Computers & Industrial Engineering 110 (2017) 413–423

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Iterated reference greedy algorithm for solving distributed no-idle permutation flowshop scheduling problems

Kuo-Ching Ying [a,*], Shih-Wei Lin [b,c], Chen-Yang Cheng [a], Cheng-Ding He [a]

[a] Department of Industrial Engineering and Management, National Taipei University of Technology, Taipei, Taiwan
[b] Department of Information Management, Chang Gung University, Taoyuan, Taiwan
[c] Department of Neurology, Linkou Chang Gung Memorial Hospital, Taoyuan, Taiwan

ARTICLE INFO

Article history:
Received 25 September 2016
Received in revised form 15 March 2017
Accepted 13 June 2017
Available online 15 June 2017

Keywords:
Scheduling
Distributed permutation flowshop
No-idle
Iterated reference greedy algorithm

ABSTRACT

This paper investigates the Distributed No-idle Permutation Flowshop Scheduling Problem (DNIPFSP) with the objective of minimizing the makespan, which has not been discussed in any previous study. This study presents an Iterated Reference Greedy (IRG) algorithm for effectively solving this problem. The performance of the proposed IRG algorithm is compared with a state-of-the-art iterated greedy (IG) algorithm, as well as the Mixed Integer Linear Programming (MILP) model on two well-known benchmark problem sets. Computational results show that the proposed IRG algorithm outperforms the IG algorithm. Given the *NP*-Complete nature of the DNIPFSP problem, this paper is the first study to contribute a feasible approach for solving it effectively.

1. Introduction

The permutation flowshop scheduling problem (PFSP) is one of the most important scheduling problems. Johnson (1954) first investigated the 2-machine PFSP and provided the so-called Johnson's rule to minimize the makespan (i.e. the maximum completion time), while the 3-machine version is *NP*-complete (Garey, Johnson, & Sethi, 1976). Over the past six decades, a variety of studies have been developed on this problem with respect to theories, applications, and solution approaches (Lee & Chung, 2013). A few critical reviews (Framinan, Gupta, & Leisten, 2004; Ruiz & Maroto, 2005; Yenisey & Yagmahan, 2014) have also been given on this topic. Because of the *NP*-Complete property of the problem, meta-heuristics have become useful and efficient methods when solving large-scale PFSPs. Efficient meta-heuristics proposed for the PFSPs include Tabu search (TS; Grabowski & Wodecki, 2004), genetic algorithm (GA; Ruiz, Maroto, & Alcaraz, 2006), simulated annealing (SA; Ishibuchi, Misaki, & Tanaka, 1995), variable neighborhood search (VNS; Lei, 2015; Naderi & Ruiz, 2010), and iterated greedy (IG; Ruiz & Stutzle, 2007). However, most studies on this problem concentrate on traditional single flowshop scheduling problems. With the increasing popularity of globalized production, additional studies are required in order to develop different methods for dealing with the newly emerged distributed permutation flowshop scheduling problem (DPFSP).

The DPFSP is a generalization of the classical PFSP from the traditional single factory to the current decentralized and globalized manufacturing environment, where multiple factories might be available for a firm. Reflecting the gap between practical applications and theory progress, the DPFSP that concerns the assignment of jobs to various factories and their subsequent scheduling has received increasing attention during the last decade. DPFSPs in the multi-factory environment were first studied by Wang (1997), considering that jobs could be distributed to identical factories for the same production process, thus achieving lower production cost, and better risk management. Naderi and Ruiz (2010) proposed six different alternative Mixed Integer Linear Programming (MILP) models, two simple factory assignment rules together with 14 heuristics based on dispatching rules, effective constructive heuristics and two variable neighborhood descent (VND) methods to minimize the makespan for the DPFSP. The performance of the proposed algorithms was properly evaluated via benchmark instances extended from the well-known benchmark problem set of Taillard (1993). Later, Gao and Chen (2011) proposed a GA-based algorithm, denoted by GA_LS, which utilizes an efficient local search method to explore neighboring solutions. Intensive experiments were conducted on the same testbed as that used by Naderi and Ruiz (2010), and the computational results indicate that GA_LS can obtain better solutions than all the existing algorithms as well as the two VND algorithms (Naderi & Ruiz, 2010) for the DPFSP. Recently, Gao, Chen, and Deng (2013) presented a TS algorithm exploiting a novel Tabu strategy and an enhanced local search method, able to achieve better performance

* Corresponding author.
E-mail address: kcying@ntut.edu.tw (K.-C. Ying).

http://dx.doi.org/10.1016/j.cie.2017.06.025
0360-8352/

than the GA_LS algorithm (Gao & Chen, 2011). In the same year, Wang, Wang, Liu, and Xu (2013) developed an estimation of distribution algorithm (EDA) for solving the DPFSP. Hatami, Ruiz, and Andrés-Romano (2013) further investigated a generalization of DPFSPs, the distributed assembly permutation flowshop scheduling problem, and proposed the VND algorithm with the demonstrated outperformance. Although their proposal was not directly compared with the state-of-the-art algorithms, the comparative results with some existing heuristics demonstrate the effectiveness of the proposed EDA in solving this problem. Next, Lin, Ying, and Huang (2013) proposed a modified IG (MIG) algorithm for the DPFSP, and showed its superiority over the above methods. The best-known solutions for almost half of instances of the extended benchmark problem set of Taillard are updated by using the MIG algorithm. Naderi and Ruiz (2014) presented a scatter search (SS) algorithm, which employs some advanced techniques like a reference set made up of complete and partial solutions, along with other features like restarts and local search for the DPFSP. Their experimental results show that the SS algorithm produces better results than 10 existing algorithms, including the (Gao & Chen, 2011) and TS (Gao et al., 2013) algorithms, by a significant margin. Basing their work on the IG algorithm, Fernandez-Viagas and Framinan (2015) presented a bounded-search IG (BSIG) algorithm, and showed that it outperforms TS (Gao et al., 2013), EDA (Wang et al., 2013), and MIG (Lin et al., 2013) algorithms. In addition, better upper bounds for more than one quarter of problems in the testbed of Naderi and Ruiz (2010) are obtained using the BSIG algorithm. To the best of the authors' knowledge, BSIG is by far the most effective meta-heuristic algorithm for solving the DPFSP to optimize makespan.

The no-idle PFSP (NIPFSP) is a key issue in various manufacturing systems, especially in fiberglass processing and foundry operations, where the setup costs of machines are so high that shutting down and reactivating of machines is not cost-effective (Pan & Ruiz, 2014; Saadani, Guinet, & Moalla, 2003; Tasgetiren, Pan, Suganthan, & Buyukdagli, 2013). Therefore, most managers would rather extend the completion time of jobs by postponing the start of processing the first job on a given machine. This scheduling problem was first considered and proven to be *NP*-Complete (Adiri & Pohoryles, 1982). After this, some algorithms (e.g. Deng & Gu, 2012; Kalczynski & Kamburowski, 2005; Pan & Wang, 2008) were developed for NIPFSPs. Most recently, Pan and Ruiz (2014) extended the successful IG algorithm of Ruiz and Stutzle (2007) to minimize the makespan of the mixed NIPFSP. The designed algorithm is called IG_{PR}, which considers the technique of $FRB4_1$ algorithm (Rad, Ruiz, & Boroojerdian, 2009) for the insertion neighborhoods in the phases of initialization and reconstruction, and thus provides more diversity to search better candidate solutions. They showed the IG_{PR} algorithm is superior to other existing heuristics by conducting numerical comparisons of 1750 instances. In terms of meta-heuristics, IG-based algorithms have been successfully applied to various scheduling problems (Hatami, Ruiz, & Andrés-Romano, 2015; Pan & Ruiz, 2014; Ruiz & Stutzle, 2007; Ying, 2008, 2012a, 2012b; Ying, Lin, & Wan, 2014), and have consistently showed their supremacy over NIPFSP with their tractability and superior performance in attaining near-optimal.

