Computers & Operations Research 64 (2015) 86–96

Contents lists available at ScienceDirect

Computers & Operations Research

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

Efficient non-population-based algorithms for the permutation flowshop scheduling problem with makespan minimisation subject to a maximum tardiness

Victor Fernandez-Viagas *, Jose M. Framinan

Industrial Management, School of Engineering, University of Seville, Ave. Descubrimientos s/n, E41092 Seville, Spain

ARTICLE INFO

Available online 27 May 2015

Keywords:
Scheduling
Flowshop
Heuristics
NEH
PFSP
Maximum tardiness
Makespan
Bounded insertion
Non-population algorithm

ABSTRACT

This paper focuses on the problem of scheduling jobs in a permutation flowshop with the objective of makespan minimisation subject to a maximum allowed tardiness for the jobs, a problem that combines two desirable manufacturing objectives related to machine utilisation and to customer satisfaction. Although several approximate algorithms have been proposed for this NP-hard problem, none of them can use the excellent speed-up method by Taillard (1990) [22] for makespan minimisation due to the special structure of the problem under consideration. In this paper, several properties of the problem are defined in order to be able to partly apply Taillard's acceleration. This mechanism, together with a novel feasible tabu local search method, allows us to further exploit the structure of solutions of the problem, and are incorporated in two proposed algorithms: a bounded-insertion-based constructive heuristic and an advanced non-population-based algorithm. These algorithms are compared with state-of-the-art algorithms under the same computer conditions. The results show that both algorithms improve existing ones and therefore, constitute the new state-of-art approximate solution procedures for the problem.

1. Introduction

A permutation flow shop is a manufacturing layout in which a set of machines are arranged to be visited by a number of jobs in the same order, assuming that the sequence of the jobs remains the same for all machines. Usual additional hypotheses include the simultaneous availability of all jobs and all machines and deterministic processing times, among others (see e.g. Framinan et al. [10] for a complete list of assumptions). Several criteria can be established to measure the performance of the different schedules (see e.g. Sun et al. [21]). Among them, the maximum completion time of a sequence or makespan is related to resource usage (see e.g. Ruiz and Maroto [19] and Fernandez-Viagas and Framinan [7]), while tardiness refers to the delay of the completion time of a job with respect to its committed due date (see e.g. Ruiz and Maroto [5] and Fernandez-Viagas and Framinan [9]). Since these are key aspects in manufacturing companies' competitiveness, it seems appropriate to consider both objectives together. Regarding tardiness minimisation, customer due dates may be regarded as 'hard' constraints (i.e. deadlines) in some manufacturing scenarios, while in others some flexibility is allowed by the customer as long as the deviation from the completion times of the jobs is limited. In contrast, makespan is an intra-company criteria that is related to maximising machine utilisation, which in turns minimises fixed unit costs. Therefore, one option to balance both objectives is to seek the minimisation of the makespan while allowing only a given deviation from the committed due dates, expressed as the maximum tardiness allowed. Note that this problem includes the special case where no deviation from the jobs' due dates is allowed, thus forcing the fulfilment of the committed due dates.

According to the notation by T'Kindt and Billaut [24], the problem described in the previous paragraph can be denoted as $Fm|prmu|\epsilon(C_{max}/T_{max})$. This problem belongs to the class of ϵ-constrained multi-criteria scheduling problems, and it has been the subject of several research contributions in the last decades. Since the minimisation of any of the individual criteria (either makespan or maximum tardiness) in a flow shop is NP-hard, the research effort has focused on approximate procedures providing good – but not necessarily optimal – solutions in a relative short period of time. In this regard, the works by Daniels and Chambers [2], Chakravarthy and Rajendran [1], Framinan and Leisten [8], and Ruiz and Allahverdi [18] develop different heuristics either for the problem, or for general cases of the problem. In this paper, we propose a constructive heuristic and a metaheuristic that exploits the specific structure of solutions of the problem to reduce the search space and to accelerate the evaluation of solutions. Both

* Corresponding author. Tel.: +34 954487220.
E-mail addresses: vfernandezviagas@us.es (V. Fernandez-Viagas), framinan@us.es (J.M. Framinan).

http://dx.doi.org/10.1016/j.cor.2015.05.006
0305-0548/

algorithms improve existing ones by a larger degree and constitute therefore the new state-of-art approximate solution procedures for the problem.

The remainder of the paper is structured as follows: Section 2 describes the problem under consideration and its state-of-the-art. In Section 3, some definitions and properties of the problem are defined. Section 4 is devoted to propose two algorithms (a constructive heuristic and a metaheuristic) which use the properties discussed previously. The algorithms are compared with the (up to now) state-of-the-art algorithms in Section 4 and, finally, conclusions are discussed in Section 5.

2. Problem statement and state of the art

In the $Fm|prmu|\epsilon(C_{max}/T_{max})$ problem under study, n jobs have to be scheduled in a flowshop composed of m one-machine stage. The processing time of job l on machine i is defined as p_{il}. Following this notation, given a sequence of jobs $\Pi:=(\pi_1,\ldots,\pi_n)$, the processing time of job in position j, π_j, is denoted as $p_{i\pi_j}$.

Analogously, $C_{i\pi_j}(\Pi)$ denotes the completion time of job π_j on machine i according to the schedule given by Π. The makespan of the sequence Π is given by the completion time of last job in last machine, $C_{m,\pi_n}(\Pi)$, which is denoted as $C_{max}(\Pi)$. Whenever it does not lead to confusion, the sequence Π is omitted in the notation of the completion times and makespan.

