Applied Soft Computing 24 (2014) 63–77

Contents lists available at ScienceDirect

Applied Soft Computing

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

A hybrid variable neighborhood search for solving the hybrid flow shop scheduling problem

Jun-qing Li [a,b], Quan-ke Pan [a,b,*], Fa-tao Wang [b,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] School of Economics and Management, Beijing University of Posts and Telecommunications, Beijing 100876, PR China

ARTICLE INFO

Article history:
Received 26 June 2013
Received in revised form 3 July 2014
Accepted 5 July 2014
Available online 12 July 2014

Keywords:
Hybrid flow shop scheduling problem
Chemical-reaction optimization
Estimation of distribution
Variable neighborhood search

ABSTRACT

This paper proposes a hybrid variable neighborhood search (HVNS) algorithm that combines the chemical-reaction optimization (CRO) and the estimation of distribution (EDA), for solving the hybrid flow shop (HFS) scheduling problems. The objective is to minimize the maximum completion time. In the proposed algorithm, a well-designed decoding mechanism is presented to schedule jobs with more flexibility. Meanwhile, considering the problem structure, eight neighborhood structures are developed. A kinetic energy sensitive neighborhood change approach is proposed to extract global information and avoid being stuck at the local optima. In addition, contrary to the fixed neighborhood set in traditional VNS, a dynamic neighborhood set update mechanism is utilized to exploit the potential search space. Finally, for the population of local optima solutions, an effective EDA-based global search approach is investigated to direct the search process to promising regions. The proposed algorithm is tested on sets of well-known benchmark instances. Through the analysis of experimental results, the high performance of the proposed HVNS algorithm is shown in comparison with four efficient algorithms from the literature.

Introduction

In modern manufacturing and production systems, production scheduling plays an important role in increasing production efficiency and profit. In recent years, the flow shop scheduling problem (FSSP) has been investigated by more and more researchers because of its important role in a realistic production systems. The hybrid flow shop (HFS) scheduling problem, as one branch of the classical FSSP, is more complex because of the addition of machine selection. In 1988, Gupta proved that HFS is a NP-hard problem [1]. Recent and comprehensive reviews on HFS can be found in [2,3], which illustrate the published HFS literature before 2010. It can be concluded from the two literature reviews that: (1) there are more than 200 current papers discussing the problem; (2) more and more algorithms, including exact algorithms, dispatching rules, heuristics, and meta-heuristics, have been used for solving HFS; and (3) HFS has played an important role in present production systems. For instance, in most production systems with a flexible scheduling environment, such as the oil food, paper, textile, chemical and cosmetic industries, HFS plays a key role and becomes an important factor that can lead to improvements in production efficiency.

The branch and bound (B&B) algorithm proposed for solving the HFS was published in 1970 (Rao [4]). After that, many published papers have discussed the HFS with many different algorithms. We can classify these algorithms by the stage size of the considered problems. There are three types of problem, that is, two-stage, three-stage, and m-stage. The two-stage problem is the HFS with two consecutive stages, while the m-stage problem is the m stage series.

To solve the two-stage HFS, in 1988 Gupta studied an HFS for which there is only one machine at the second stage [1]. In 1991, the B&B approach was introduced by Brash and Hunsucker [5]. After that, many different types of approaches have been developed. In 1994, Gupta solved the two-stage HFS problem with separable setup and removal times [6]. In 2003, Lin and Liao [7] studied the two-stage HFS problem with setup time and dedicated machines. Lee and Kim [8] introduced the B&B approach to solve the two-stage HFS with any number of identical parallel machines at the second stage. Haouari et al. [9] considered the problem without any machine number constraints at any stage.

* Corresponding author at: State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, PR China. Tel.: +8618606356701.
E-mail addresses: lijunqing.cn@gmail.com, p2p_jql@126.com (J.-q. Li), panquanke@mail.neu.edu.cn (Q.-k. Pan).

http://dx.doi.org/10.1016/j.asoc.2014.07.005
1568-4946/

To solve the three-stage HFS, in 1998 Riane et al. [10] developed an efficient heuristic to minimize makespan in a three-stage HFS problem. Jin et al. [11] modeled a real printed circuit board manufacturing system by a HFS with three stages, and solved the problem by GA. Carlier and Neron [12] proposed an exact algorithm for solving the multi-processor flow shop. The benchmark problems generated by them were used by many studies as test problems. In 2004, Babayan and He [13] presented an agent-based approach for a three-stage HFS with identical parallel machines.

To solve the HFS with m-stages, in 1998 Portmann et al. [14] introduced an enhanced version of the branch and bound algorithm crossed with GA. In 2001, Neron et al. [15] proposed an algorithm using energetic reasoning and global operations for an HFS with up to five stages. The HFS with multiprocessor task problems was studied by Oguz and Ercan [16] with a GA approach. Ruiz and Maroto [17] developed a GA for HFS with sequence dependent setup times and machine eligibility. Janiak et al. [18] applied three constructive algorithms and three meta-heuristics based on Tabu Search (TS) and SA, to solve the HFS with cost-related criteria. Niu et al. [19] developed a quantum-inspired immune algorithm for HFS. In 2010, Kahraman et al. [20] investigated a parallel greedy algorithm for solving the multistage HFS. Engin et al. [21] developed an efficient GA for an HFS with multiprocessor task problems. In 2012, Liao et al. [22] proposed an approach using PSO and bottleneck heuristic for the problem. The other meta-heuristics were also used for solving HFS with multi-stages, such as ant colony optimization (ACO) [23,24] and artificial immune systems (AIS) [25,26].

