Computers & Industrial Engineering 85 (2015) 376–383

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

A local search genetic algorithm for the job shop scheduling problem with intelligent agents ☆

Leila Asadzadeh *

Department of Computer Engineering and Information Technology, Payame Noor University, Iran

A R T I C L E I N F O

Article history:
Received 15 March 2013
Received in revised form 24 September 2013
Accepted 6 April 2015
Available online 11 April 2015

Keywords:
Job shop scheduling problem
Genetic algorithms
Local search
Intelligent agents
Multi agent systems

A B S T R A C T

The job shop scheduling problem is one of the most important and complicated problems in machine scheduling and is considered to be a member of a large class of intractable numerical problems known as NP-hard. Genetic algorithms have been implemented successfully in many scheduling problems, in particular job shop scheduling. Hybridization is an effective way of improving the performance and effectiveness of genetic algorithms. Local search techniques are the most common form of hybridization that can be used to enhance the performance of these algorithms. Agent-based systems technology has generated lots of excitement in recent years because of its promise as a new paradigm for conceptualizing, designing, and implementing software systems. This paper presents an agent-based local search genetic algorithm for solving the job shop scheduling problem. A multi agent system containing various agents each with special behaviors is developed to implement the local search genetic algorithm. Benchmark instances are used to investigate the performance of the proposed approach. The results show that the proposed agent-based local search genetic algorithm improves the efficiency.

1. Introduction

Job shop scheduling problem (JSSP) is one of the most important problems in machine scheduling. Due to factorial explosion of possible solutions, job shop scheduling problems are considered to be a member of a large class of intractable numerical problems known as NP-hard (Jain & Meeran, 1999).

Historically JSSP has been primarily treated using the branch and bound (Carlier & Pinson, 1989), heuristic rules (Kannan & Ghosh, 1993) and shifting bottleneck procedure (Adams, Balas, & Zawack, 1988).

In recent years, meta-heuristic approaches such as taboo search (TS) (Nowicki & Smutnicki, 1996), simulated annealing (SA) (Laarhoven, Aarts, & Lenstra, 1992), genetic algorithm (GA) (Goncalves, Mendes, & Resende, 2005; Wang & Zheng, 2001, 2002), neural networks (NN) (Foo, Takefuji, & Szu, 1995), ant colony optimization (ACO) (Huang & Liao, 2008), particle swarm optimization (PSO) (Ge, Du, & Qian, 2007), artificial bee colony algorithm (ABC) (Banharnsakun, Sirinaovakul, & Achalakul, 2012) and bee colony optimization (BCO) (Chong & Low, 2006) are widely applied to solve this problem.

Genetic algorithms have been implemented successfully in many scheduling problems, in particular job shop scheduling. For more complicated problems a genetic algorithm needs to couple with problem-specific methods in order to make the approach really effective. Traditional heuristic methods are incorporated to enhance the performance of genetic search. Ombuki and Ventresca (2004) proposed a local search genetic algorithm that uses an efficient solution representation strategy in which both checking of the constraints and repair mechanism can be avoided. In their approach, at local search phase a new mutation-like operator is used to improve the solution quality. Lin, Goodman, and Punch (1995) introduced a hybrid model consisting of coarse-grain genetic algorithms connected in a fine-grain style topology. Their method can avoid premature convergence, and it produced excellent results on standard benchmark job shop scheduling problems. Wang and Zheng (2001) by combining simulated annealing and genetic algorithms developed a general, parallel and easily implemented hybrid optimization framework, and applied it to job shop scheduling problem. Based on effective encoding scheme and some specific optimization operators, some benchmark job shop scheduling problems are well solved by the hybrid optimization strategy. Watanabe, Ida, and Gen (2005) introduced a genetic algorithm with a modified crossover operator and a search area

☆ This manuscript was processed by Area Editor Subhash C. Sarin.
* Tel.: +98 9143633494; fax: +98 4434242899.
E-mail address: leila_asadzadeh_cs@yahoo.com

http://dx.doi.org/10.1016/j.cie.2015.04.006
0360-8352/© 2015 Published by Elsevier Ltd.

adaptation method for controlling the tradeoff balance between global and local searches. Asadzadeh and Zamanifar (2010) proposed an agent-based parallel genetic algorithm approach. This parallel approach is based on a coarse-grained model. The initial population is divided into sub-populations, and each sub-population is evolved separately. Communication between sub-populations is restricted to the migration of chromosomes. Qing-dao-er-ji and Wang (2012) proposed a new hybrid genetic algorithm for job shop scheduling problem. In their paper, some genetic operators were designed. In order to increase the diversity of the population, a mixed selection operator based on the fitness value and the concentration value was given and a local search operator was designed, which can improve the local search ability of genetic algorithm greatly. Yusof, Khalid, Hui, Yusof, and Othman (2011) solved the job shop scheduling problem by using a hybrid parallel micro genetic algorithm. A new hybrid parallel genetic algorithm (PGA) based on a combination of asynchronous colony genetic algorithm (ACGA) and autonomous immigration genetic algorithm (AIGA) is employed to solve benchmark job shop scheduling problem.