In a similar manner, if d_{π_j} is the due date of job π_j, its tardiness is defined as $T_{\pi_j}(\Pi)=\max\{C_{m\pi_j}(\Pi)-d_{\pi_j},0\}$ and the maximum tardiness of the sequence Π as $T_{max}(\Pi)=\max_{j=1,\ldots,n}\{T_{\pi_j}(\Pi)\}$. As with the makespan, Π is omitted when it is clear from the context. The goal of the problem is to find a schedule Π for which the makespan is minimum subject to $T_{max}(\Pi)\le\epsilon$.

As mentioned in Section 1, the problem is NP-hard since the minimisation of each individual criterion is already an NP-hard problem for the permutation flowshop (see e.g. T'Kindt and Billaut [24] for a detailed proof). Consequently, the interest lies in finding efficient approximate methods or *heuristics*. Given the clear connection between our problem and that of makespan minimisation, most of the algorithms to solve the problem are based on the best heuristic for makespan minimisation: i.e. the NEH heuristic by Nawaz et al. [14]. It is then useful to recall the main steps in the NEH heuristic, which can be described as follows:

1. Jobs are ordered according to non-increasing sum of processing times.
2. A partial sequence is constructed only with first job of the previous phase.
3. Each remaining job of initial phase is iteratively inserted in all positions of the partial sequence. The makespan of all these sequences is evaluated, and the partial schedule for which the lowest makespan is reached is selected for the next iteration.
4. The procedure is repeated until no more jobs are available.

The above steps make clear that the computational burden of the NEH lies on the evaluation of all possible insertions in Step 3. In Taillard [22], a mechanism – named in the following *Taillard's acceleration* – is proposed so the computational complexity of evaluating all insertions is equivalent to that of evaluating one sequence. In order to explain Taillard's acceleration, let us first define three variables (for a more detailed description of the variables, see Taillard [22]):

- e_{i,π_j}: Earliest completion time of job in position j in machine i.
- q_{i,π_j}: Once a sequence of jobs has been defined (and therefore the makespan of this sequence is obtained), q_{i,π_j} is the difference between the makespan and the latest starting time of job in position j in machine i.
- f_{i,π_j}: Earliest completion time of the new job σ when it is inserted before job in position j in machine i. These are computed using e_{i,π_j} and the processing times of σ.

By means of these variables, the partial makespan C^j_{max} when introducing job σ before job in position j can be determined using the following expression:

$$C^j_{max}=\max_i(f_{i,\pi_j}+q_{i,\pi_j}) \quad (1)$$

As the job σ is inserted in the position with minimum makespan, the makespan of the sequence is defined by

$$C_{max}=\min_j(C^j_{max}) \quad (2)$$

As it can be seen, although the cost of evaluating the insertion slot with lowest makespan is greatly reduced by Taillard's acceleration, the completion time of each job cannot be obtained using this mechanism, and therefore its tardiness cannot be computed. As a consequence, none of the heuristics proposed up-to-now in the literature for the problem under consideration use this mechanism.

Among the contributions on the $Fm|prmu|\epsilon(C_{max}/T_{max})$ problem, Daniels and Chambers [2] were the first in proposing a constructive heuristic. In their heuristic, assuming a partial sequence Π formed by already scheduled jobs, a (partial) sequence is constructed for each non-scheduled job u_k by placing it as the first job, and then scheduling the jobs in Π after u_k according to the NEH algorithm. Out of these so-obtained sequences, the one with the lowest makespan is chosen for the next iterations (consequently, u_k is removed from the non-scheduled jobs set for the next iteration).

Chakravarthy and Rajendran [1] propose a simulated annealing algorithm to solve the $Fm|prmu|\epsilon(Z/T_{max})$ where $Z=\lambda\cdot C_{max}+(1-\lambda)\cdot T_{max}$, $\lambda\in[0,1]$. Clearly, our problem is a special case of their problem when $\lambda=1$. Their algorithm begins with the best sequence among the solutions found by the NEH heuristic, the earliest due date rule and the least slack rule (jobs ordered according to ascending order of $d_j-\sum_{i=1}^m p_{ij}$). The procedure iteratively samples neighbour solutions (using an adjacent pairwise interchange neighbourhood) until the stopping criterion is fulfilled.

Framinan and Leisten [8] propose a constructive heuristic, denoted in the following as *FL*, based on the NEH algorithm to solve the $Fm|prmu|\epsilon(C_{max}/T_{max})$ problem. The heuristic tries to improve the makespan without worsening the tardiness by using a property of the problem. The heuristic is compared with those of Daniels and Chambers [2] and Chakravarthy and Rajendran [1] for small and big instances. The results show that the *FL* outperforms the other ones in terms of both the quality of the solutions and the number of the feasible solutions obtained.

Finally, Ruiz and Allahverdi [18] propose an iterated optimisation algorithm to solve the $Fm|prmu|\epsilon(Z/T_{max})$ problem. More specifically, they proposed a high-performance Genetic Algorithm (*GA* in the following) where the selection procedure is based on n-tournament (see Ruiz and Allahverdi [17]). The fitness values of the individuals are calculated depending on whether all individuals are feasible; feasible and infeasible; or only infeasible. The algorithm outperforms the *FL* for the $Fm|prmu|\epsilon(Z/T_{max})$ problem in an extended benchmark. Nevertheless, *GA* and *FL* were not compared for the specific $Fm|prmu|\epsilon(C_{max}/T_{max})$ problem.

To summarise the state of the art regarding the problem under consideration, there are some efficient heuristics for the problem, but their performance is not completely clear, as the comparison between the most efficient contributions (i.e. *GA* and *FL*) has been only partially conducted. In addition, both mechanisms made