Variable Neighborhood Search (VNS) is a recent and effective meta-heuristic for solving both continuous and global optimization problems. By systematic changes of the neighborhood structures within the search, VNS is capable of escaping from the local optima [27–43]. By simulating the behavior of molecules in chemical reactions, an efficient chemical-reaction optimization (CRO) algorithm was proposed by Lam and Li [44] to optimize combinatorial problems. In recent years, CRO has been applied to solve many continuous and discrete optimization problems [44–46]. The estimation of distribution algorithm (EDA) was introduced by Mühlenbein and Paaß [47]. In EDA, global statistical information was extracted from the current population. Meanwhile, a probabilistic model was constructed that represents the global information and characterizes the distribution of promising solutions in the search region [47–50].

In this study, we make a hybridization of the VNS, CRO, and EDA. To enhance the exploration ability of the proposed algorithm, we assign a certain *KE* value for each individual solution. A kinetic-energy-based neighborhood change procedure is introduced that can avoid being stuck at the local optima and thus further increase the exploration ability. In addition, after collecting a group of local optima solutions by a VNS process, an EDA-based global search process is performed on the local optima population to search global optimal solutions, which can further enhance the exploitation and exploration ability of the proposed algorithm. The rest of this paper is organized as follows: The Problem Formulation deals with the job/task allocation details together with the stage by stage work flow. Next The Related Algorithms deal with how VNS was construed by others. Then the technical approach is briefly discussed in the The Proposed HVNS algorithm. Further, The Experimental Results provide a fair account of results derived and compare them with other algorithms in the literature to demonstrate the performance of HVNS. Finally, Conclusions provide a concise description of solving the hybrid flow shop scheduling problem of our work.

Problem formulation

In an HFS, there are n jobs to be processed on m machines in a predefined order. The m machines are grouped into k stages in series. In each stage i, there are m_i identical machines in parallel, where $m_i \geq 1$, and there are at least two parallel machines in one stage. Each job should visit each stage following the same production flow: stage$_1$, stage$_2$, . . ., stage$_k$. When a job arrives at a stage i, it can select exactly one machine from m_i available identical machines. The assumptions for the considered HFS in this study are given as follow.

- Each machine in the same stage can process only one job at a time;
- Each job can be operated by only one machine at a time;
- All of the jobs and machines are available at time zero;
- Preemption is not allowed, that is, any job cannot be interrupted before the completion of its current operation;
- Setup times are negligible and problem data are deterministic and known in advance;
- Overlap of operations is not allowed, that is, each operation of the same job should not start until the previous operation has completed its work;
- There is an unlimited intermediate buffer between two successive stages.

The aim of HFS is to assign an optimal machine for each job at each stage, and schedule each job on each machine to minimize the makespan or the maximum completion time. The mathematical model can be found in [2,17,22].

Related algorithms

The canonical VNS

In 1997, VNS was introduced by Mladenovic and Hansen [27] for solving the traveling salesman problem (TSP). After that, VNS has been applied by more and more researchers to solve continuous and global optimization problems. In 2010, Hansen et al. [28] gave a comprehensive survey of VNS. The survey literature shows that since it was first proposed in 1997, VNS has seen rapid development and enhancement in two aspects, i.e., methods and applications.

Very recently, VNS has been the focus of substantial research. The recent developments with VNS are also taken in two fields: (1) designing improved VNS by combining other dispatching rules, heuristics, meta-heuristics, and local search methods and (2) applying VNS for new applications. In 2009, Wang and Tang [29] developed a population-based VNS for the single machine total weighted tardiness problem. By combining VNS with GA, Wen et al. [30] presented a hybrid genetic based VNS for task scheduling in a heterogeneous multiprocessor system. Stenger et al. [31] investigated a routing problem arising in the last-mile delivery of small packages by using an improved VNS that embeds cyclic-exchange neighborhoods and an adaptive mechanism. Yazdani et al. [32] developed a parallel VNS for the flexible job-shop scheduling problem (FJSP). Recently, more and more literature has addressed solving scheduling problems using VNS. For parallel machine scheduling problems, Behnamian et al. [33] proposed a hybrid algorithm using ACO, SA and VNS; Driessel et al. [34] developed several variants of VNS; in 2012, Bilyk and Mönch [35] investigated VNS for planning and scheduling of jobs on unrelated parallel machines. For HFS problems utilizing VNS, Naderi et al. [36] proposed a VNS that uses advanced neighborhood search structures for flexible flow line problems with sequence dependent setup time and preventive maintenance activity; Tavakkoli-Moghaddam et al. [37] presented a memetic algorithm for HFS with processor blocking. The other recent applications of VNS include vertex separation problems [38], resource allocation problems [39], TSP [40], location