Information Sciences 181 (2011) 668–685

Contents lists available at ScienceDirect

Information Sciences

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

An effective hybrid discrete differential evolution algorithm for the flow shop scheduling with intermediate buffers

Quan-Ke Pan [a], Ling Wang [b], Liang Gao [c,*], W.D. Li [d]

[a] College of Computer Science, Liaocheng University, Liaocheng 252059, PR China
[b] Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation, Tsinghua University, Beijing 100084, China
[c] State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan 430074, PR China
[d] Faculty of Engineering and Computing, Coventry University, Coventry, UK

A R T I C L E   I N F O

Article history:
Received 24 May 2009
Received in revised form 29 September 2010
Accepted 7 October 2010

Keywords:
Flow shop with intermediate buffers
Makespan
Discrete differential evolution
Hybrid algorithm
Local search

A B S T R A C T

In this paper, an effective hybrid discrete differential evolution (HDDE) algorithm is proposed to minimize the maximum completion time (makespan) for a flow shop scheduling problem with intermediate buffers located between two consecutive machines. Different from traditional differential evolution algorithms, the proposed HDDE algorithm adopted job permutation to represent individuals and applies job-permutation-based mutation and crossover operations to generate new candidate solutions. Moreover, a one-to-one selection scheme with probabilistic jumping is used to determine whether the candidates will become members of the target population in next generation. In addition, an efficient local search algorithm based on both insert and swap neighborhood structures is presented and embedded in the HDDE algorithm to enhance the algorithm's local searching ability. Computational simulations and comparisons based on the well-known benchmark instances are provided. It shows that the proposed HDDE algorithm is not only capable to generate better results than the existing hybrid genetic algorithm and hybrid particle swarm optimization algorithm, but outperforms two recently proposed discrete differential evolution (DDE) algorithms as well. Especially, the HDDE algorithm is able to achieve excellent results for large-scale problems with up to 500 jobs and 20 machines.

1. Introduction

Among all types of scheduling problems, a flow shop scheduling problem with intermediate buffers has important applications in different industries including the petrochemical processing industries and cell manufacturing, where a buffer is either non-existent or with limited size due to limited room and storage facilities (e.g., storage tanks, intermediate inventory) [1–3,19]. In a flow shop with intermediate buffers, each of n jobs consists of m operations with a predetermined processing order through machines, and there exist a certain number of intermediate buffers between two consecutive machines. After finishing processing on a machine, a job is either directly processed on the next machine or it has to be stored in the buffer. If the buffer is full and the next machine is occupied, the job has to wait on its current machine and holds this machine to process other jobs until a buffer unit or the next machine becomes available. Given that the release time of all jobs is zero and setup time on each machine is included in the processing time, the widely used objective for the flow shop scheduling problem with intermediate buffers is to minimize the maximum completion time, i.e., makespan. For the computational complexity of the flow shop problem with intermediate buffers, Papadimitriou and Kanellakis [18]

* Corresponding author. Address: State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, PR China. Tel.: +86 27 87559419; fax: +86 27 87543074.
E-mail address: gaoliang@mail.hust.edu.cn (L. Gao).

0020-0255/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2010.10.009

proved that it was NP-hard even only for two machines. Therefore, it is of significance both in theory and in engineering applications to develop effective, efficient and novel solutions for such a problem.

However, compared with a lot of literature on the classic flow shop scheduling problem, the research on flow shop scheduling with intermediate buffers has not been so extensive. Hall and Sriskandarajah provided a comprehensive survey for scheduling problems with blocking and no-wait in process [6]. Ducclos and Spencer [3] discussed the impact of a constrained buffer in a flow shop. Thoronton and Hunsucker [32] developed a constructive heuristic for hybrid flow shop scheduling without intermediate storage. Leisten [7] presented some priority-based heuristics for both the permutation and general flow shop problems with intermediate buffers, and the numerical experiments indicated that the NEH method [11] was the best heuristic for the three machine examples. A few meta-heuristics were further proposed to solve the flow shop scheduling problem with intermediate buffers. Norman [12] applied the taboo search (TS) algorithm for flow shop scheduling problems containing both sequence-dependent setup time and finite buffers. Smutnicki [26] proposed another TS approach for the two-machine flow shop scheduling problem with intermediate buffers, whereas Nowicki [13] generalized the approach to the problem with an arbitrary number of machine. Later, Brucker et al. [1] further generalized Nowicki's idea to the cases where different job-sequences on machines are allowed. Recently, Wang et al. [34] developed a hybrid genetic algorithm (HGA) for the general flow shop scheduling problem with limited buffers to minimize the makespan. Case and benchmarking studies in the paper proved that the proposed HGA algorithm performed much better than the TS algorithm of Nowicki [13]. More recently, Liu et al. [8] proposed a hybrid particle swarm optimzation (HPSO) algorithm with extensive experiments showing that the HPSO outperformed the HGA algorithm [34] in most cases. For a multi-objective case, a hybrid differential evolution (DE) algorithm was developed by Qian et al. [22].

