Omega 38 (2010) 57 – 67

Contents lists available at ScienceDirect

Omega

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

Genetic algorithms with path relinking for the minimum tardiness permutation flowshop problem☆

Eva Vallada*, Rubén Ruiz

Grupo de Sistemas de Optimización Aplicada, Instituto Tecnológico de Informática, Universidad Politécnica de Valencia, Ciudad Politécnica de la Innovación, Edificio 8G, Acc. B. Camino de Vera S/N, 46021 Valencia, Spain

ARTICLE INFO

Article history:
Received 17 December 2007
Accepted 15 April 2009
Available online 5 May 2009

Keywords:
Flowshop
Tardiness
Genetic algorithm
Path relinking
Diversity

ABSTRACT

In this work three genetic algorithms are presented for the permutation flowshop scheduling problem with total tardiness minimisation criterion. The algorithms include advanced techniques like path relinking, local search and a procedure to control the diversity of the population. We also include a speed up procedure in order to reduce the computational effort needed for the local search technique, which results in large CPU time savings. A complete calibration of the different parameters and operators of the proposed algorithms by means of a design of experiments approach is also given. We carry out a comparative evaluation with the best methods that can be found in the literature for the total tardiness objective, and with adaptations of other state-of-the-art methods originally proposed for other objectives, mainly makespan. All the methods have been implemented with and without the speed up procedure in order to test its effect. The results show that the proposed algorithms are very effective, outperforming the remaining methods of the comparison by a considerable margin.

1. Introduction

The permutation flowshop scheduling problem (PFSP) is a widely studied combinatorial optimisation problem where there is a set $N=\{1,\ldots,n\}$ of n jobs that have to be processed on a set $M=\{1,\ldots,m\}$ of m machines. In the PFSP all jobs visit the machines in the same order which can be assumed to be $1,2,\ldots,m$. Therefore, each job is made up of m tasks. Furthermore, in the PFSP once the job sequence is established, it is not changed from machine to machine. This yields $n!$ job permutations and possible processing sequences. Each operation requires a given processing time denoted by p_{ij}. The most commonly studied objective in the literature is to find a minimum completion time sequence, a criteria that is known as makespan or C_{max}. According to Pinedo [1] this problem is denoted as $F/prmu/C_{max}$ and the associated decision problem was shown to be $\mathcal{NP}$-complete by Rinooy Kan [2] for $m \geq 3$.

In the literature, there is a large body of papers dealing with the PFSP and the makespan objective. Some of the most recent methods proposed for this problem are those presented by Ruiz et al. [3], Grabowski and Pempera [4] and Farahmand et al. [5]. We can find in Ruiz and Maroto [6] an updated extensive comparison and evaluation. Other recent reviews centred around makespan criterion can be found in Framinan et al. [7] or Hejazi and Saghafian [8]. Recent research has also focused on other objectives, especially those related to due dates. These objectives are important in real-life, mainly in industry where the fulfillment of due dates agreed with customers is of uttermost importance. Among due date based criteria, the minimisation of the total tardiness is probably the most common one. Therefore, the PFSP with this criterion is denoted as $F/prmu/\sum T_j$ [1] where $T_j = \max\{C_j - d_j, 0\}$ is the *tardiness* of job j, being d_j its *due date* and C_j its completion time at the last machine of the shop. Du and Leung [9] showed that the problem is $\mathcal{NP}$-hard in the ordinary sense even when there is only one machine and $\mathcal{NP}$-hard in the strong sense for $m \geq 2$.

Recently, Vallada et al. [10] reported an exhaustive review and comparative evaluation of different methods for the PFSP minimising total tardiness. In this review, it was shown that it is not common to adapt recent, high performing methods from the makespan to the total tardiness objective. As it was shown, only a few classic algorithms for the makespan objective are usually adapted to the total tardiness objective [11–13].

Given that the makespan objective is the most studied, it seems plausible to take advantage of this research when developing new algorithms for total tardiness minimisation. Like for example the advanced genetic algorithms (GAs) depicted by Ruiz et al. [3] for the makespan criterion. Working with such methods for other objectives seems promising. Furthermore, genetic algorithms have been

☆This article was processed by Associate Editor Lev.

* Corresponding author. Tel.: +34 96 387 70 07x74911; fax: +34 96 387 74 99.
E-mail addresses: evallada@eio.upv.es (E. Vallada), rruiz@eio.upv.es (R. Ruiz).

0305-0483/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.omega.2009.04.002

sparsely used for the total tardiness objective. Therefore, the main goal of this paper is to propose high performing genetic algorithms for the PFSP minimising total tardiness. These algorithms make extensive use of advanced techniques like local search, diversification and path relinking (PR). The resulting algorithms show excellent performance when compared against recent proposed methods under careful and comprehensive computational and statistical experiments.

The remainder of this paper is organised as follows. In Section 2 we review the literature on this problem. In Section 3 we describe in detail the genetic algorithms proposed. In Section 4, a design of experiments (DOE) approach is applied in order to calibrate the genetic algorithms. Results of a comparative computational and statistical evaluation are reported in Section 5. Finally, conclusions are given in Section 6.

