Computers & Operations Research 60 (2015) 27–36

Contents lists available at ScienceDirect

Computers & Operations Research

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

NEH-based heuristics for the permutation flowshop scheduling problem to minimise total 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 20 February 2015

Keywords:
Scheduling
Flowshop
Heuristics
NEH
Tie-breaking mechanism
Tardiness
PFSP

ABSTRACT

Since Johnson's seminal paper in 1954, scheduling jobs in a permutation flowshop has been receiving the attention of hundreds of practitioners and researchers, being one of the most studied topics in the Operations Research literature. Among the different objectives that can be considered, minimising the total tardiness (i.e. the sum of the surplus of the completion time of each job over its due date) is regarded as a key objective for manufacturing companies, as it entails the fulfilment of the due dates committed to customers. Since this problem is known to be NP-hard, most research has focused on proposing approximate procedures to solve it in reasonable computation times. Particularly, several constructive heuristics have been proposed, with NEHedd being the most efficient one, serving also to provide an initial solution for more elaborate approximate procedures. In this paper, we first analyse in detail the decision problem depending on the generation of the due dates of the jobs, and discuss the similarities with different related decision problems. In addition, for the most characteristic tardiness scenario, the analysis shows that a huge number of ties appear during the construction of the solutions done by the NEHedd heuristic, and that wisely breaking the ties greatly influences the quality of the final solution. Since no tie-breaking mechanism has been designed for this heuristic up to now, we propose several mechanisms that are exhaustively tested. The results show that some of them outperform the original NEHedd by about 25% while keeping the same computational requirements.

1. Introduction

The Permutation Flowshop Scheduling Problem (denoted as PFSP) is one of the most studied problems in the Operations Research literature (see the reviews by [4,27]). This decision problem deals with scheduling n jobs that have to be processed on m machines in the same order and with the same job sequence on every machine. Different criteria have been considered in the literature for this decision problem (see e.g. the reviews by [6,32,19]), such as the maximum completion time among the jobs or makespan, the total flowtime (sum of completion times of all jobs), and the total tardiness (sum of the tardiness of each job). Makespan and total completion time are related to the fast processing of the products and to a balanced use of resources, both issues being of great importance in make-to-stock manufacturing scenarios. In contrast, total tardiness focuses on the satisfaction of customers and it is therefore better suited for make-to-order manufacturing scenarios as due dates play a key role [21,15]. Thereby, among the objectives considered for the PFSP, the total tardiness highlights a critical concern for manufacturing systems (see e.g. [25,20]), since delays may lead to an increase in costs such as penalty clauses in a contract, loss of customers and/or bad reputation for other customers [29]. The PFSP with total tardiness minimisation objective is denoted as $Fm|prmu|\sum T_j$ (see e.g. [22]).

Since the problem is known to be NP-hard, most researchers have focused on developing solution procedures (i.e. heuristics) that do not guarantee the optimality of the solution, but that can provide a (hopefully) good solution in a reasonable time interval. More specifically, several heuristics and metaheuristics have been proposed in the literature for the $Fm|prmu|\sum T_j$ problem, such as those by e.g. [8,16,24,5,31]. Among them, the NEHedd heuristic [14] stands out since, as we will discuss later, many works employ it to obtain an initial solution. The NEHedd is an adaptation for the tardiness objective of the well-known NEH heuristic by Nawaz et al. [18] for makespan minimisation. In the NEH heuristic, jobs are initially arranged in non-ascending order of their processing times. Then, a job sequence is constructed by evaluating the partial schedules originating from the initial order: assuming a sequence already set for the first $k-1$ jobs, k candidate (sub)sequences are obtained by inserting job k in the k possible slots of the current

* 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.2015.02.002
0305-0548/

sequence. Out of these k (sub)sequences, the one yielding the minimum makespan is kept as the relative (sub)sequence for these first k jobs. Then, job $k+1$ from the initial order is considered analogously, and so on until all n jobs have been sequenced. In order to reduce the computational burden of the NEH heuristic, Taillard [30] proposed a mechanism (known as Taillard's acceleration) to reduce the complexity of the NEH heuristic from $n^3 \cdot m$ to $n^2 \cdot m$. The excellent performance of the NEH heuristic and its easy adaptation to similar problems have led to its application to other scheduling decisions, such as the PFSP with total completion time minimisation (see e.g. [7]), denoted as $Fm|prmu|\sum C_j$, or the hybrid flowshop scheduling problem (see e.g. [1]). For these problems, Taillard's acceleration cannot be applied, but Li et al. [17] present a mechanism that saves between 30 and 50% of CPU time for the $Fm|prmu|\sum C_j$ problem, however without reducing its complexity.

The NEHedd heuristic differs from the NEH heuristic in the starting order (jobs are arranged now according to the Earliest Due Date or EDD rule), and in the evaluation of the partial sequences (as the one with lowest total tardiness is selected). Taillard's acceleration cannot be applied to the NEHedd, and, although Vallada and Ruiz [31] propose a mechanism similar to that by Li et al. [17], the complexity of the NEHedd remains $O(n^3 \cdot m)$.

