Applied Soft Computing 24 (2014) 1033–1046

Contents lists available at ScienceDirect

Applied Soft Computing

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

Scheduling of no buffer job shop cells with blocking constraints and automated guided vehicles☆

Chengkuan Zeng[a], Jiafu Tang[b,*], Chongjun Yan[b]

[a] Department of Systems Engineering, State Key Lab of Synthetic Automation of Process Industry, Northeastern University (NEU), Shenyang 110004, PR China

[b] College of Management Science and Engineering, Dongbei University of Finance and Economics (DUFE), Shahekou, Dalian 116025, PR China

ARTICLE INFO

Article history:
Received 6 October 2013
Received in revised form 11 June 2014
Accepted 11 August 2014
Available online 23 August 2014

Keywords:
Job shop scheduling
Blocking
AGV
Disjunctive graph model
Local search

ABSTRACT

The blocking job shop (BJS) problem is an extension of a job shop problem with no buffer constraints. It means that after a job is completed on the current machine, it remains on that machine until the next machine becomes available. This paper addresses an extension of the BJS problem, which takes into account transferring jobs between different machines using a limited number of automated guided vehicles (AGV), called a BJS–AGV problem. Two integer non-linear programming (INLP) models are proposed. A two-stage heuristic algorithm that combines an improving timetabling method and a local search is proposed to solve the BJS–AGV problem. A neighborhood structure in the local search is proposed based on a disjunctive graph model. According to the characteristics of the BJS–AGV problem, four principles are proposed to guarantee the feasibility of the search neighborhood. Computation results are presented for a set of benchmarking tests, some of which are enlarged by transportation times between different machines. The numerical results show the effectiveness of the proposed two-stage algorithm.

1. Introduction

The classical job shop (JS) scheduling problem is one of the hardest combinatorial optimization problems and has a broad engineering background [1–3]. With the development of manufacturing technology and demands, extensions of the JS problem have been considered. To reduce production expenses, just-in-time (JIT) manufacturing philosophies have been implemented, which maintain a limited amount of process inventory or buffer [4]. Buffer is defined as the amount of free machine space on which parts are allowed to wait, either after the completion of processing on a machine or on the next machine. These are called input and output buffers, respectively. JIT with limited buffers or no buffers has been investigated by many researchers [4,7,8,11–14]. When dealing with no buffer conditions, JS problems will be under blocking or no wait constraints depending on whether parts are allowed to wait on the machines [15]. The problem considered in this paper assumes blocking constraints.

Scheduling with blocking constraints means that after a job has completed its current machine operation, it remains on the present machine until the next machine becomes available for processing. Most of the exist researches related to limited buffer or blocking are on flow shop scheduling [5–8]. Liu et al. [5] propose an effective hybrid algorithm based on particle swarm optimization for permutation flow shop scheduling problem with the limited buffer between consecutive machines to minimize make-span. Both Qian et al. [6] and Pan et al. [9] propose effective hybrid algorithm using differential evolution to deal with the same problem as Liu et al. [5]. Wang et al. [10,11] propose a novel hybrid discrete differential evolution algorithm and a hybrid modified global-best harmony search algorithm, respectively, for solving the blocking permutation flow shop scheduling with the makespan criterion. Liang et al. [12] present a dynamic multi-swarm particle swarm optimizer for solving the same problem as Wang et al. [10]. Pan et al. [13,14] also present three improved heuristics basing on two simple constructive heuristics and a mimetic algorithm, respectively. Job shop problems with blocking constraints (BJS) have been investigated by several researchers. Mascis et al. [15] studied the BJS problem and formulated it with an alternative graph and proposed several heuristic algorithms. Brucker et al. [16] propose a tabu search algorithm to solve the BJS problem for cyclical scheduling. GrÖflin et al. [17] and [18] address the BJS problem and the flexible blocking job shop (FBJS) problem, by taking into account the transfer time associated with moving a job from one machine to another, and sequence-dependent setup times between consecutive operations

☆ This paper is financially supported by NSFC of China (71420107028).
* Corresponding author. Tel.: +86 411 84713571; fax: +86 411 84710475.
E-mail addresses: tangjiafu@dufe.edu.cn, jftang@mail.neu.edu.cn (J. Tang).

http://dx.doi.org/10.1016/j.asoc.2014.08.028
1568-4946/

on a machine. AitZai [19] proposes a combination of a branch and bound algorithm with alternative graphs and develops two methods based on genetic algorithms to solve the BJS problem. Hayasaja et al. [20] describe a procedure for local modifications of the planned JS schedule under no buffer constraints. From GrÖflin et al. [17] and [18], the transportation time between different machines is shown to be a considerable element of the BJS problem. However, the source of transportation media is found to be sufficient ([17,18]). More applications related to the BJS problem have also been reported in the processing and logistics industries, such as scheduling for the manufacturing of concrete blocks by Grabowski et al. [21], steel-making by Pacciarelli et al. [22], chemical batch production by Romero et al. [23], container handling at a port by Chen et al. [24] and railway networks by D′Ariano et al. [25].