In this paper, an agent-based local search genetic algorithm is proposed for solving the job shop scheduling problem. A multi agent system containing various agents each with special behaviors is developed to implement the local search genetic algorithm. In this approach, two local search procedures are applied to enhance the efficiency of the genetic algorithm.

The reminder of this paper is organized as follow. In Section 2, we describe the job shop scheduling problem. In Section 3, our proposed agent-based local search genetic algorithm is presented. Results and discussion are described in Section 4. Our conclusion is given in Section 5.

2. Job shop scheduling problem

Job shop scheduling problem can be described as a set of n jobs J_i where $i = 1,2,\dots,n$ which have to be processed on a set of m machines M_k where $k = 1,2,\dots,m$. Each job consists of a chain of operations, each of which needs to be processed during an uninterrupted time period of a given length on a given machine. Operation of ith job on the kth machine will be denoted by O_{ik}. There are several constraints on jobs and machines:

- A job does not visit the same machine twice.
- There are no precedence constraints among the operations of different jobs.
- Each machine can process only one job at a time.
- Each job can be processed by only one machine at a time
- Neither release times nor due dates are specified.

Usually we denote the general job shop scheduling problem as $n \times m$, where n is the number of jobs and m is the number of machines. The duration in which all operations for all jobs are completed is referred to as the makespan. A schedule determines the execution sequence of all operations for all jobs on machines. The objective is to minimize the makespan value. An example of a 4×4 JSSP is presented in Table 1.

Table 1
An example of a 4×4 JSSP.

Job	Machine, processing time			
J_1	1,3	2,3	3,3	4,2
J_2	1,2	4,5	2,4	3,3
J_3	2,3	1,2	4,4	3,1
J_4	4,3	2,8	1,5	3,4

3. Local search genetic algorithm for JSSP with intelligent agents

Genetic algorithms have been implemented successfully in many combinatorial problems. For more complicated problems a genetic algorithm needs to be integrated with problem-specific methods in order to make the approach really effective. Hybridization can be an extremely effective way of improving the performance and effectiveness of genetic algorithms. The most common form of hybridization is to couple genetic algorithms with local search techniques and to incorporate domain-specific knowledge into the search process. A common form of hybridization is to incorporate a local search operator into the genetic algorithm by applying the operator to each member of the population after each generation. This hybridization is often carried out in order to produce stronger results than the individual approaches can achieve on their own (Sastry, Goldberg, & Kendall, 2005).

Agent-based systems technology has generated lots of excitement in recent years because of its promise as a new paradigm for conceptualizing, designing, and implementing software systems. A software agent is a computer system situated in an environment that acts on behalf of its user and is characterized by a number of properties. A multi agent system is one composed of multiple interacting software agents, which are typically capable of cooperating to solve problems.

In this paper, we propose a hybrid genetic algorithm that combines local search heuristics with crossover operators. To implement our local search genetic algorithm, a multi agent system containing some intelligent agents is developed. Agents of multi agent system have special actions that are used to implement the genetic algorithm and local search procedures.

In this section the proposed agent-based model for JSSP is introduced and its structure and details are described. The architecture of agent-based model and behavior of agents are introduced in Section 3.1. In Section 3.2, local search procedures are explained. Genetic operators are described in Section 3.3.

3.1. Architecture of agent-based model

The architecture of proposed agent-based model for JSSP is shown in Fig. 1. To implement the agent-based model, we used JADE (Bellifemine, Poggi, & Rimassa, 2001) as a platform and built our agents on it. The model contains various agents that communicate over the context that provided by JADE middleware. JADE is compliant with FIPA standard specifications so that agents developed on it can thus interoperate with other agents built with the same standard. JADE allows each agent to dynamically discover other agents and to communicate with them in a peer-to-peer manner.

Each running instance of the JADE runtime environment is called a container as it can contain several agents. The set of active containers is called a platform. A single special main container must always be active in a platform and all other containers register with it as soon as they start. Once the platform is activated, the JADE default agents in the main container, including AMS (Agent Management System), DF (Directory Facilitator), and RMA (Remote Monitoring Agent) are instantiated. The AMS agent exerts supervisory control over access to and use of the agent platform; the DF agent provides the default yellow page service in the platform; and the RMA allows controlling the life cycle of the agent platform and of all the registered agents. We have two containers in our platform that agents are executed on them: *Main-container* and *Container-1*. Each agent of the proposed model has been developed for a special purpose. The agents can be described as follows.