The extensive computational evaluation of heuristics for the $Fm|prmu|\sum T_j$ problem carried out by Vallada et al. [32] shows that NEHedd is a key constructive heuristic for the problem since, aside to being very efficient, the rest of efficient heuristics in the literature with more average CPU time employ NEHedd as an initial solution. More specifically, more than half of the state-of-the-art improvement heuristics or metaheuristics for the problem use NEHedd as a starting solution. This fact can be also seen in more recent works, such as Vallada and Ruiz [31], or Schaller [28].

Despite the excellent performance of the NEHedd heuristic, we believe that additional improvements could be gained by further analysis of the problem under consideration. First, the tardiness minimisation problem could resemble different scheduling problems depending on the due dates of the jobs for each specific instance: intuitively, it is clear that, for an instance with due dates much greater than the sum of the processing times of its jobs, almost every schedule may yield zero total tardiness, thus turning the problem into a trivial one. Analogously, unachievable due dates for each job results in an instance for which almost every sequence yields tardiness for every job and therefore the problem resembles that of minimising flowtime. By conducting an analysis of these possible scenarios, further insights on the problem can be obtained, so the performance of the NEHedd procedure can be enhanced. More specifically, we will show in this paper that such an analysis reveals the importance of adequately addressing the high number of ties appearing in the constructive phase of the NEHedd. In order to handle these ties in an efficient way, we propose several tie-breaking mechanisms for the problem and conduct an extensive computational experiment to test their performance. The results show that one of these mechanisms (based on machine idle time) improves the original results obtained by NEHedd by roughly 25% while requiring the same CPU time. Another one (based on Taillard's acceleration for makespan) outperforms the NEHedd by 15% while employing less CPU time. Furthermore, when using the idle time-based version of the NEHedd as starting solution for the metaheuristic by Vallada and Ruiz [31] (which is the state-of-the-art metaheuristic for the problem), the metaheuristic improves its result for different stopping criteria.

The remainder of the paper is organized as follows: in Section 2 the problem under consideration is described and analysed to

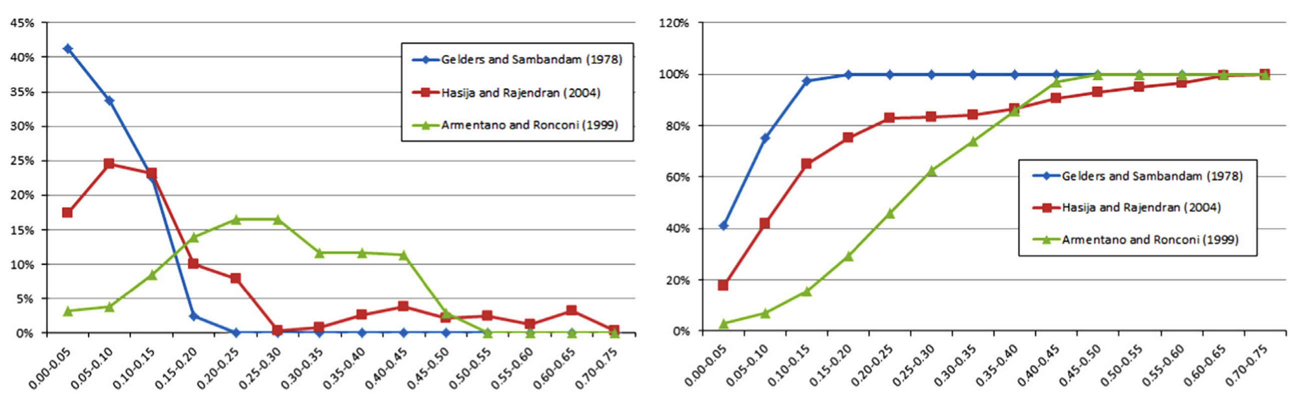

Fig. 1. Distribution of the percentage of instances depending on v for different generation of due dates (in the left, the percentage of instances of the testbeds in each interval of v is shown, while the right figure shows the cumulative percentage of instances).

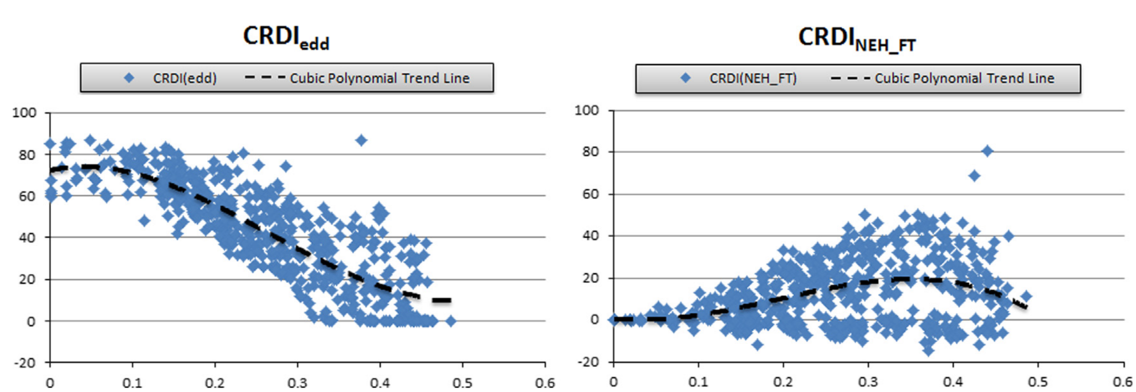

Fig. 2. $CRDI_{edd}$ and $CRDI_{NEH\_FT}$ for different values of v in each instance of the Benchmark of Vallada et al. [32].