Expert Systems With Applications 42 (2015) 9434–9440

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Effective hierarchical optimization by a hierarchical multi-space competitive genetic algorithm for the flexible job-shop scheduling problem

Shudai Ishikawa [a,*], Ryosuke Kubota [b], Keiichi Horio [a]

[a] Graduate School of Life Science and Systems Engineering, Kyushu Institute of Technology, 2-4 Hibikino, Wakamatsu-ku, Kitakyushu, Fukuoka Pref., Japan
[b] Department of Intelligent Systems Engineering, Ube National College of Technology, Tokiwadai, Ube City, Yamaguchi Pref., Japan

ARTICLE INFO

Keywords:
Multiple solution space
Solution space competition
Genetic algorithm
Hierarchical optimization
Flexible job-shop scheduling problem

ABSTRACT

In this paper, we propose a new optimization technique, the hierarchical multi-space competitive distributed genetic algorithm (HmcDGA), which is effective for the hierarchical optimization problem. It is an extension of the multi-space competitive distributed genetic algorithm (mcDGA), which was proposed by the authors. The mcDGA efficiently finds an optimal solution with a low computational cost by increasing the number of individuals in a solution space in which it is likely to exist. An optimization method that is divided into several levels of hierarchy is called a *hierarchical optimization*. Several hierarchical optimization techniques have been proposed, including the hierarchical genetic algorithm (HGA). In hierarchical optimization, a complex problem is divided into a hierarchical collection of simpler problems, and each level is optimized independently. In this way, complex problems can be solved without the need to develop problem-specific operators. However, in the conventional HGA, this results in a high computational cost because the genetic algorithm (GA) is repeated many times at upper and lower level. The HmcDGA is a hybrid of the mcDGA and HGA, and it has some of the advantages of each one; for example, the HmcDGA can find an optimal solution at low computational cost and without requiring special operations. This allows it to be applied to a wide variety of optimization problems. Therefore, the HmcDGA may become the powerful optimization algorithm that can solve various problems. In this paper, we apply the proposed HmcDGA to the flexible job-shop scheduling problem (FJSP) which is one of the complex combinational optimization problem and confirm its effectiveness. Simulation results show that the HmcDGA can find solutions that are comparable to those found by using GAs developed specifically for the FJSP, the HmcDGA is not required a lot of computational costs comparing to the HGA.

1. Introduction

Scheduling problems are well known and important, and they appear in various arenas. One example of this is the job-shop scheduling problem (JSP), which is one of the hardest combinatorial optimization problems (Garey, Johnson, & Sethi, 1976) in the field of production scheduling. In the JSP, a set of jobs are performed by a set of machines. Each job is formed by a sequence of consecutive operations, and each operation requires exactly one machine. Each machine is continuously available, and it can process one operation at a time without interruption. The problem is to find the sequence of operations on the machines that will minimize the total time required to complete the set of jobs; this is known as the *makespan*.

The flexible job-shop scheduling problem (FJSP) (Brucker & Schlie, 1990) is an extension of the JSP in which each operation may be processed by any of the machines in the set. The JSP is well known to be NP-hard (Garey et al., 1976), therefore FJSP is also NP-hard. Compared to the JSP, with the FJSP, it is necessary to make more machine assignments and to determine an appropriate processing sequence. That is, the FJSP can be decomposed into two sub-problems which consisted of an optimization of machine assignments and processing sequences. Consequently, FJSP has huge solution space, and an evaluation of the machine assignments is determined by the processing sequences, thereby it is difficult to solve this problem.

In recent years, there has been a lot of interest in using a meta-heuristic approach to solving scheduling problems; these approaches include the genetic algorithm (GA) (Holland, 1962), particle swarm optimization (PSO) (Kennedy, 2010), and tabu search (TS) (Glover, 1986). In particular, the GA has been successfully used to solve scheduling problems, including the FJSP and the vehicle routing problem (VRP) (M.Baker & Ayechew, 2003). Chen and colleagues

* Corresponding author. Tel.: +819068371172.
E-mail addresses: ishikawa-shudai@edu.brain.kyutech.ac.jp (S. Ishikawa), kubota@ube-k.ac.jp (R. Kubota), horio@brain.kyutech.ac.jp (K. Horio).

http://dx.doi.org/10.1016/j.eswa.2015.08.003
0957-4174/

developed the GA for the FJSP. They split the chromosome of an individual into two parts; the first part was used to assign the machine to be used, and the second part was used to determine the sequence of operations to be performed by each machine (Chen, Ihlow, & Lehmann, 1999). Pezzella and colleagues proposed an algorithm that integrates different strategies for generating the initial population, selecting the individuals that will reproduce, and producing new individuals (Pezzella, Morganti, & Ciaschetti, 2008). Ida and Oka modified the methods for initialization, selection, and mutation, and they also proposed a new operator called the *escaping method* (Ida & Oka, 2009). Zhang and colleagues developed a new way of generating a high-quality initial population; their proposed operators include global selection (GS) and local selection (LS) (Zhang, Gao, & Shi, 2011). Teekeng and Arit proposed a modified GA with the following features: 1) an effective selection method called *fuzzy roulette wheel selection*, 2) a new crossover operator that applies hierarchical clustering to each generation, and 3) a new mutation operator that helps to maintain population diversity and to overcome premature convergence (Teekeng & Thammano, 2012). In the methods described above, an individual is represented by a single string that includes both the machine assignment and the processing sequence, and these are simultaneously updated by a genetic operator. Although each processing sequence is evaluated as an individual, each machine is assigned in many processing sequences. When each individual is represented by a string, it is necessary to search a huge solution space, and a large population with many generations is required in order to find a good solution. Moreover, in these methods, some genetic operators are specialized for the FJSP, and thus they may not be effective for other problems. For example, the multiple vehicle routing problem (mVRP) (Laporte, 1992) is in many ways similar to the FJSP, and when the GA is applied to the FJSP, individuals are represented by the allocation of customers to each vehicle and by the route of each vehicle. However, this problem is subject to various constraints (*e.g.,* those on the time window, capacity, or number of depots), and the individual with the shortest route is not always the optimal solution. That is, using initialization or selection to minimize the vehicle route is not always effective. In addition, complex processing may cause an increase in the computational cost and make implementation difficult.

