Computers & Industrial Engineering 113 (2017) 160–184

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Efficient heuristic for solving non-permutation flow-shop scheduling problems with maximal and minimal time lags

Song Ye, Ning Zhao *, Kaidian Li, Chuanjin Lei

School of Mechanical Engineering, University of Science and Technology Beijing, Xueyuan Road 30, Haidian District, Beijing 100083, PR China

A R T I C L E   I N F O

Article history:
Received 3 November 2016
Received in revised form 17 July 2017
Accepted 18 August 2017
Available online 9 September 2017

Keywords:
Non-permutation
Flow shop
Time lags
Makespan
Iterated greedy algorithm

A B S T R A C T

Flow-shop scheduling problem is an attractive subject in the scheduling field, which has attracted the attention of many researchers in the past five decades. However, few studies focused on non-permutation flow-shop problems with time lag consideration. In the present work, the non-permutation flow-shop scheduling problem with time lags has been studied to minimize the makespan as a performance measure. First, we obtain a near-optimal permutation solution using the permutation flow-shop problem (PFSP) heuristic. Then, an effective iterated greedy heuristic, which can identify high-quality non-permutation solutions, is presented. Using the neighbourhood non-PFSP searching heuristic, we searched for non-PFSP schedules using the proposed heuristic algorithms in the second stage. Finally, the computational results were used to evaluate the performance and effectiveness of the proposed heuristic. The proposed heuristics were able to find near optimal non-PFSP solutions with very short computational time. Thus, the proposed algorithms is efficient and can be used in industrial applications. Moreover, the proposed heuristic algorithms are very simple to implement, which is attractive for industrial applications.

1. Introduction

Flow-shop problems (FSPs) have been studied for more than half a century. The classical FSP investigates n jobs for processing in m machines. It is called permutation flow-shop scheduling (PFSP) if all machines process their jobs in the same sequence; otherwise, it is a non-permutation scheduling (non-PFSP) Lageweg, Lenstra, & Rinnooy Kan, 1978; Potts, Shmoys, & Williamson, 1991. Obviously, non-PFSP has more sequences and is more complex than PFSP. According to some literatures (Cui, Lu, Zhou, Li, & Han, 2016), the PFSP schedules are no longer dominant compared with non-PFSP in more than three machines (Cui et al., 2016). Consequently, non-PFSP schedules arises and possible to be the optimal solution when machine number is greater than three.

Time lag means the waiting-time constraints between two consecutive operations in the same job. Maximal time lags used to demand the waiting time between operations must not be overly long to avoid deterioration of products (Hodson, Muhlemann, & Price, 1985). Minimal time lags may be used when waiting time between operations is required for processing, such as cooling (Fondrevelle, Oulamara, & Portmann, 2006), material handling (Soukhal, Oulamara, & Martineau, 2005), and chemical reactions (Chu & Proth, 1996). Minimal and maximal time lags exist in many industrial applications such as fabrication of printed circuits (Kim, Lim, & Park, 1996), hoist-scheduling problems (Manier & Bloch, 2003), perishable-product production (Johnson, 1954), and in biotechnology and chemistry (Nawaz, Enscore, & Ham, 1983). Thus, FSPs that consider time lags are realistic industrial problems.

Since the original PFSP paper was published by Johnson (1954), various researchers have studied the problem. Consequently, many heuristic and meta-heuristic methods have been developed to date. Among these methods, NEH heuristic (Nawaz et al., 1983) is commonly regarded as the most effective constructive heuristic. Fondrevelle et al. (2006) investigated two-machine PFSPs with minimal and maximal time lags and showed that PFSP is not dominant when the objective is to minimize the makespan, especially when the minimal and maximal time lags are greater than the processing time. Subsequently, Ruiz and Stutzle proposed a meta-heuristic method in 2007 based on the NEH heuristic, which is called the iterated greedy (IG) algorithm Ruiz & Stutzle, 2007. Nikbakhsh, Mohammad, and Mohammad (2012) proposed an immune algorithm for hybrid FSPs that consider time lags and sequence-dependent setup times. Dhouib, Teghem, and Loukil (2013) proposed a mixed-integer mathematical programming

* Corresponding author.
E-mail address: zhning@sina.com (N. Zhao).

and simulated annealing algorithm to minimize the number of tardy jobs and the makespan. Sheikh (2013) proposed a genetic algorithm (GA) with the twin objectives of maximizing the total profit and minimizing deviation from the due date. Most of these studies focused on the PFSP and neglected the non-PFSP. However, as stated by Fondrevelle, PFSP is not always dominant. In other words, in some industrial applications, non-permutation solutions may bring more benefit than permutation solutions. However, these benefits are consequent neglected when only PFSP algorithms are applied.

Only very few papers deal with the non-PFSPs. Lin and Ying (2009) proposed a hybrid approach that draws on the advantages of simulated annealing and tabu search for the non-PFSP without time lags. Mehravaran and Logendran (2012) considered non-PFSP schedules for a flowshop problem with sequence-dependent setup times. Ying (2008) proposed an effective IG heuristic for the non-PFSPs. Vahedi-Nouri, Fattahi, and Ramezanian (2013) proposed an effective improvement heuristic for the non-PFSPs with learning effects and availability constraints. To the best of the author's knowledge, almost all of these studies did not consider time lags.

