the literature of SDST JSS. This hybrid algorithm is a genetic algorithm incorporating some additional features, namely restart phase and local search. Many genetic operators and parameters are evaluated in this paper.

Vinod and Sridharan [34] address dynamic SDST JSS and develop a discrete event simulation model of the job shop. Two types of scheduling rules (ordinary and setup-oriented rules) are applied in simulation model. Their experimental results demonstrate that setup-oriented rules outperform ordinary ones. This difference rises with the increase in shop load and setup time ratio. Roshanaei et al. [35] employ a variable neighborhood search to solve SDST JSS. Their metaheuristic employs three different neighborhood search structures centered on insertion operator concept.

3. Simulated annealing

Simulated annealing (SA) is a local-search-based metaheuristic which has exhibited some promise when it is applied to NP-hard problems [4–6]. A typical SA starts from an initial solution and proceeds sequentially and slowly toward the area that might be far from the search area of the initial solution. The SA accepts moves to inferior neighboring solution under the control of a randomized scheme to reduce the probability of getting trapped in local optima. It seems that the performance of SA is strongly determined by precise calibration of its operators and parameters. In the following subsections we describe all parameters and operators used in the proposed SA.

3.1. Encoding scheme and initialization

Encoding schemes are used to make a candidate solution recognizable for algorithms. A proper encoding scheme plays a key role in maintaining the search effectiveness of any algorithms. One of the most extensively used encoding schemes in the literature is the operation-based representation [36]. By making use of this representation, the relative order of the operations of the jobs on the machines on which they are processed is determined. Since there are precedence constraints among the operations of each job, not all the permutations of the operations give feasible solutions. With respect to the above explanations, Gen et al. [37] propose an alternative scheme which is as follows: Each job i has a set of n_i operations. So in the representation, each job number i occurs as the number of its operations. By scanning the permutation from left to right, the kth occurrence of a job number refers to the kth