Flexible job-shop scheduling with parallel variable neighborhood search algorithm

M. Yazdani [a], M. Amiri [b], M. Zandieh [c,*]

[a] *Faculty of Industrial and Mechanical Engineering, Qazvin Islamic Azad University, Qazvin, Iran*
[b] *Department of Industrial Management, Management and Accounting Faculty, Allameh Tabatabaei University, Tehran, Iran*
[c] *Department of Industrial Management, Management and Accounting Faculty, Shahid Beheshti University, G.C., Tehran, Iran*

ARTICLE INFO

ABSTRACT

Flexible job-shop scheduling problem (FJSP) is an extension of the classical job-shop scheduling problem. FJSP is NP-hard and mainly presents two difficulties. The first one is to assign each operation to a machine out of a set of capable machines, and the second one deals with sequencing the assigned operations on the machines. This paper proposes a parallel variable neighborhood search (PVNS) algorithm that solves the FJSP to minimize makespan time. Parallelization in this algorithm is based on the application of multiple independent searches increasing the exploration in the search space. The proposed PVNS uses various neighborhood structures which carry the responsibility of making changes in assignment and sequencing of operations for generating neighboring solutions. The results obtained from the computational study have shown that the proposed algorithm is a viable and effective approach for the FJSP.

© 2009 Published by Elsevier Ltd.

1. Introduction

Scheduling is the allocation of resources to perform a set of tasks over a period of time. Many real scheduling problems in the manufacturing industries are quite complex and very difficult to be solved by conventional optimization techniques (Watanabe, Ida, & Gen, 2005). The complexity of these problems has a direct dependence on constraints and shop environments upon which these problems are defined. The job-shop scheduling problem (JSP) is one of the hardest combinatorial optimization problems in the field of scheduling. JSP consists in performing a set of n jobs on a set of m machines, where the processing of each job i is composed of n_i operations performed on these machines. This problem aims to find the appropriate sequencing of operations on the machines to optimize the performance indicator. Herein, a typical performance indicator for JSP is the makespan, which is defined as the time needed to complete all the jobs. It is well known that this problem is NP-hard (Garey, Johnson, & Sethi, 1976).

In order to match nowadays market requirements, manufacturing systems have to become more flexible and efficient. To achieve these objectives, the systems need not only the automated and flexible machines, but also the flexible scheduling systems. The flexible job-shop scheduling problem (FJSP) is a generalization of the classical JSP, where operations are allowed to be processed on any among a set of available machines. In fact, the FJSP mainly presents two difficulties. The first one is to assign each operation to

a machine out of a set of capable machines, and the second one is to sequence the assigned operations on all machines. The FJSP is a much more complex version of the JSP, so the FJSP is strongly NP-hard and combinatorial.

Although the complexity of FJSP is well-known, there are far more literatures focusing on JSP rather than on FJSP. Brucker and Schlie (1990) are the pioneers in addressing FJSP. They developed a polynomial algorithm for solving this problem with two jobs. Also various heuristic methods such as dispatching rules and local search and meta-heuristic methods such as tabu search (TS), simulated annealing (SA), and genetic algorithm (GA) have been applied for solving FJSP in recent years. These heuristic and meta-heuristic algorithms can be classified into two main categories: hierarchical approaches and integrated approaches. In hierarchical approaches, the assignment of operations to machines and the sequencing of operations on the machines are treated separately, whereas in integrated approaches, the assignment and sequencing problems are not separated. The hierarchical approaches are based on the idea of decomposing the original problem in order to reduce its complexity. Despite the fact that the integrated approaches are much more difficult to solve, they generally achieve better results.

Brandimarte (1993) applied a hierarchical approach for FJSP based on decomposition. He solved the routing sub-problem using some existing dispatching rules and then concentrated on sequencing sub-problem which is solved by using a TS algorithm. Hurink, Jurisch, and Thole (1994) based on the integrated approach and Barnes and Chambers (1996) based on the hierarchical approach developed TS algorithms to solve FJSP. Dauzére-Pérés and Paulli (1997) defined a new neighborhood structure for the

* Corresponding author.
  E-mail address: m_zandieh@sbu.ac.ir (M. Zandieh).