In practical production, the source of transferring media is time restricted and uses robots or automated guided vehicles (AGV). It is convenient to use robots or AGVs as transferring media to transfer jobs between different machines, and in general, the number of the robots or AGVs is also restricted. Such situations have already appeared in flexible manufacturing systems (FMSs) and robotic cells (RCs). FMSs are highly automated production systems producing a wide variety of job types [26]. Blazewicz et al. [27] propose a dynamic programming approach to construct optimal machine and vehicle schedules. Corréa et al. [28] propose a hybrid method designed to solve the problem of dispatching and conflict free routing for automated guided vehicles (AGVs) in a FMS. Caumond et al. [29] use a mixed integer linear program (MILP) to find the optimal solutions for FMS scheduling problems with one vehicle. Zhang et al. [26] propose a model for a flexible job shop scheduling problem with transportation constraints and bounded processing times. A manufacturing cell consisting of a number of machines and material handling performed by one (or a few) robots or AGVs is called a robotic cell [30]. Robotic cells with no storage buffers between machines are widely used in practice and are called no wait or bufferless RCs [26]. Che et al. [31] address a multi-robot, 2-cycle problem in a no wait-RC and develop a polynomial algorithm to find the minimum number of robots for all feasible cycle times. Brauner et al. [32] describe a new class of equivalent one-machine, no wait-RC problems and propose new algorithm procedures to solve this RC problem. Ulusoy et al. [33] propose a genetic algorithm to schedule a flexible manufacturing system consisting of several machine centers and two identical automated guided vehicles to minimize make-span. Abdelmaguid et al. [34], Reddy et al. [35], Deroussi et al. [36], Chaudhry et al. [37] and Lacomme et al. [38] address this same problem and propose several heuristic algorithms. Zhang et al. [39] present a Pareto-based genetic algorithm that incorporates a local search module to solve a job shop scheduling problem with two new objective functions related to vehicles.

To the best of our knowledge, no research has been performed on BJS problems that consider transferring jobs using a limited number of AGVs. Based the above discussion, this is an important problem. Brucker et al. [40] have already described cyclic job-shop problems with one robot and blocking. Therefore, this paper addresses the BJS–AGV problem with the objective of minimizing the make-span. Two integer non-linear programming (INLP) models are proposed to describe the BJS–AGV problem. A two-stage heuristic algorithm that combines an improved timetabling method and local search is proposed to solve the BJS–AGV. The improved timetabling method is proposed based on non-delay timetabling to find a feasible solution for the BJS–AGV. The neighborhood in the local search is the proposed base for a disjunctive graph model. According to the characteristics of the BJS–AGV problem, four principles are proposed to guarantee the feasibility of the neighborhood in the local search. Computation results are presented for a set of benchmarks, some of which are enlarged by transportation time between different machines. The results show the effectiveness of the proposed two-stage algorithm.

This paper is organized as follows. Section 2 discusses the definition and assumptions of the BJS–AGV problem and describes the BJS–AGV problem using the two INLP models. Section 3 introduces the structure and procedure for the improved timetabling method to find a feasible solution. Section 4 introduces the theory and structure of the neighborhood in the local search to optimize the base in disjunctive graph model and the four principles that are necessary to guarantee the feasibility of the proposed neighborhood. Section 5 discusses a series of experimental tests and computational results, showing the effectiveness of the proposed two-stage heuristic algorithm. Finally, Section 6 discusses the conclusions of this paper.

2. Problem description and assumption

The classical job shop problem with no buffer constraints, transportation time and several AGVs can be described as follows: there is a set of n jobs ($J = \{1,2,\ldots,n\}$) required to be processed on a set of m machines ($M = \{1,2,\ldots,m\}$). The jobs are transferred between

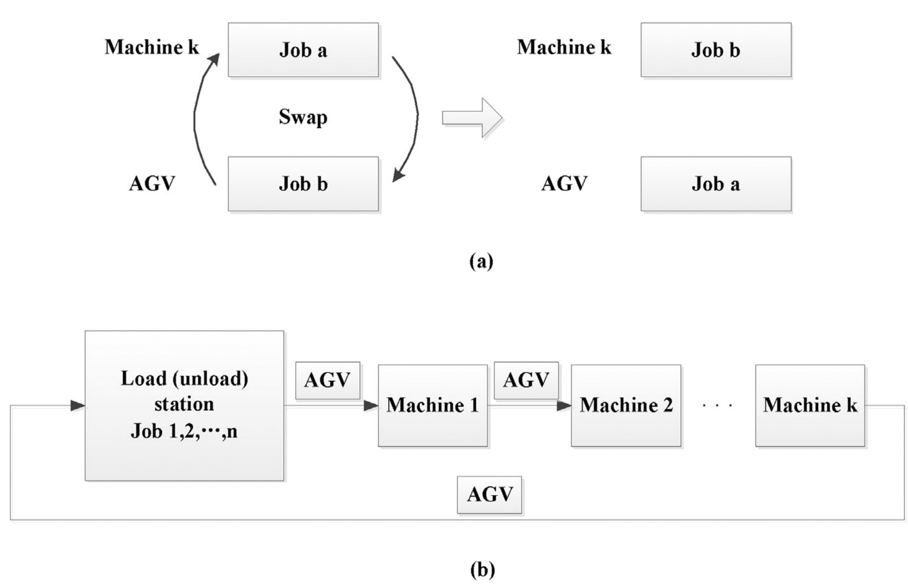

Fig. 1. Example of swapping and AGV transferring job between L/U station.