2. Literature review

Vallada et al. [10] provided an extensive literature review along with a comparative evaluation about the PFSP with the objective of total tardiness minimisation. In this section we review the most important existing methods, from the classical exact approaches to the most recent and effective metaheuristic algorithms.

Exact methods for the PFSP minimising total tardiness are mainly focused in the two machine case. There exist several papers where branch and bound algorithms for this case are proposed [14–18]. The best performing algorithms among those presented in the aforementioned papers are able to solve instances of up to 18 or 20 jobs maximum. Regarding the m-machine case, we only find two papers, Kim [19] and more recently, Chung et al. [20]. The former solved all the instances of up to 13 jobs and eight machines and some instances of up to 14 jobs and four machines. The latter solved optimally all instances up to 15 jobs and two machines and some with 20 jobs and eight machines. As we can see, only the most recent exact methods are able to solve some problems of up to 20 jobs and eight machines. For this reason, research is mainly focused on the development of heuristic and metaheuristic methods to tackle this complex problem.

Some of the earlier heuristics are the four proposed by Gelders and Sambandam [12] based on priority rules. A few years later we find the paper by Ow [21] where a heuristic based on a priority rule is proposed for the proportionate flowshop problem. In [22] some adaptations of algorithms for different objectives, mainly the makespan, are presented. We want to remark the NEH method [13], which is considered the best heuristic for the makespan objective, according to Ruiz and Maroto [6] and Kalczynski and Kamburowski [23]. In the adapted version of Kim [22], jobs are initially sorted following the earliest due date (EDD) rule, that is, in non-decreasing order of due dates and according to the author we name this NEH version as NEH_{edd}. We can also find in [24] several rules and heuristics originally proposed for the one machine and two machine cases and adapted to the m-machine case. Finally, Kim et al. [25] reported several improvement heuristics which start from the NEH_{edd} method [22] and local search procedures based on insertion, interchange and permutation of jobs are applied to improve the solution.

With respect to the metaheuristic methods, some of the earliest existing papers are Adenso-Díaz [26] and Kim [22]. In both papers we find adapted versions of the well-known tabu search by Widmer and Hertz [27] originally proposed for the makespan objective. The former starts from the solution provided by Ow [21] and a restricted neighbourhood is applied. The latter starts from the *earliest due date* rule. Later, the same authors extended these results in [28,25]. In this case, Adenso-Díaz [28] proposed a hybrid algorithm based on simulated annealing and tabu search starting from the solution provided by Ow [21]. Kim et al. [25] devised four tabu search and four simulated annealing methods all of them starting from the NEH_{edd} method. In two related papers [29,30], two very similar simulated annealing methods for the flowshop problem with sequence dependent setup times and the objective to minimise the mean weighted tardiness are proposed. Another two simulated annealing algorithms are those reported by the same authors in [31] which start from a specific rule and perturbation schemes are applied to improve the initial solution. In [32], a genetic algorithm is proposed which considered three objective functions: minimising total tardiness, minimising number of tardy jobs and minimising both objectives at the same time. In [33], a basic tabu search was proposed and then diversification, intensification and restricted neighbourhood strategies are applied to form four tabu search variations. Another work is that by Rajendran and Ziegler [34] where heuristic and metaheuristic methods are proposed with the objective of minimising the sum of weighted flowtime and weighted tardiness. A simulated annealing method can be found in Hasija and Rajendran [35] which starts from the solution provided by Parthasarathy and Rajendran [31] and then applies a local search procedure to improve it. After this, the simulated annealing method itself is applied to improve this initial solution and two local search procedures are considered, the first one is very similar to that presented in [31] and the second one is based on the interchange of jobs. This simulated annealing shows the best performance for this problem up to the moment according to the results of Vallada et al. [10]. Finally, in a recent work, Onwubolu and Davendra [36] proposed a differential evolution algorithm to minimise the makespan, flowtime and total tardiness and Ronconi and Henriques [37] presented heuristics for the flowshop problem with blocking.

It is important to point out that most of the aforementioned published papers only compare results with a few classical algorithms for the same problem and the experiments were carried out using different benchmarks of instances of up to 100 jobs and 20 machines maximum in the best case. For more details about the performance of all these methods see [10].

3. Proposed genetic algorithms

Genetic algorithms are bio-inspired optimisation methods that are widely used to solve combinatorial problems such as the PFSP. There is a rich literature where GAs are successfully applied to this problem (see for example [3,38–42], among others). However, as we have mentioned in the previous section, despite showing a very good performance for the makespan objective, only one paper can be found in the literature for the total tardiness objective [32]. In our proposed algorithms, among other innovative features, we add path relinking techniques and a procedure to control the diversity of the population. We have studied all these characteristics in a series of calibration experiments in order to set the best operators and parameter values. In the next subsections a detailed description about the algorithms is reported.

3.1. Representation of solutions, initialisation of the population and selection operator

The most commonly used solution representation for the PFSP is a simple permutation of jobs that indicates the processing order by the machines. With this representation it is easy to construct an active schedule by sequencing the first job of the permutation on all m machines, then the second, and so on until all n jobs are scheduled. The GAs are formed by a population of P_{size} individuals or n job permutations.

It is also common to randomly generate the initial population in a genetic algorithm. However, a recent trend consists in including in the population some good individuals provided by some effective