Several researchers have focused on the job-shop problem with time lags which can be also solve the non-PFSPs. Caumond, Lacomme, and Tchernev (2008) proposed a memetic algorithm based on a disjunctive graph that is suitable for various industrial situations. His algorithm covered both job-shop problems and flowshop problems. Artigues, Huguet, and Lopez (2010) proposed an insert heuristic and generalized resource constraint propagation approach for the job-shop problem. The results of the experiments showed that their approach is suitable for instances with tight time lags. The algorithms cited above provided excellent results. However, these algorithms are not specially designed for non-PFSP problems, and therefore these algorithms are not sufficiently efficient for industrial application, especially in complex scheduling problem. Consequently, more efficient approaches for non-PFSP industrial applications are needed.

Because both PFSP and non-PFSP exist in real industrial flow shop scenarios with time lags, therefore scheduling approaches suit for both PFSP and non-PFSP have strong practical values. For this reason, Zhao, Ye, Li, and Chen (2017) propose a universal approach to resolve both of them with time lag consideration. The approach includes two main steps: First, improving traditional IG algorithm developed by Ruiz and Stutzle (2007) and search optimal PFSP solutions. Second, search non-PFSP schedules neighbours to the optimal PFSP solution and keep the best one. This approach results PFSP solution or non-PFSP solution, it depends on whether better non-PFSP schedules can be reached by neighbourhood searching. However, it is destined to be useless when PFSP is dominated.

In general, although some studies have focused on PFSP and non-PFSP problems, but studies about non-PFSP with time lags are not sufficient. Due to the complexity of non-PFSP and in practical viewpoint, we further the study with the idea that neighbourhood searching with initial PFSP solution (Zhao et al., 2017). In order to avoid useless non-PFSP neighbourhood searching, we focused on studying the dominance conditions of PFSP and non-PFSP in the present work. Furthermore, we develop hybrid neighbourhood searching heuristic to improve the efficiency of non-PFSP searching. Consequently, we integrated the heuristics with the PFSP searching approaches and validate them with realistic scheduling problems.

The remainder of this paper is organized as follows. Section 2 analyses the problem and defines the condition for PFSP and non-PFSP respectively. Section 3 outlines our proposed algorithms for non-PFSPs. Section 4 shows the evaluation performance by computational results. Finally, our conclusions are presented in Section 5.

2. Problem definition and dominance condition

In this section, we analyse the effects of time lags of the non-PFSP. To illustrate the studied problem, we provide the following definitions.

2.1. Notation definition

We assume a set of n independent jobs ($J_1, J_2, \ldots, J_n$) scheduled on a set of m machines ($M_1, M_2, \ldots, M_m$). Each job comprises m operations. All n jobs are to be processed in the same machine sequence from 1 to m. Furthermore, we assume that the processing time of the jobs in each machine is known in advance and that all jobs are available at the start time. All machines are constantly available to process all scheduled jobs when required. Every job can be processed by at most one of the machines at any given time without preemption. Every machine can process no more than one job at a time, and the processing of each job cannot be interrupted. Time lag is defined as the gap between the stopping time of the job in the upstream machine and its starting time in the downstream machine (stop–start lags). Our objective is to minimize the completion time of the final job. The following notations are defined:

$p_{j,i}$	Processing time of job j in machine i ($p_{j,i}$ is fixed and non-negative)
$t_{j,i}$	Start time of job j in machine i
$c_{j,i}$	Completion time of job j in machine i
$\theta_{j,i}^{min}$	Minimal time lag of job j from machine i to machine $i+1$
$\theta_{j,i}^{max}$	Maximal time lag of job j from machine i to machine $i+1$
$\theta_{j,i}$	Actual time lag of job j from machine i to machine $i+1$

Our objective is to determine $t_{j,i}$ that satisfies

$$\forall (j \leqslant n, i \leqslant m), \min(\max c_{j,i}) \tag{1}$$

s.t.

$$\forall (j \leqslant n, i \leqslant m), c_{j,i} = t_{j,i} + p_{j,i} \tag{2}$$

$$\forall (j \leqslant n, i < m), c_{j,i} + \theta_{j,i}^{max} \geqslant t_{j,i+1} \geqslant c_{j,i} + \theta_{j,i}^{min} \tag{3}$$

$$\forall (j \leqslant n, i \leqslant m), t_{j+1,i} \geqslant c_{j,i} \tag{4}$$

$$\forall (j \leqslant n, i \leqslant m), \theta_{j,i}^{min} \leqslant \theta_{j,i} \leqslant \theta_{j,i}^{max} \tag{5}$$

Eq. (1) indicates that the objective is to minimize the makespan, Eq. (2) denotes the processing time constraint, Eq. (3) denotes the time lag constraints, and Eq. (4) denotes the machine capacity constraints. Eq. (5) denotes the actual time lag constraints.

2.2. Feasibility and dominance of non-PFSP in two machines problem

Fondrevelle (Fondrevelle et al., 2006) has proved the NP hardness of two machines problem and analysed the dominance of permutation and non-permutation schedules. In this paper, we investigate the feasibility and advantage for transforming permutation schedules to neighbourhood non-permutation schedules. We also start the study from two-machine problem. We consider