The DE algorithm was first introduced by Storn and Price [27] for complex continuous optimization problems. Due to its simplicity, easy implementation, fast convergence, and robustness, the DE algorithm has gained increasing attention and a wide range of successful applications were implemented, such as digital PID controller design, digital filter design, earthquake hypocenter location, and etc. [14,21,25,28]. However, due to its continuous nature, the traditional DE algorithm is unable to be directly applied for solving scheduling problems with discrete characteristic [9,17,33]. Therefore, several discrete versions, called the discrete differential evolution (DDE) algorithms, were presented recently. For example, Pan et al. [17] proposed a DDE algorithm, denoted by P_DDE, for solving the permutation flow shop scheduling problem, whereas Wang et al. [33] developed another DDE algorithm (W_DDE for short) for solving the flow shop scheduling problem with blocking constraints. Unlike the traditional DE algorithm, the above two DDE variants represented solutions as discrete job permutations, and presented mutation and crossover operators based on permutations. The mutant scale factor and crossover probability were limited in the range of [0 1]. The DDE algorithms started from a population of initial solutions generated by some heuristics, and generated candidates by using the permutation-based mutation and crossover operators with probability parameters. Like the traditional DE algorithm, a one-to-one selection operator was utilized to decide individuals for the next generation. The computational simulations and comparisons demonstrated that both the algorithms were effective and efficient for the problems considered. Following the successful application of the DDE algorithm, in this paper, we proposed an effective hybrid DDE (HDDE) algorithm for solving flow shop scheduling problems with intermediate buffers to minimize makespan. Different from the above two DDE algorithms, the proposed HDDE algorithm was applies to the newly designed mutation and crossover operators to generate new candidate solutions. A one-to-one selection operator with probabilistic jumping was utilized to decide the target population for next generation. Furthermore, an efficient local search algorithm based on both insertion and swap neighborhood structures was embedded in the algorithm to enhance exploitation. Simulation results and comparisons demonstrate the effectiveness of the proposed HDDE algorithm in the flow shop scheduling problem with intermediate buffers with makespan criterion.

The remaining contents of this paper are organized as follows. In Section 2, the flow shop scheduling problem with intermediate buffers is stated and formulated. In Section 3, the traditional DE is introduced. The HDDE algorithm is proposed and its implementation is explained in detail in Section 4. The computational results and comparisons are provided in Section 5. Finally, we end the paper with some conclusions in Section 6.

2. Problem statement

The flow shop scheduling problem with intermediate buffers can be described as follows. There are n jobs from the set $J = \{1,2,\ldots,n\}$ and m machines from the set $M = \{1,2,\ldots,m\}$. Each job $j \in J$ will be sequentially processed on machines $1,2,\ldots,m$. Operation $o_{j,k}$ corresponds to the processing of job $j \in J$ on machine k ($k = 1,2,\ldots,m$) during an uninterrupted processing period $p_{j,k}$, where $p_{j,k}$ is non-negative. A common simplification measure is to avoid job passing in the sequence, that is, the sequence in which the jobs are to be processed is the same on each machine. At any time, each machine can process at most one job and each job can be processed on at most one machine. Between each successive pairs of machines k and $k + 1$, there exists a buffer with the size equal to $B_k \geqslant 0$ (i.e., at most B_k jobs can be held simultaneously), $k \in M\backslash\{m\}$, and jobs obey the First In First Out rule in the buffer. Each job must go through buffer B_k on its route from machine k to $k + 1$. If B_k is completely occupied, the job has to remain on the current machine k until a free place is available in the buffer. Given that the release times of all jobs are zero and the setup times on each machine are included in the processing times, the aim is to find a sequence for processing all jobs on all machines so that the maximum completion time (i.e. makespan) is minimized.

Let a job permutation $\pi = \{\pi(1),\pi(2),\ldots,\pi(n)\}$ represent the sequence of jobs to be processed, and $d_{\pi(i),k}$ be the departure time of operation $o_{\pi(i),k}$ from machine k, $d_{\pi(i),k}$ can be calculated as follows: