Applied Soft Computing 10 (2010) 888–896

Contents lists available at ScienceDirect

Applied Soft Computing

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

A Knowledge-Based Ant Colony Optimization for Flexible Job Shop Scheduling Problems

Li-Ning Xing*, Ying-Wu Chen, Peng Wang, Qing-Song Zhao, Jian Xiong

Department of Management Science and Engineering, College of Information System and Management, National University of Defense Technology, Changsha 410073, China

ARTICLE INFO

Article history:
Received 22 April 2008
Received in revised form
12 September 2009
Accepted 18 October 2009
Available online 24 October 2009

Keywords:
Combinatorial optimization
Ant Colony Optimization
Flexible Job Shop Scheduling

ABSTRACT

A Knowledge-Based Ant Colony Optimization (KBACO) algorithm is proposed in this paper for the Flexible Job Shop Scheduling Problem (FJSSP). KBACO algorithm provides an effective integration between Ant Colony Optimization (ACO) model and knowledge model. In the KBACO algorithm, knowledge model learns some available knowledge from the optimization of ACO, and then applies the existing knowledge to guide the current heuristic searching. The performance of KBACO was evaluated by a large range of benchmark instances taken from literature and some generated by ourselves. Final experimental results indicate that the proposed KBACO algorithm outperforms some current approaches in the quality of schedules.

© 2009 Elsevier B.V. All rights reserved.

1. Introduction

Scheduling problems have a vital role in recent years due to the growing consumer demand for variety, reduced product life cycles, changing markets with global competition and rapid development of new technologies. The Job Shop Scheduling Problem (JSSP) is one of the most popular scheduling models existing in practice, which is among the hardest combinatorial optimization problems [1]. Many approaches, such as, Simulated Annealing (SA) [2], Tabu Search (TS) [3], Genetic Algorithm (GA) [4], Ant Colony Optimization (ACO) [5], Neural Network (NN) [6], Evolutionary Algorithm (EA) [7] and other heuristic approach [8–10], have been successfully applied to JSSP.

In order to match today's market requirements, manufacturing systems need not only automated and flexible machines, but also flexible scheduling systems. The Flexible Job Shop Scheduling Problem extends JSSP by assuming that, for each given operation, it can be processed by any machine from a given set. Bruker and Schlie [11] were among the first to address this problem. The difficulties of FJSSP can be summarized as follows.

(1) Assignment of an operation to an appropriate machine;
(2) Sequencing the operations on each machine;
(3) A job can visit a machine more than once (called recirculation).

These three features significantly increase the complexity of finding even approximately optimal solutions.

Although EA has been applied to solve numerous applications, but there have two disadvantages for solving combinational optimization problems by a canonical EA.

(1) A canonical EA is a 'generation-evaluation' type of searching technique, which only uses fitness value or objective value to guide the evolutionary search [12].
(2) The optimization results obtained by the canonical EA are still limited due to the reliance on randomized natural selection and recombination [13].

For improving the performance of EA, several researches integrated some optimization strategies into the EA [14,15]. Also, the study of interaction between evolution and learning for solving optimization problems has been attracting much attention [16–19]. The diversity of these approaches has motivated our pursuit for a uniform framework called Knowledge-Based Heuristic Searching Architecture (KBHSA), which integrates knowledge model and heuristic searching model to search an optimal solution. We demonstrate the performance of this architecture in the instantiation of the Knowledge-Based Ant Colony Optimization (KBACO) which is applied to common benchmark problems. Experimental results show that KBACO algorithm outperforms previous approaches for solving the FJSSP.

The remainder of this paper is organized as follows. Section 2 reviews some recent works related to the FJSSP. Section 3 describes the proposed architecture (KBHSA) and its instantiation (KBACO).

* Corresponding author.
E-mail addresses: xln_2002@nudt.edu.cn, xinglining@gmail.com (L.-N. Xing), ywchen@nudt.edu.cn (Y.-W. Chen), phd2999@gmail.com (P. Wang), zqszqr@163.com (Q.-S. Zhao), xiongjian1984@hotmail.com (J. Xiong).

1568-4946/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2009.10.006

Section 4 presents and analyzes the performance of KBACO when applied to solve common benchmarks in literature and others generated by ourselves. Finally, Section 5 gives some concluding remarks and directions for future work.

2. Related works

There are three parts in this section. Section 2.1 gives the formal definition of FJSSP. Also, a practical model of the FJSSP is also described. Section 2.2 reviews recent related works for solving the FJSSP. Current studies of the interaction between evolution and learning for solving optimization problems are discussed in Section 2.3.

2.1. Problem formulation

Generally, the FJSSP can be formulated as follows [20].

(1) There is a set of n jobs that plan to process on m machines;
(2) Let $J = \{J_i\}_{1 \le i \le n}$, indexed i, be a set of n jobs;
(3) Let $M = \{M_k\}_{1 \le k \le m}$, indexed k, be a set of m machines;
(4) Each job J_i consists of a predetermined sequence of operations;
(5) Each operation O_{ij} (operation j of job i) can be processed without interruption on one of a set of machines $M_{ij}(M_{ij} \subseteq M)$. We denote p_{ijk} to be processing time of O_{ij} on machine M_k.
(6) The objective of FJSSP is to find a minimum makespan.

