Applied Soft Computing 13 (2013) 1400–1406

Contents lists available at SciVerse ScienceDirect

Applied Soft Computing

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

Inventory based two-objective job shop scheduling model and its hybrid genetic algorithm

Ren Qing-dao-er-ji, Yuping Wang*, Xiaoli Wang

School of Computer Science and Technology, School of Science, Xidian University, Xi'an, 710071, China

ARTICLE INFO

Article history:
Received 26 December 2011
Received in revised form 17 March 2012
Accepted 22 March 2012
Available online 26 April 2012

Keywords:
Multi-objective job shop scheduling problem
Inventory capacity
Genetic algorithm
Genetic operator

ABSTRACT

Job shop scheduling problem is a typical NP-hard problem. An inventory based two-objective job shop scheduling model was proposed in this paper, in which both the make-span (the total completion time) and the inventory capacity were as objectives and were optimized simultaneously. To solve the proposed model more effectively, some tailor made genetic operators were designed by making full use of the characteristics of the problem. Concretely, a new crossover operator based on the critical path was specifically designed. Furthermore, a local search operator was designed, which can improve the local search ability of GA greatly. Based on all these, a hybrid genetic algorithm was proposed. The computer simulations were made on a set of benchmark problems and the results demonstrated the effectiveness of the proposed algorithm.

1. Introduction

The job shop scheduling problem (JSSP) is one of the most well-known scheduling problems with strong engineering backgrounds and has been proved to be NP-hard [1]. Since the benchmarks of JSSP were presented by Fisher and Thompson in 1963 [2], the single objective JSSPs have attracted wide research attention. Most studies of single objective JSSPs result in a schedule to minimize the time required to complete all jobs, i.e., to minimize the makespan. Historically, JSSP has been primarily treated by exact methods [3], such as branch and bound, linear programming and Lagrangian relaxation. Over the past two decades, meta-heuristics have gained wide research attention [4–6], including such topics as the shifting bottleneck approach, particle swarm optimization, simulated annealing, Tabu search and genetic algorithm.

However, in many real-life scheduling problems, the decision makers are frequently faced with situations in which the appropriateness of a schedule is measured against multiple objectives. This means the final schedule must consider various objectives simultaneously. Consequently, scheduling falls into the category of multi-objective optimization problems. So far, few researchers have attempted to tackle the multi-objective JSSPs (MJSSPs). The importance of multi-criterion scheduling problems and some techniques to find solutions were briefly summarized in Nagar et al. [7]. Sakawa and Kubota [8] presented a genetic algorithm incorporating the concept of similarity among individuals. The objectives were to maximize the weighted sum of the minimum agreement index, the average agreement index and the maximum fuzzy completion time. Ponnambalam et al. [9] proposed a multi-objective genetic algorithm and the objectives were to minimize the weighted sum of makespan, the total idle time of machines and the total tardiness. Esquivel et al. [10] proposed an enhanced evolutionary algorithm with new multi-re-combinative operators and incest prevention strategies for single and multi-objective job shop scheduling problem. Low et al. [11] proposed a mathematical model for MOJSSP to minimize the total job flow time, total job tardiness, and machine idle time. Lei and Wu [12] developed a crowding measure-based multi-objective evolutionary algorithm for the problems of job shop scheduling to minimize the makespan and the total tardiness of jobs. A Pareto archived simulated annealing method was developed by Suresh and Mohanasndaram [13] to find non-dominated solution sets for the MOJSSP with the objectives of minimizing the makespan and the mean flow time of jobs. Ripon et al. [14] presented a jumping genes genetic algorithm by imitating a jumping gene phenomenon to solve MOJSSP. Petrovic et al. [15] used GA with a linguistically quantified decision function for solving a MOJSSP. Lei [16] presented a particle swarm optimization for the MOJSSP to minimize makespan and total job tardiness simultaneously. Qian et al. [17] proposed a memetic algorithm based on differential evolution for MOJSSP. Kachitvichyanukul and Sitthitham [18] presented a two-stage genetic algorithm (2S-GA) for MOJSSP. The 2S-GA was proposed with three criteria: minimizing makespan, minimizing total weighted earliness, and minimizing

* Corresponding author.
E-mail addresses: renqingln@sina.com (R. Qing-dao-er-ji), ywang@xidian.edu.cn (Y. Wang).

1568-4946/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.asoc.2012.03.073

total weighted tardiness. Sha and Lin [19] constructed a particle swarm optimization for an elaborate MOJSSP and modified the particle position representation, particle movement, and particle velocity. Vázquez-Rodríguez and Petrovic [20] proposed a new hybrid dispatching rule based genetic algorithms. Tavakkoli-Moghaddam et al. [21] presented a combination of particle swarm optimization and genetic operators for MOJSSP that minimizes the mean weighted completion time and the sum of the weighted tardiness/earliness costs. Huang [22] focused on lot splitting in the job-shop scheduling problem and attempted to minimize the weighted total of stock, machine idle and carrying costs. Adibi et al. [23] applied a multi-objective performance measure to the MOJJSP as objective function that consists of makespan and tardiness. Moslehi and Mahnam [24] present a new approach based on a hybridization of the particle swarm and local search algorithm to solve the multi-objective flexible job-shop scheduling problem. Fang et al. [25] proposed an optimization method of many goals scheduling based on grey relation theory and ant colony algorithm for the MOJJSP.

Previous literature indicates that the completion time, tardiness, flow time, lateness, earliness and the number of tardy jobs are often used as the optimization criteria in MOJSSP. These objectives are constructed based on the completion situation of the jobs and are hardly taken the inventory capacity into account. So in this paper, we considered both the completion situation of the jobs and the inventory capacity in the objectives, constructed an inventory based MOJJSP model. In order to solve the proposed MOJJSP model more effectively, some genetic operators was designed. Based on these genetic operators, we proposed a hybrid genetic algorithm (HGA). Finally, the efficiency of the proposed algorithm was verified by computer simulations on some typical scheduling problems.

The remainder of the paper is organized as follows. We first introduce the concepts of multi-objective problems in Section 2. Then, we describe the problem considered and its mathematical model in Section 3. In Section 4, a hybrid genetic algorithm to the MOJSSP is presented. Section 5 presents the experimental results. The conclusions are made in Section 6.

2. Multi-objective optimization and Pareto optimality

Generally, a multi-objective optimization problem with k objectives can be described as follows [16]:

$$\text{Minimize}\, y = f(\vec{X}) = [f_1(\vec{X}), f_2(\vec{X}), \ldots, f_k(\vec{X})]$$

$$\text{Subject to}\, g_i(\vec{X}) \geq 0 \quad i = 1, 2, \ldots, D \tag{1}$$

where $\vec{X} = (x_1, x_2, \ldots, x_n)^T$ is called decision vector, $\vec{X} \in \Theta \subset R^n$, Θ is search space. $y \in Y$ is objective vector and Y is objective space. g_i, $i = 1, 2, \ldots, D$ is a constraint function.

The following basic concepts are often used in multi-objective optimization problems.

Definition 1. Let a decision vector $\vec{X}_1 \in \Theta$

1. $\vec{X}_1$ is said to dominate a decision vector $\vec{X}_2 \in \Theta (\vec{X}_1 \prec \vec{X}_2)$ if and only if $f_i(\vec{X}_1) \leq f_i(\vec{X}_2)$ $i = 1, 2, \ldots, k$, and $\exists i \in \{1, 2, \ldots, k\}$ satisfying $f_i(\vec{X}_1) < f_i(\vec{X}_2)$.
2. $\vec{X}_1$ is said to be Pareto optimal if and only if $\neg\exists \vec{X}_2 \in \Theta$ satisfying $\vec{X}_2 \prec \vec{X}_1$.
3. $P_S = \{\vec{X}_1 \in \Theta \,|\, \neg\exists \vec{X}_2 \in \Theta$ satisfying $\vec{X}_2 \prec \vec{X}_1\}$ is said to be Pareto optimal set of all Pareto optimal decision vectors.
4. $P_F = \left\{ f(\vec{X}) = (f_1(\vec{X}), f_2(\vec{X}), \ldots, f_k(\vec{X})) \,\middle|\, \vec{X} \in P_S \right\}$ is said to be Pareto optimal front of all objective function values corresponding to the decision vectors in P_S.
5. $\vec{X}_1$ is said to be non-dominated regarding a given set if $\vec{X}_1$ is not dominated by any decision vectors in the set.

Pareto optimal decision vector cannot be improved in any objectives without causing degradation in at least one other objective. When a decision vector is non-dominated on the whole search space, it is Pareto optimal.

3. Problem definition and mathematical modeling

Usually, the JSSP with which we are concerned can be described as follows [26]. Given a set of m machines and a set of n jobs; each job consists of m operations that have to be processed in a specified sequence; each operation has to be processed on a definite machine without preemption and has a processing time which is known. A schedule defines the time intervals in which the operations are processed and is feasible only if it complies with the following constraints:

- Each machine can only process one operation at a time and the operation sequence is respected for every job.
- A job can visit a machine once and only once.
- There are no precedence constraints among the operations of different jobs.
- Preemption of operations is not allowed.
- Each job can be processed by only one machine at a time.
- Neither release times nor due dates are specified.

The objective of JSSP is to find the optimal schedule, i.e., the schedule of the operation sequences and starting time on each machine so that one or more given criteria are optimized.

In the most of the previous MOJSSPs, the completion time, tardiness, flow time, lateness, earliness and the number of tardy jobs are often used as the optimization criteria. These objectives are constructed based on the completion of the jobs and are almost not taken the inventory capacity into account. Of course, the completion of each job is our top priority goal, but the inventory capacity is also an important issue which should not be ignored. Especially for those factories with high inventory costs but very low transportation costs, if the volume (or size) of each operation is large, then we need a very large depot to put all the operations, and this will increase the inventory cost. In order to reduce the inventory cost, we can try to use as little space of depot as possible by delivering the operations required in batches to the depot, and transported them away after process. Of course, the operations are different for different schedules at a fixed time, and what we need to do is finding a reasonable scheduling scheme with the inventory capacity needed as small as possible. Thus, we can construct an inventory based MOJJSP model to minimize the makespan and the inventory capacity simultaneously.

Given a feasible schedule, we can know the operation sequences and the completion time on each machine. We introduce several symbols at first.

- O_{ik} : The operation of job i when processed on machine k.
- c_{ik} : The completion time of operation O_{ik}.
- p_{ik} : The processing time of operation O_{ik}.
- W_{ik} : The volume of operation O_{ik}.

We use $C_{\max} = \max\limits_{1 \leq k \leq m} \left\{ \max\limits_{1 \leq i \leq n} \{c_{ik}\} \right\}$ to represent the maximum makespan and $S_{\max} = \max\limits_{0 \leq p \leq C_{\max}} \sum\limits_{i=1}^{n} \sum\limits_{k=1}^{m} (N_{ikp} \times W_{ik})$ to represent the maximum inventory capacity needed, where $N_{ikp} = \begin{cases} 1, & c_{ik} - p_{ik} = p \\ 0, & c_{ik} - p_{ik} \neq p \end{cases}$, $i = 1, 2, \ldots, n$, $k = 1, 2, \ldots, m$, $p = 0, 1, \ldots, c_{\max}$.