Although NIPFSPs have been broadly studied over the past decades, literature searches indicate that no study has been conducted to date on the Distributed No-idle Permutation Flowshop Scheduling Problem (DNIPFSP). Therefore, this study provides the first attempt to investigate this significant problem. The aim is simultaneously to assign jobs to various factories and to determine their production sequences in each factory to minimize the makespan. Using the three field classification scheme of Graham, Lawler, Lenstra, and Rinnooy Kan (1979), the addressed problem can be designated by a triplet $DF_m|prmu, no-idle|C_{\max}$. Since this problem with only one factory reduces to a general NIPFSP, which is *NP*-Complete (Adiri & Pohoryles, 1982), the $DF_m|prmu, no-idle|C_{\max}$ problem can be confidently concluded also to be *NP*-Complete. To this end, this study proposes a Mixed Integer Linear Programming (MILP) model, and an Iterated Reference Greedy (IRG) algorithm to effectively solve it. The designed IRG algorithm has significant improvements in the so-called perturbation mechanism, acceleration of makespan calculation for neighborhoods evaluation, and acceptance criterion of IG algorithm. The performance of the proposed IRG algorithm is to be demonstrated by numerical comparisons with the MILP and the state-of-the-art IG_{PR} algorithm, based on existing simulation instances (Naderi & Ruiz, 2010) augmented from the Taillard benchmark problem set (Taillard, 1993). The remainder of this paper is organized as follows. The following section formulates the investigated problem. Section 3 offers a detailed description of the proposed IRG algorithm. tion 4 presents the performance evaluations of the proposed algorithm. Section 5 gives the conclusions of this study.

2. Problem formulation

This section defines the $DF_m|prmu, no-idle|C_{\max}$ problem considered herein. It will offer some critical assumptions, and formulate the corresponding MILP model.

- The problem considers a group of *f* identical factories, which are located in different areas, and share the processing responsibility of a set of *n* jobs.
- More than one factories, i.e. $f > 1$, are taken into consideration to avoid the addressed problem become a trivial one.
- All factories have the same workflows, each with a flowshop production system that consists of the same set of *m* machines in the fixed permutation.
- A set of *n* jobs is to be assigned to any one of *f* identical factories, and to be sequentially processed through the *m* machines of the assigned factory in an identical sequence without preemption.
- There must be no idle intervals between the processing of any consecutive jobs on each machine. That is, each machine must process jobs without any interruption from the start of processing the first job to the completion of processing the last job.
- All jobs are assumed to be ready at time zero, when needed, the start time of the first job on a given machine should be delayed in order to meet the no-idle requirement.
- Each machine can only process one job at a time, and the processing time of a job at one specific machine is the same from factory to factory.
- Any interruption during processing a job and breakdown of a machine are ignored.
- The objective of scheduling is simultaneously to assign jobs to various factories and to determine their production sequences in each factory to minimize the makespan ($C_{\max}$).

Let p_{ij}, $i = 1, \ldots, n, j = 1, \ldots, m$ be the processing time of job *i* on machine *j*, and $C_{\ell jk}$, $\ell = 1, \ldots, n_k$, $j = 1, \ldots, m$, $k = 1, \ldots, f$ be the completion time of the job at the ℓth priority on machine *j* of factory *k*, in which n_k is the number of jobs processed in factory *k*, such that $\sum_{k=1}^{f} n_k = n$. Note that the completion time of each job, i.e. $C_{\ell jk}$, is also treated as the decision variable to be optimally determined. Let $x_{i\ell k}$, $i = 1, \ldots, n$, $\ell = 1, \ldots, n_k$, $k = 1, \ldots, f$ be the binary decision variable to determine the assignment of job *i*. If job *i* is assigned to the ℓth processing priority in factory *k*, then $x_{i\ell k} = 1$; otherwise, $x_{i\ell k} = 0$. Based on the above assumptions and defined notations, the $DF_m|prmu, no-idle|C_{\max}$ problem can be formulated as the following MIP mathematical model: