Computers & Industrial Engineering 101 (2016) 167–176

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Self-adaptive ruin-and-recreate algorithm for minimizing total flow time in no-wait flowshops

Kuo-Ching Ying [a], Shih-Wei Lin [b,c,*], Wen-Jie Wu [b]

[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] Stroke Center and Department of Neurology, Linkou Chang Gung Memorial Hospital, Taoyuan, Taiwan

ARTICLE INFO

Article history:
Received 16 September 2015
Received in revised form 25 April 2016
Accepted 21 August 2016
Available online 22 August 2016

Keywords:
Scheduling
No-wait flowshop
Total flow time
Self-adaptive ruin-and-recreate algorithm

ABSTRACT

This paper studies the no-wait flowshop scheduling problem with total flow time criterion, which is NP-complete in the strong sense. A self-adaptive ruin-and-recreate (SR&R) algorithm is proposed to solve this complex problem. The performance of the proposed SR&R algorithm is compared with that of the best available heuristics and the SR&R algorithm without the self-adaptive mechanism by application to a set of classic benchmark instances that were presented by Taillard. Computational results show that the proposed SR&R algorithm improves upon the best known solutions in more than half benchmark instances, and provides the best known solutions for the remaining unimproved instances in a reasonable computational time. The contribution of this work is to provide an easy-to-use approach to solve effectively and efficiently this practical but complex scheduling problem.

1. Introduction

The permutation flowshop scheduling problem (PFSP) has been a major concern for scheduling researchers who have proposed various algorithms to effectively and efficiently solve it (Chang & Chen, 2014; Chang, Chen, Tiwari, & Iquebal, 2013; Chen, Chang, Cheng, & Zhang, 2012; Chen, Chang, & Lin, 2014; Chen, Chen, Chang, & Chen, 2012; Hsu, Chang, & Chen, 2015). This paper deals with the no-wait flowshop scheduling problem (NWFSP) with the objective of minimizing the total flow time. Consistent with the standard classification scheme that was proposed by Graham, Lawler, Lenstra, and Rinnooy Kan (1979), this problem can be synthetically denoted as an $F_m|nwt|\sum C_j$ problem, where F_m represents a flowshop problem with m machines; nwt means that jobs are not allowed to be held up between two successively used machines, usually owing to technological restrictions of the manufacturing process, and $\sum C_j$ denotes that the objective is to minimize the total flow time, which criterion is one of the most-studied optimization criteria for the NWFSP. Minimizing the total flow time can lead to a stable or uniform utilization of resources, a rapid turn-around of jobs, and minimization of the in-process inventory (Rajendran, 1994). If all jobs are assumed to be ready for processing at the beginning of the scheduling horizon, then minimizing the total flow time is equivalent to minimizing the sum of completion times and, consequently, the mean processing time. The $F_m|nwt|\sum C_j$ problem is a typical scheduling problem that has important applications in myriad industries, including the steel, pharmaceutical, chemical, plastic, electronic and food-processing industries (Sapkal & Laha, 2013). The modern agile manufacturing system, in which robots and industrial machines implement a highly coordinated process, can be modeled as an NWFSP (Bertolissi, 2000). For a review of the various applications of the $F_m|nwt|\sum C_j$ problem, see Pan, Tasgetiren, and Liang (2008). Despite its widely applications, the NWFSP is one of the most challenging problems in the field of scheduling, and is NP-complete (Rock, 1984) in the strong sense, even in the two-machine case.

The $F_m|nwt|\sum C_j$ problem was first posed by Van Deman and Baker (1974). Only a few algorithms have been proposed for solving it. With respect to exact methods, Van Deman and Baker (1974) presented a set of procedures for generating the lower bounds and then found the optimal solution by applying a branch and bound (B&B) algorithm. Their experimental results revealed that the proposed B&B algorithm can solve the $F_m|nwt|\sum C_j$ problem as rapidly as it can solve the traditional flowshop problem with makespan as the performance criterion. Given the NP-nature of the $F_m|nwt|\sum C_j$ problem, only small instances can be solved optimally using these exact algorithms with an acceptable computational time and

* Corresponding author at: Department of Information Management, Chang Gung University, Taoyuan, Taiwan.
E-mail address: swlin@mail.cgu.edu.tw (S.-W. Lin).

http://dx.doi.org/10.1016/j.cie.2016.08.014
0360-8352/

memory requirements. Therefore, heuristic algorithms that yield (near-) optimal solutions in a reasonable computational time with reasonable memory requirements have become viable for tackling the $F_m|nwt|\sum C_j$ problem, and especially for scheduling a large number of jobs. However, relevant literature on the $F_m|nwt|\sum C_j$ problem is limited.

The limited heuristic algorithms for solving the $F_m|nwt|\sum C_j$ problem can be broadly classified into two categories, which are constructive heuristics and improvement heuristics. With respect to constructive heuristics, Rajendran and Chaudhuri (1990) were the first to present a simple construction heuristic for solving the $F_m|nwt|\sum C_j$ problem. Ten years later, Bertolissi (2000) presented a constructive heuristic that was based on job insertion and compared it with one of the constructive heuristics of Rajendran and Chaudhuri (1990), and with another that was proposed by Bonney and Gundry (1976) for solving the NWFSP with the objective of minimizing makespan. His experimental results indicated that his heuristic outperformed the other two. Fink and Voß (2003) developed two simple construction heuristics, namely nearest neighborhood (NN) and cheapest insertion (Chins), as well as a local search procedure (Pilot-1-Chins) to improve the solution to the problem. Their computational results showed that Chins clearly outperforms NN but they did not compare their methods with the aforementioned constructive heuristics. Aldowaisan and Allahverdi (2004) presented six constructive heuristics for solving the $F_m|nwt|\sum C_j$ problem. Experimental results revealed that among them, the PH1p heuristic outperformed the others and the two constructive heuristics of Rajendran and Chaudhuri (1990). Later, Framinan, Nagano, and Moccellin (2010) presented a new constructive heuristic that was based on an analogy with the two-machine problem to select a candidate job to be appended to the incumbent partial solution. Their computational results demonstrated that the proposed constructive heuristic outperformed existing constructive heuristics and was competitive with a highly time-consuming local search procedure. Gao, Pan, Suganthan, and Li (2013) proposed two constructive heuristics - the improved standard deviation heuristic (ISDH) and the improved Bertolissi heuristic (IBH) - for solving the $F_m|nwt|\sum C_j$ problem, and then utilized the insertion-based local search method and an iteration operator to improve their solutions. Recently, Laha and Sapkal (2014) developed a constructive heuristic, called the LS heuristic, which is based on the assumption that the priority of a job in the initial sequence of jobs is given by the sum of its processing times on the bottleneck machines. Their computational results revealed that the LS heuristic significantly outperformed the best known heuristics while retaining a time complexity of $O(n^2)$. Although these constructive heuristics are rather easy to understand and can be straightforwardly implemented, their lack of robustness inhibits their application in practice. This major problem may be partially solved by using improvement heuristics, which can yield more robust solutions with polynomial time complexity.

In 1996, Chen, Neppalli, and Aljaber (1996) became the first to apply improvement heuristics, having a genetic algorithm (GA) for solving the $F_m|nwt|\sum C_j$ problem. Their computational results revealed that GA was an effective technique for solving the scheduling problem of interest. Fink and Voß (2003) proposed a steepest descent (SD) algorithm, an iterated steepest descent (ISD) algorithm, a simulated annealing (SA) algorithm and a reactive Tabu search (RTS) algorithm to solve the same problem. Their computational results demonstrated that high-quality solutions could be obtained efficiently by applying the SA and RTS algorithms without knowledge of their inner workings. Then, Shyu, Lin, and Yin (2004) applied an ant colony optimization (ACO) algorithm for application to the two-machine case. Computational results demonstrated that the ACO algorithm performed impressively in solving the scheduling problem of interest. Kumar, Prakash, Shankar, and Tiwari (2006) presented a psycho-clonal algorithm, which outperformed both the GA of Chen et al. (1996) and the six constructive heuristics of Aldowaisan and Allahverdi (2004). Tasgetiren, Pan, Suganthan, and Liang (2007) presented a discrete differential evolution (DDE) algorithm that was hybridized with the variable neighborhood descent (VND) algorithm to solve the $F_m|nwt|\sum C_j$ problem. Their computational results revealed that the DDE algorithm outperforms the SA and TS algorithms of Fink and Voß (2003). Subsequently, Pan et al. (2008) proposed an improvement heuristic, denoted as DPSO_{vnd}, that hybridized the discrete particle swarm optimization algorithm with the VND algorithm and further improved upon the best known solutions, which were obtained by Fink and Voß (2003). Their experimental results indicated that the DPSO_{vnd} algorithm yielded results that were either competitive with or better than those reported in the literature. Gao, Pan, and Li (2011) developed a discrete harmony search algorithm (DHS) for solving the $F_m|nwt|\sum C_j$ problem. Their computational results revealed their algorithm outperformed the DPSO_{vnd} algorithm of Pan et al. (2008). Gao, Pan, Li, and Wang (2012) proposed a hybrid harmony search (HHS) algorithm for solving the same problem. Computational simulations based on the well-known benchmarks showed that HHS was superior to a hybrid differential evolution algorithm and a hybrid particle swarm optimization algorithm that had been proposed for solving other FSPs. Recently, Zhu, Li, and Wang (2009) proposed an objective increment-based iterative greedy algorithm, denoted FIG, and the computational results demonstrated that it outperformed PH1p but underperformed the DPSO_{vnd} algorithm. Zhu and Li (2015) developed an iterative search (IS) algorithm for solving the $F_m|nwt|\sum C_j$ problem; they compared it with the best existing algorithms using classical benchmark instances and revealed that it outperformed them.

The ruin-and-recreate (R&R) algorithm is the novel improvement heuristic presented by Schrimpf, Schneider, Stamm-Wilbrand, and Dueck (2000) for solving traveling salesman problems, vehicle routing problems and network optimization problems. The main advantages of the R&R are its simplicity, effectiveness and suitability for complex optimization problems, such as quadratic assignment problems (Misevicius, 2003), permutation flowshop scheduling problems (Burke et al., 2009), bin-packing problems (Burke et al., 2009) and paratransit scheduling problems (Häll & Peterson, 2013). As astounding results have been obtained for some classical optimization problems, in this work we present a self-adaptive ruin-and-recreate (SR&R) algorithm for solving the $F_m|nwt|\sum C_j$ problem. The main idea of the proposed SR&R algorithm is the use of a new constructive heuristic based on the famous NEH heuristic (Nawaz, Enscore, & Ham, 1983) to generate an initial solution, which is then iteratively improved by a self-adaptive ruin-and-recreate procedure with a speed-up method. The presented self-adaptive mechanism dynamically adjusts the size and range of the neighborhood during the ruin phase, thus enabling the incumbent solution to escape from the local minima by enlarging the perturbation, while the proposed speed-up method accelerates the evaluation of the neighborhoods of the insertion operation during the recreate phase. The Boltzmann function, which is commonly used in the annealing process in the SA algorithm (Lin, Lu, & Ying, 2011; Lu, Lin, & Ying, 2012; Ying, Lin, & Lu, 2011), is also employed to decide whether or not to update the incumbent solution with the new solution in order to escape from the local minima. To the best of our knowledge, this work provides the first reported application of the ruin-and-recreate (R&R) algorithm to the $F_m|nwt|\sum C_j$ problem. The remainder of this paper is structured as follows. Following a brief