problem where there is no distinction between reassigning and resequencing an operation, and proposed a TS algorithm based on the integrated approach. Mastrolilli and Gambardella (2000) improved Dauzére-Pérés TS techniques and presented two neighborhood functions. Chen, Ihlow, and Lehmann (1999) developed a GA for FJSP. They split the chromosome representation into two parts, the first one defined the routing policy, and the second one delineated the sequence of operations on each machine. Kacem, Hammadi, and Borne (2002a, 2002b) proposed a GA controlled by the assigned model which was generated by the approach of localization and they used it to solve mono-objective and multi-objective FJSP. Zhang and Gen (2005) proposed a multistage operation-based GA to deal with the problem from the point view of dynamic programming. Xia and Wu (2005) worked on a hierarchical approach for solving multi-objective FJSP. The approach made use of particle swarm optimization (PSO) to assign operations to machines and SA algorithm to schedule operations on each machine. Fattahi, Saidi Mehrabad, and Jolai (2007) proposed a mathematical model and two meta-heuristics algorithms (SA and TS) for solving FJSP. Moreover, considering two developed meta-heuristics and concerning the integrated and hierarchical approaches, they presented six different algorithms. Pezzella, Morganti, and Ciaschetti (2008) propounded a GA for FJSP based on the integrated approach, in which a mix of different strategies for generating the initial population, selecting individuals for reproduction, and reproducing new individuals is presented. Gao, Sun, and Gen (2008) studied the FJSP with three objectives: min makespan, min maximal machine workload and min total workload. They developed a hybrid genetic algorithm (hGA) based on the integrated approach for this problem. Given the literature of FJSP, the heuristics and meta-heuristics have corroborated their merits to solve the FJSP. Variable neighborhood search (VNS) (Amiri et al., 2009; Hansen & Mladenovic, 2001; Mladenovic & Hansen, 1997) is a modern meta-heuristic based on systematic changes of the neighborhood structure within a search to solve combinatorial optimization problems. In this paper, we develop a parallel variable neighborhood search (PVNS) algorithm for solving FJSP. Parallelization in the presented optimization method increases the diversification and the exploration in the search space. In this method, neighborhood structures related to sequencing and assignment problems are employed to generate a neighboring solution. Considering the aforementioned neighborhood structures, the proposed algorithm executes parallel search process in the solution space.

The rest of the paper is organized as follows. In Section 2, we present the formulation and constraints of the problem. Section 3 which is subdivided into five subdivisions explains proposed PVNS algorithm for solving FJSP. In Section 4, the computational study performed with the proposed algorithm and its results are reported. The conclusion and some further research suggestions are presented in Section 5.

2. Problem description

The FJSP consists in performing a set of n jobs $J = \{J_1, J_2, \ldots, J_n\}$ on a set of m machines $Ma = \{M_1, M_2, \ldots, M_m\}$. A job J_i is formed by a sequence of operations $(O_{i,1}, O_{i,2}, \ldots, O_{i,n_i})$ to be performed one after the other according to the given sequence. Each operation $O_{i,j}$, i.e., the operation j of job i, can be executed on any among a subset of compatible machines $(Ma_{i,j} \subseteq Ma)$. All jobs and machines are available at time 0, and a machine can only execute one operation at a given time. There are no precedence constraints among the operations of different jobs. Nonetheless, there are precedence constraints among the operations of the same job. Preemption is not allowed, i.e., each operation must be completed without interruption once it starts. Setup time of machines and move time between operations are negligible. The processing time of operation $O_{i,j}$ on machine M_r is $P_{i,j,r} > 0$.

The problem is to assign each operation to an appropriate machine, and to sequence the operations on the machines in order to minimize the makespan, i.e., the time needed to complete all the jobs, which is defined as $C_{max} = \max(C_i)$, where C_i is the completion time of job J_i.

Flexibility of FJSP can be categorized into partial flexibility and total flexibility. It is partial when there exists a proper subset $Ma_{i,j}$ of Ma, for at least one operation $O_{i,j}$, and it is total when we have $Ma_{i,j} = Ma$ for all operations. Data related to a given problem with partial flexibility is shown in Table 1, in which rows correspond to operations and columns represent machines. In this Table, symbol ∞ means that a machine can not execute the corresponding operation. In other words, it does not belong to the subset of compatible machines for that operation.

3. A parallel variable neighborhood search for FJSP

As it happens to meta-heuristics, it takes a long time to reach some useful solutions, while solving some sort of hard combinatorial optimization problems. To overcome this trouble, we can make use of parallelization as one of the most considerable policies. Parallel meta-heuristic allows us to run concurrent multiple searches over the domain of the problem; each of these searches is carried out in different regions of solution space. The application of the parallelization to a meta-heuristic algorithm causes the reduction of the computational time (the partition of the sequential program) or the increase of the exploration in the search space (the application of independent search threads). To search efficiently in the solution space and attain solutions with high quality in FJSP, we propose parallel variable neighborhood search (PVNS) algorithm. Several strategies for parallelizing a VNS algorithm have been presented in the literature (Crainic, Gendreau, Hansen, & Mladenovic, 2004; Garcia Lopez, Melian Batista, Moreno Perez, & Moreno Vega, 2002). In our algorithm, parallelization is based on the application of multiple independent searches increasing the exploration in the search space. Section 3 is subdivided into the following five subdivisions:

Basic structures of VNS algorithm are discussed in Section 3.1; Section 3.2 provides solution representation method; we describe, in Section 3.3, how the initial solution is generated; neighborhood structures employed in the presented algorithm are explained in detail in Section 3.4. In Section 3.5, a comprehensive description of the proposed PVNS algorithm is given.

3.1. Variable neighborhood search

The basic steps of VNS meta-heuristic are shown in Fig. 1. During the initialization step, a set of neighborhood structures and the sequence of their implementations are determined. In this step,

Table 1
Processing time table.

Operations	Machines			
	M_1	M_2	M_3	M_4
$O_{1,1}$	4	7	6	5
$O_{1,2}$	2	6	∞	5
$O_{2,1}$	4	5	7	∞
$O_{2,2}$	5	∞	6	3
$O_{2,3}$	∞	5	4	7
$O_{3,1}$	5	3	∞	6
$O_{3,2}$	∞	∞	4	∞
$O_{4,1}$	2	4	∞	5
$O_{4,2}$	∞	4	2	∞
$O_{4,3}$	5	4	6	3