Int. J. Production Economics 141 (2013) 87–98

Contents lists available at SciVerse ScienceDirect

Int. J. Production Economics

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

A simple and effective evolutionary algorithm for multiobjective flexible job shop scheduling

Tsung-Che Chiang *, Hsiao-Jou Lin

Department of Computer Science and Information Engineering, National Taiwan Normal University, Taiwan, ROC

ARTICLE INFO

Article history:
Received 16 October 2011
Accepted 31 March 2012
Available online 11 May 2012

Keywords:
Flexible job shop scheduling
Multiobjective optimization
Pareto optimal
Evolutionary algorithm

ABSTRACT

This paper addresses the multiobjective flexible job shop scheduling problem (MOFJSP) regarding minimizing the makespan, total workload, and maximum workload. The problem is solved in a Pareto manner, whose goal is to seek for the set of Pareto optimal solutions. We propose a multiobjective evolutionary algorithm, which utilizes effective genetic operators and maintains population diversity carefully. A main feature of the proposed algorithm is its simplicity—it needs only two parameters. Performance of our algorithm is compared with seven state-of-the-art algorithms on fifteen popular benchmark instances. Only our algorithm can find 70% or more non-dominated solutions for every instance.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

A job shop is a multi-stage production system. Each job needs to undergo several operations to become a finished product. In a job shop, only a single machine is capable of processing each operation. This one-to-one relationship will cause blocking of production when any machine breaks down. To reduce the risk of blocking, a flexible job shop forms a group of capable machines for each operation. The term "flexible" comes from the flexibility of routing jobs. If each machine is capable of processing all operations, the shop is totally flexible; otherwise, it is partially flexible. Scheduling is always one of the keys to the success of a production system. Properly utilizing the resources increases machine utilization, reduces work-in-process (WIP) level, shortens time to market, and meets customers' demands. In a job shop, scheduling mainly refers to the task of sequencing operations on the machines. In a flexible job shop, the tasks include not only operation sequencing but also machine assignment (routing). Job shop scheduling has been proven to be NP-hard (Garey et al., 1976). Flexible job shop scheduling, as an extended problem, should also be a hard problem. Mati and Xie (2004) proved that flexible job shop scheduling with two machines is NP-hard.

In the industry, production managers are usually concerned about more than one objective. Multiobjective scheduling meets the practical needs and has been studied in many different production environments, such as parallel machines (Chang et al., 2005), flow shops (Chang et al., 2002), job shops (Chiang and Fu, 2006), and batch machines (Reichelt and Mönch, 2006). In the last decade, there were also many studies on multiobjective FJSP (MOFJSP). One approach to deal with multiple objectives is through aggregation of the objective values. A common aggregation function is the linear weighted summation. However, linear summation might not always be able to represent the trade-off relationship between the objectives. Besides, determination of weights on objectives is not an easy job. Users may need to run the aggregation-based approaches many times to obtain a satisfactory solution. The Pareto approach provides an alternative to multiobjective optimization. Solutions are compared based on the Pareto dominance relation. We say that a solution x dominates a solution y if x is not worse than y for all objective values and is better than y for at least one objective value. A solution is Pareto optimal iff it is not dominated by any solution. In contrast to the aggregation-based approach, which seeks for a single optimal solution in terms of the aggregated objective, the Pareto approach seeks for the set of Pareto optimal solutions. Running the Pareto approach needs no *a priori* information. It can show the trade-off between objectives through the distribution of obtained solutions in a single run. This helps users to evaluate solutions. When the number of Pareto optimal solutions is small, users can even select the favorite solution directly.

In this study, we adopt the Pareto approach and propose a multiobjective evolutionary algorithm (MOEA) to solve the MOFJSP. The three concerned objectives are makespan, total workload, and maximum workload, whose definitions will be given in Section 3. The quality and diversity of the initial population is important for an MOEA to achieve good performance. We propose an adaptive procedure to generate the initial population properly. It utilizes

* Correspondence to: No. 88, Sec. 4, Tingzhou Rd., Wenshan District, Taipei City, Taiwan, ROC. Tel.: +886 2 77346692; fax: +886 2 29322378.
E-mail address: tcchiang@ieee.org (T.-C. Chiang).

0925-5273/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ijpe.2012.03.034

several promising heuristics and adjusts the ratio of usage depending on the uniqueness of the generated solutions. To avoid the premature convergence, we apply effective mutation operators to the duplicate individuals in the population and allow them to participate in the evolutionary process. Performance of the proposed MOEA is verified by comparing with seven state-of-the-art algorithms on fifteen classic problem instances. Our algorithm is the only one that is able to find at least 70% of the non-dominated solutions for all instances. Another advantage of our algorithm is the ease of use. Most of the metaheuristic-based approaches have five or more parameters, and parameter tuning of these algorithms could be a time-consuming process. By careful designs, the proposed MOEA has only two parameters and still performs well.

The rest of this paper is organized as follows. Section 2 gives a literature review on (MO)FJSP. Problem descriptions and the definitions of objective functions are presented in Section 3. Section 4 details the proposed simple evolutionary algorithm (SEA). Experiments and results are presented in Section 5, and conclusions and future work are given in Section 6.

2. Literature review

Brandimarte (1993) solved the FJSP considering makespan minimization. He used dispatching rules to generate an initial solution. Then, the routing decisions in the initial solution were fixed to derive a job shop scheduling problem (JSP). A tabu search (TS) was applied to solve the JSP. He also proposed an advanced approach with two levels of TS. The high-level TS was responsible for re-routing. It generated a set of neighboring solutions by inserting all operations on the critical path to all possible positions on all eligible machines. The solution with the minimal makespan served as the initial solution of the low-level TS, whose task was re-sequencing. This approach resolved routing and sequencing separately and alternatively. A special case of FJSP was addressed by Hurink et al. (1994). They applied TS in some different ways from what Brandimarte did. They generated the initial solution by a beam search-based procedure, reduced the neighborhood size, and changed routing and sequencing decisions simultaneously. Effective neighborhood functions help the TS to find high-quality solutions in a short time. Mastrolilli and Gambardella (2000) proposed two neighborhood functions. The size of neighborhood was reduced, and meanwhile the neighboring solution with the lowest makespan was kept in the neighborhood.

A cultural algorithm was developed by Ho and Tay (2004) to minimize the makespan in flexible job shops. The algorithm maintained two belief spaces for mutation and environmental selection. Pezzella et al. (2008) solved the FJSP by a genetic algorithm (GA). Their algorithm exploited a lot of domain knowledge to generate the initial population. The approach by localization (AL) (Kacem et al., 2002a) was adopted to make routing decisions, and three dispatching rules were taken to make sequencing decisions. Xing et al. (2010) proposed an ant colony optimization (ACO) algorithm. It adjusted parameters dynamically according to the optimization performance, which was measured by the number of iterations where the best solution was updated. Bagheri et al. (2010) proposed an artificial immune algorithm. They followed Pezzella et al. approach to generate the initial population. Four mutation operators were applied, and receptor editing introduced randomly produced solutions into the population. Zhang et al. (2011) presented a GA, in which a solution was represented by two strings for routing and sequencing separately. They proposed a modified version of the AL to do routing for the initial population.

Recently, parallel metaheuristics started to attract attention of researchers of FJSP. Bozejko et al. (2010) proposed two double-level parallel metaheuristics and implemented them on the environment of general purpose graphics processing units (GPGPU). They also proposed a reduced neighborhood function about re-positioning of the head or tail operations in critical blocks. Yazdani et al. (2010) proposed a parallel variable neighborhood search (VNS) algorithm. They generated a large number of solutions by the AL-based routing and random sequencing. The solution with the minimal makespan then initiated the parallel VNS, which consisted of nine neighborhood functions. Defersha and Chen (2010) developed a parallel GA to minimize the makespan in a complex flexible job shop, which includes sequence dependent setup times, machine release dates, and time lag requirements. Their GA is based on Pezzella et al., GA. The parallelism was achieved by the island model and random connection topology. The approach was realized by MPI (Quinn, 2003) in an environment with 48 processors.

Besides minimization of the makespan, Kacem et al. (2002a) considered minimization of total workload and maximum workload in solving the MOFJSP. They proposed the above-mentioned AL, which became a popular approach to make the routing decisions for the initial solutions. They generated a set of five problem instances in another study (Kacem et al., 2002b). These instances became a standard set of benchmark instances and initiated the growth of research studies on MOFJSP. As mentioned earlier, one type of approach to deal with multiobjective optimization is the aggregation-based approach. Xia and Wu (2005) adopted this type of approach and aggregated the three objective values by linear weighted summation. They combined particle swarm optimization (PSO) and simulated annealing (SA) to a two-level algorithm. The PSO searched for good routing decisions, and the SA searched for good sequencing decisions. Gao et al. (2007) also took linear weighted sum of three objective values as the fitness function in their hybrid GA, which integrated a local search procedure. Two kinds of neighborhood functions were used in the local search procedure. The first one was proposed by Nowicki and Smutnicki (1996) and generated solutions by swapping particular critical operations. The second one was proposed by Gao et al. and generated solutions by assigning a different machine to critical operations. Tay and Ho (2008) addressed an MOFJSP regarding some different objectives (makespan, mean tardiness, and mean flow time). They aggregated these objective values by equal-weight linear summation. Genetic programming (GP) was applied to seek for good dispatching rules for operation sequencing. Routing decisions were made by a simple heuristic, assigning operations to the machine with the shortest waiting time. Zhang et al. (2009), Xing et al. (2009a), and Li et al. (2010) all used linear weighted sum as the aggregation function. Zhang et al. (2009) hybridized PSO and TS. The PSO adopted a two-string encoding and dealt with routing and sequencing simultaneously. The TS used the neighborhood function by Mastrolilli and Gambardella (2000) and focused on sequencing only. Xing et al. (2009a) integrated ACO and local search. The local search was invoked when a new best solution was found by the ACO. Neighboring solutions were produced by re-assigning each operation to the machines with the shortest or second shortest processing time or the shortest total processing time and then sequencing the operations by one of five dispatching rules. Li et al. (2010) applied a TS for routing and a hill climbing procedure for sequencing. The TS had two neighborhood functions, one changing the machine for an operation randomly and the other moving an operation from the machine with the most critical operations to a machine with fewer critical operations. The hill climbing procedure had three neighborhood functions, which swapped or re-inserted head or tail operations of critical blocks.

Download English Version:

https://daneshyari.com/en/article/5080715

Download Persian Version:

https://daneshyari.com/article/5080715

Daneshyari.com