Hypotheses considered in this paper are listed as follows [21].

(1) Setting up times of machines and move times between operations are negligible;
(2) Machines are independent from each other;
(3) Jobs are independent from each other;
(4) At a given time, a machine can execute at most one operation. It becomes available to other operations only if the operation which is processing is completed;
(5) There are no precedence constraints among the operations of different jobs;
(6) No more than one operation of the same job can be executed at a time.

Kacem et al. [22] classified the FJSSP into two sub-problems as follows.

(1) Total FJSSP (T-FJSSP): each operation can be processed on any machine;
(2) Partial FJSSP (P-FJSSP): each operation can be processed on one machine of subset of M.

An instance of a FJSSP with two-job and three-machine is presented in Fig. 1. The operation sequence, machine alternatives and processing times are displayed in Table 1. The Gant chart of a feasible solution is presented in Fig. 1. The makespan for this solution is 53. This solution is not the optimal solution. The optimal solution must be found by the algorithm.

2.2. Previous works

Hierarchical approaches and integrated approaches are two types of approaches, which have been used to solve the realistic instance of FJSSP [21].

(1) In hierarchical approaches, assignment of operations to machines and the sequencing of operations on the resources or machines are treated separately. Hierarchical approaches are based on the idea of decomposing the original problem in order to reduce its complexity. Brandimarte [23] was the first to use the decomposition for the FJSSP. He solved the routing sub-problem using some existing dispatching rules and then focused on the scheduling sub-problem, which is solved using a TS heuristic. Tung et al. [24] developed a similar approach for scheduling a flexible manufacturing system. Kacem et al. [25] proposed a GA controlled by the assigned model which is generated by the approach of localization. Xia and Wu [21] present an effective hybrid optimization approach for the multi-objective FJSSP. They make use of Particle Swarm Optimization (PSO) to assign operations on machines and SA algorithm to schedule operations.
(2) Integrated approaches were used by considering assignment and scheduling at the same time. Hurink et al. [26] proposed a TS heuristic in which reassignment and rescheduling are considered as two different types of moves. Dauzere-Peres and Paulli presented a TS procedure based on their proposed neighborhood structure for the FJSSP [27]. In their approach, there is no distinction between reassigning and resequencing an operation. Mastrolilli and Gambardella [28] improved Dauzere-Peres' TS techniques and presented two neighborhood functions.

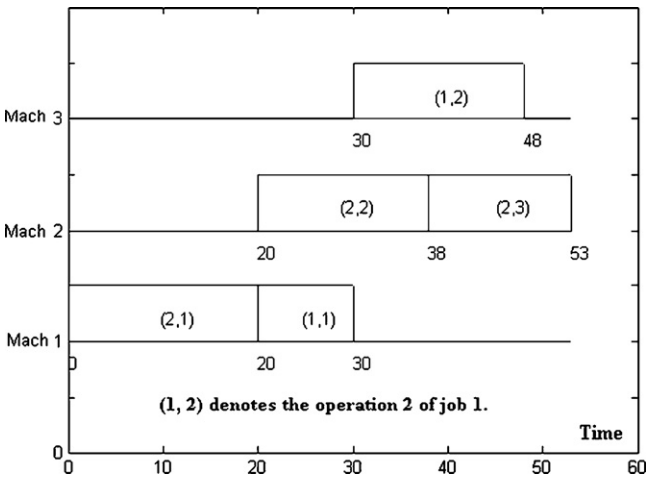

Fig. 1. The Gant chart of a feasible solution.

2.3. Learning for evolution

Generally, there are two approaches to implement the interaction between evolution and learning [20]. The first one is to keep good features of previous individuals to improve the fitness of individuals in the current generation. The second one is to keep bad features of previous individuals to avoid infeasibility in the current generation. Reynolds [16] implemented the interaction between

Table 1
Example of a two-job, three-machine FJSSP.

Jobs	Operation sequence	Operations	Machine alternative	Processing time (s)
J_1	O_{11}, O_{12}	O_{11}	M_1	10
			M_2	15
		O_{12}	M_2	12
			M_3	18
J_2	O_{21}, O_{22}, O_{23}	O_{21}	M_1	20
			M_3	25
		O_{22}	M_1	25
			M_2	18
		O_{23}	M_2	15
			M_3	25

دریافت فوری ← متن کامل مقاله

ISIArticles
مرجع مقالات تخصصی ایران

✓ امکان دانلود نسخه تمام متن مقالات انگلیسی
✓ امکان دانلود نسخه ترجمه شده مقالات
✓ پذیرش سفارش ترجمه تخصصی
✓ امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
✓ امکان دانلود رایگان ۲ صفحه اول هر مقاله
✓ امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
✓ دانلود فوری مقاله پس از پرداخت آنلاین
✓ پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات