Omega 44 (2014) 41–50

Contents lists available at ScienceDirect

Omega

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

An effective iterated greedy algorithm for the mixed no-idle permutation flowshop scheduling problem

Quan-Ke Pan [a,b], Rubén Ruiz [c,*]

[a] State Key Laboratory of Synthetical Automation for Process Industries (Northeastern University), Shenyang 110819, PR China
[b] College of Computer Science, Liaocheng University, Liaocheng 252059, PR China
[c] Grupo de Sistemas de Optimización Aplicada, Instituto Tecnológico de Informática, Ciudad Politécnica de la Innovación, Edifico 8G, Acc. B. Universitat Politècnica de València, Camino de Vera s/n, 46021 València, Spain

ARTICLE INFO

Article history:
Received 6 July 2013
Accepted 14 October 2013
Processed by B. Lev
Available online 25 October 2013

Keywords:
Flowshop
No-idle
Heuristics
Iterated greedy
Local search

ABSTRACT

In the no-idle flowshop, machines cannot be idle after finishing one job and before starting the next one. Therefore, start times of jobs must be delayed to guarantee this constraint. In practice machines show this behavior as it might be technically unfeasible or uneconomical to stop a machine in between jobs. This has important ramifications in the modern industry including fiber glass processing, foundries, production of integrated circuits and the steel making industry, among others. However, to assume that all machines in the shop have this no-idle constraint is not realistic. To the best of our knowledge, this is the first paper to study the mixed no-idle extension where only some machines have the no-idle constraint. We present a mixed integer programming model for this new problem and the equations to calculate the makespan. We also propose a set of formulas to accelerate the calculation of insertions that is used both in heuristics as well as in the local search procedures. An effective iterated greedy (IG) algorithm is proposed. We use an NEH-based heuristic to construct a high quality initial solution. A local search using the proposed accelerations is employed to emphasize intensification and exploration in the IG. A new destruction and construction procedure is also shown. To evaluate the proposed algorithm, we present several adaptations of other well-known and recent metaheuristics for the problem and conduct a comprehensive set of computational and statistical experiments with a total of 1750 instances. The results show that the proposed IG algorithm outperforms existing methods in the no-idle and in the mixed no-idle scenarios by a significant margin.

1. Introduction

It has been almost 60 years since the seminal work about the two machine flowshop problem with makespan minimization criterion by Johnson [16]. Actually, in the scheduling literature this paper has been regarded as the first in the field (with the possible exception of the paper by Salveson [48]). In a flowshop problem we deal with a set N of n jobs, modeling client orders of different products to be manufactured, that have to be produced on a set M of m machines. The layout of the machines in the production shop is in series, i.e., we have first machine 1, then machine 2 and so on until machine m. All jobs must visit the machines in the same processing sequence. This sequence can be, without loss of generality, $\{1,\dots,m\}$. Therefore, a job is composed of m tasks or operations. Each task $j, j=\{1,\dots,n\}$ requires a known, deterministic and non-negative amount of time at each machine i, $i=\{1,\dots,m\}$. This amount is referred to as processing time and denoted by p_{ij}. The objective is to find a processing sequence of all jobs at each machine so that a given criterion is optimized. There are as many possible sequences of jobs as permutations and this permutation can change from machine to machine which results in a search space of $(n!)^m$ non-delay schedules for the Flowshop Scheduling Problem (FSP). Given this huge search space, most of the time, the problem simplified by forbidding job passing, i.e., once a permutation of jobs is obtained for the first machine, it is maintained for all other machines, reducing the search space to $n!$ solutions. This somewhat simpler problem is referred to as the Permutation Flowshop Scheduling Problem or PFSP. Following the work of Johnson [16], the most studied optimization criterion is the minimization of the maximal job completion time or makespan (C_{max}) which corresponds to the time at which the last job in the sequence is finished at the last machine in the shop. The PFSP with makespan criterion is denoted as $F/prmu/C_{max}$, following the accepted three field notation of Graham et al. [13]. Reviews about flowshop scheduling with this criterion are given by Framinan

* Corresponding author. Tel.: +34 96 387 70 07; fax: +37 96 387 74 99.
E-mail addresses: panquanke@mail.neu.edu.cn (Q.-K. Pan), rruiz@eio.upv.es (R. Ruiz).

0305-0483/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.omega.2013.10.002

et al. [9], Ruiz and Maroto [40], Hejazi and Saghafian [15] and Gupta and Stafford [14]. The literature about flowshop scheduling is huge. Not only does each studied objective span a relatively large sub-field in itself with hundreds of references, like total tardiness minimization (see [53]), flowtime optimization [32] or multiobjective [20], but also problem extensions and variations abound. It is safe to say that the literature of flowshop scheduling and variants comprises thousands of papers.

One of the seldom studied extensions of the flowshop is the no-idle version. In the no-idle permutation flowshop (NPFSP), machines are not allowed to sit idle after they have started processing the first job in the sequence. The no-idle condition appears in production environments where setup times or operating costs of machines are so high that shutting down machines after the initial setup is not cost-effective. Idle times might also not be allowed on machines due to technological constraints. More specifically, in the no-idle scenario, a machine must process all jobs in the sequence without interruptions. Therefore, if needed, the start of some jobs is delayed so as to ensure the no-idle constraint. Examples of no-idle situations appear in the steppers used in the production of integrated circuits through photolithography. These fixtures are so expensive that idling is avoided at all costs. The production of ceramic frits is an example where idling is technologically impossible due to the usage of special fusing ovens (called kilns) that burn at extreme temperatures. These ovens need a continuous thermal mass and therefore, idling is not allowed. Some other examples are found in fiber glass processing [18], and foundries [46] amongst others. Ruiz et al. [44] and Goncharov and Sevastyanov [12] published recent reviews about the NPFSP or $F/prmu, no{-}idle/C_{\max}$.

The current situation is that the no-idle constraint has been so far considered all or nothing in the flowshop literature, i.e., either we have a regular idle flowshop where idle times are allowed on all machines or all machines have the no-idle constraint in the NPFSP. Real life production shops are mixed and most machines permit idle times whereas some do not accept idle times. Surprisingly, this realistic mixed no-idle flowshop problem or MNPFSP has not been studied in the literature before to the best of our knowledge. We denote this problem by $F/prmu, mixed\ no{-}idle/C_{\max}$. In the previous examples of integrated circuits and ceramic frit production, not all machines in the shop are no-idle. In the case of ceramic frits, only the central fusing kiln has the no-idle constraint. Other examples arise in the steelmaking industry. When producing steel, the charges of molten iron enter converter stages to reduce impurities (carbon, sulfur, silicon) through oxygen burning. These charges undergo several other refining stages where impurities are further reduced, alloys are added and other operations are carried out. Only after this phase, the molten steel is poured into a tundish for casting. The flow of molten steel goes to the crystallizer where it solidifies into slabs. Technological constraints force the continuous flow of charges with the same crystallizer and caster. This is where the no-idle constraint appears. All other stages do not have this no-idle constraint. There are many other examples in real-life factories. As a matter of fact, the authors are not aware of any real example in which all the machines in a flowshop have the no-idle constraint. Therefore, the MNPFSP is a more realistic problem which has not been studied before and is thus the motivation for this research. The PFSP is known to be $\mathcal{NP}-$Complete in the strong sense for more than two machines and makespan criterion [11]. Similarly, the NPFSP was shown to belong to the same complexity class for three or more machines by Baptiste and Hguny [3]. As a result the new MNPFSP studied in this paper is also $\mathcal{NP}-$Hard in the strong sense.

The rest of the paper is divided into five more sections. In the next section we review the literature mainly in the no-idle flowshop. Section 3 introduces the MNPFSP in more detail. We present a mixed integer programming model, the formulae to calculate the makespan and a speed-up method for the efficient calculation of the insertion neighborhood. Section 4 deals with the proposed Iterated Greedy method. In Section 5 we present a comprehensive computational and statistical campaign to test the proposed methodology. Finally, Section 6 concludes the paper and provides some avenues for further research.

2. Literature review

As stated, the MNPFSP has not been studied before. As a result, we focus our summarized review in the no-idle flowshop where all machines have the no-idle constraint. The NPFSP was first studied by Adiri and Pohoryles [1] where polynomial time algorithms were proposed for special cases of the NPFSP mainly with two machines and total completion time criterion. Some amendments to this paper were carried out by Čepek et al. [54]. The $C_{\max}$ objective in the NPFSP was studied for the first time by Vachajitpan [51]. The author presented mathematical models and branch and bound methods for small instances. Baptiste and Hguny [3] also presented a branch and bound method for the m-machine NPFSP and makespan criterion whereas the three machine problem was studied by Narain and Bagga [24] also with mathematical models and exact approaches. To date, no effective exact approach has been proposed for the NPFSP and rarely do any published results solve problems with more than a handful of jobs. As a result of this, the focus has been on heuristics for the problem. Some of the early heuristic methods were presented by Woollam [55] that took some existing heuristics and recalcuated their produced solutions eliminating idle times and doing some simple adjacent pairwise exchange moves on the results. The adaptation of the NEH heuristic of Nawaz et al. [27] produced the best results. Saadani et al. [45] presented a method based on heuristics for the traveling salesman problem denoted as SGM. This research was later published in paper form in Saadani et al. [47]. The three machine case was studied by Saadani et al. [46] to be improved on later by Kalczynski and Kamburowski [19]. Heuristics for special cases with dominating machines are studied by Narain and Bagga [25,26].

The general m-machine NPFSP with makespan criterion has been approached with successful heuristics by several authors. For example, [18] presented a method based on Johnson's heuristic, denoted as KK that was shown to outperform an adaptation of the NEH heuristic to the no-idle setting and the method of Saadani et al. [47]. A local search insertion method proposed by Baraz and Mosheiov [4] is also shown to outperform that of Saadani et al. [47] and is denoted by GH_BM.

Ruiz et al. [44] presented a comprehensive comparison of heuristic methods, along with adaptations of the NEH method and the best heuristics proposed for the PFSP by Rad et al. [37]. The authors also presented an improved GH_BM method. All methods were tested with and without the accelerations of the insertion neighborhood presented by Pan and Wang [33,34]. The results of the comprehensive computational and statistical campaign with a set of 250 instances were clear: the adapted method FRB3 of Rad et al. [37] and the improved GH_BM2 version, both with accelerations produced the best results.

As regards metaheuristics, the first papers are by Pan and Wang [33,34]. In the first, the authors present a discrete particle swarm optimization method, referred to as HDPSO. In the second a discrete differential evolution method is presented (DDE). Both papers are heavily based on insertion local search and an important result is given: an acceleration of the calculation of the exploration of this neighborhood. Similar to what Taillard [49] did, the authors explain a set of calculations to reduce the complexity of the calculation of a pass in the insertion neighborhood from $\mathcal{O}(n^3m)$ to $\mathcal{O}(n^2m)$ in the NPFSP. The authors hybridized their methods with the Iterated Greedy algorithm of Ruiz and Stützle [42] and demonstrated in