In hierarchical optimization, a complex system is divided hierarchically into several simple systems, and each level is then optimized. A problem that is handled in this way is called a *hierarchical problem* (Simon, 1968). Many systems have hierarchical structure; for example, a product is composed of various modules and parts, and the Internet is a hierarchical arrangement of roots and domains. The concept of hierarchical optimization is not new, and many complex systems have hierarchical structure; thus, genetic and evolutionary algorithms have been proposed for optimizing random modularity problems that can be decomposed hierarchically (Janson & Middendorf, 2005; Young-Jou, 1996). The hierarchies considered in these studies have a tree structure., which means that there are relatively few variables at the highest level. However, if a hierarchical problem is decomposed into to the search space and a solution in that search space, there will be many variables at the highest level; thus it is necessary to optimize not only the lower levels but also the higher levels. For example, mVRP is a scheduling problem that can be considered as a hierarchical problem. The allocation of customers for each vehicle is determined at a higher level, and the search for the route is at a lower level. In this problem, the search space is determined by the combination of customers that are routed by each vehicle, and thus the number of possible combinations increases enormously with the number of customers. If a traditional optimization method is applied to each level, the computation cost is prohibitive. In addition, if no optimal solution is found for one or more of the lower levels, it will be difficult to find an optimal solution for the entire system. The hierarchical genetic algorithm (HGA) has been proposed as a general framework for solving hierarchical problems (de Jong, Thierens, & Watson, 2004). However, the main aim of the initial paper (de Jong et al., 2004) was to investigate under what conditions hierarchical problems can be addressed efficiently, and it did not consider the application of HGA to complex hierarchical problems. In recent studies, there are several application examples and interpretations about HGA. For example, Zhao applied HGA to simultaneous optimization of the PbLaZrTi-based actuator configuration and corresponding applied light intensity for morphing beam structural shapes (Zhao, Zheng, Wang, & Yang, 2015). In this paper, two chromosomes which consist of some control genes and parametric genes are used, and these are composed hierarchically. As another example, HGA is applied to the optimization of the design parameter of a modular neural networks (Sanchez & Melin, 2014) and the automatically extraction of fuzzy parameters of a Fuzzy Logic Controller (Mendes, Araujo, Matias, Seco, & Belchior, 2014). Thus, HGA is applied to several fields. However, in these studies, HGA is just used in order to simplify the problem, and an increase in computational cost due to that the problem is composed hierarchically is not considered.

On the other hand, our proposed multi-space competitive distributed genetic algorithm (mcDGA) (Ishikawa et al., 2011) is effective for problems that have multiple solution spaces. In the mcDGA, a hierarchical structure is created, and then the search for an optimal solution and the selection of the solution space are performed simultaneously. We note that the upper and lower levels correspond to the selection of a solution space and the search for a solution, respectively. In the mcDGA, the evaluation values of each of the solution spaces are compared, and the population is increased for the solution space that is most likely to contain an optimal solution. Comparison of the evaluation values between solution spaces is called *competition*, and the adjustment of the population size is called *migration*. By using competition and migration, there is a relatively low computational cost for finding an optimal solution. However, in the mcDGA, it is necessary that the solution space be small, and thus the mcDGA cannot be applied to a problem (such as the mVRP) that has a huge solution space.

In this paper, we propose the hierarchical multi-space competitive distributed genetic algorithm (HmcDGA) as a new hierarchical optimization technique that is based on the HGA; we then apply it to the FJSP. The HmcDGA is an extension of the mcDGA; it not only optimizes the individuals, but it also optimizes the solution space. The proposed method is expected to reduce the computational cost and to improve the rate of determining the optimal solution at a lower level (Ishikawa, Kubota, & Horio, 2014). Moreover, the HmcDGA has a general genetic operator and thus does not require special processing for specific problems, since the initial problem is divided into a set of simpler problems. That is, HmcDGA is not likely to depend on the nature of the problem and can thus be applied to many types of optimization problems.

The paper is organized as follows. Section 2 describes the FJSP. Section 3 describes the concept underlying the mcDGA. Section 4 presents the proposed HmcDGA using the FJSP and evaluates the performance of the HmcDGA. Section 5 presents the result of an experiment in which the HmcDGA is applied to the FJSP. Section 6 discusses our conclusions and some areas of future work.

2. FJSP and GA

2.1. FJSP

The FJSP can be formulated as follows. Consider a set of l jobs $J = \{J_1, J_2, \ldots, J_i, \ldots J_l\}$. A job J_i is formed by a sequence of operations $O_{i1}, O_{i2}, \ldots, O_{ij}, \ldots, O_{im_i}$ that are to be performed. Let n be the number of total machines, and let the set of machines be $M = \{M_1, M_2, \ldots, M_k, \ldots, M_n\}$. Each operation O_{ij} requires a machine from the set of available machines. Namely, the FJSP is to determine a way