Computers & Industrial Engineering 110 (2017) 75–82

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

A reinforcement learning approach to parameter estimation in dynamic job shop scheduling

Jamal Shahrabi [a,*], Mohammad Amin Adibi [b], Masoud Mahootchi [a]

[a] Faculty of Industrial Engineering and Management Systems, Amirkabir University of Technology, Tehran, Iran
[b] Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran

ARTICLE INFO

Article history:
Received 24 May 2013
Received in revised form 27 June 2016
Accepted 24 May 2017
Available online 26 May 2017

Keywords:
Reinforcement learning
Q–factor
Dynamic job shop scheduling
Variable neighborhood search

ABSTRACT

In this paper, reinforcement learning (RL) with a Q-factor algorithm is used to enhance performance of the scheduling method proposed for dynamic job shop scheduling (DJSS) problem which considers random job arrivals and machine breakdowns. In fact, parameters of an optimization process at any rescheduling point are selected by continually improving policy which comes from RL. The scheduling method is based on variable neighborhood search (VNS) which is introduced to address the DJSS problem. A new approach is also introduced to calculate reward values in learning processes based on quality of selected parameters. The proposed method is compared with general variable neighborhood search and some common dispatching rules that have been widely used in the literature for the DJSS problem. Results illustrate the high performance of the proposed method in a simulated environment.

1. Introduction

Selecting appropriate scheduling method or optimization parameters in the dynamic job shop scheduling (DJSS) has been noted by many researchers in recent years (Liu & Hsu, 2015; Nguyen, Zhang, Johnston, & Tan, 2015; Park et al., 2016; Scholz-Reiter, Hildebrandt, & Tan, 2013), because dynamic environment of problem due to changes in shop floor condition, causes a scheduling method not to be the best all over the scheduling horizon.

The DJSS problem emerges when real time events such as random job arrivals and machine breakdowns are taken into account in the ordinary JSS problem. In the DJSS problem, one or more conditions of the job shop scheduling (JSS) problem like the number of jobs or the number of operable machines are changed by a random event. In addition to the scheduling problem of DJSS, it is necessary to address two new issues; "when" and "how" to react to dynamic events in DJSS problems.

In some researches published regarding the DJSS problem, a scheduling method with predefined parameters is used at any rescheduling point (Chryssolouris & Subramanian, 2001; Dominic, Kaliyamoorthy, & Saravana, 2004; Fatemi Ghomi & Iranpoor, 2010; Liu, Ong, & Ng, 2005; Qi, Burns, & Harrison, 2000; Wang, Xiao, Li, & Wang, 2010; Yang, Yu-si, & Hong-yn, 2009; Zhou, Nee, & Lee, 2009). However, due to changes in problem conditions during the planning horizon, using a scheduling method with fixed parameters can reduce the performance of the method. Preventing such a problem, Aydin and Oztemel (2000) used reinforcement learning agents to select appropriate dispatching rules for scheduling according to the shop floor conditions in real time. Wang and Usher (2005) used reinforcement learning agents to select an appropriate dispatching rule for a single machine dynamic scheduling problem.

Although their work is a successful use of RL in DJSS, dispatching rules do not have good quality to meet optimization objectives. Sha and Liu (2005) presented a model that incorporates a data mining tool for mining the knowledge of job scheduling regarding due date assignment in a dynamic job shop environment to adjust an appropriate parameter according to the condition of the shop floor at the instant of job arrival.

Adibi, Zandieh, and Amiri (2010) used a trained artificial neural network (ANN) to update parameters of a metaheuristic method at any rescheduling point in a DJSS problem according to the problem condition. In their proposed method, offline training was used and no adaptation was taken during the scheduling horizon. Chen, Hao, Lin, and Murata (2010) proposed a rule driven dispatching method based on data envelopment analysis and reinforcement learning for the multi objective dynamic scheduling problem. Vinod and Sridharan (2011) studied the effects of due date assignment methods and scheduling rules on the performance of a job shop produc-

* Corresponding author at: 424 Hafez Ave., Tehran 1591634311, Iran.
E-mail addresses: jamalshahrabi@aut.ac.ir (J. Shahrabi), adibi@qiau.ac.ir (M.A. Adibi), mmahootchi@gmail.com (M. Mahootchi).

http://dx.doi.org/10.1016/j.cie.2017.05.026
0360-8352/

tion system in a dynamic environment. Kapanoglu and Alikalfa (2011) introduced an unsupervised learning scheduling system that builds state and priority rule pairs depending on intervals of queue by using a rule based GA for the DJSS problem. Adibi and Shahrabi (2014) presented a clustering-based modified variable neighborhood search algorithm for DJSS problem. They used K-means clustering method to improve local search in the meta-heuristic method. Oktaviandri, Hassan, and Shaharoun (2016) developed a decision support tool based on promising artificial intelligent that was able to use appropriately dispatching rules for DJSS problem.

Obviously, reinforcement learning techniques in which agents have to take into account reinforcement signals against their actions, are widely employed to address DJSS problem in which learning is required to its solve methods. However, it has been used only to select appropriate scheduling method.

On the other hand, Q-learning is the most well-known reinforcement learning algorithm which is used in literature. It works in such a way that the agent gains experience by trial and error throughout the execution of the actions. During this process, the agent learns how to assign credit or blame to each of its actions in order to improve its behavior (Aydin & Oztemel, 2000).

In this study, variable neighborhood search (VNS) (Mladenovic & Hansen, 1997) is selected as a scheduling method at any rescheduling point because it brings together a lot of desirable properties for a metaheuristic such as simplicity, efficiency, effectiveness, and generality (Hansen, Mladenovic, & Moreno Perez, 2007). To enhance the efficiency and effectiveness of VNS, its parameters are updated at any rescheduling point by RL. Using few effective parameters during the optimization process, VNS is a more suitable scheduling method because using a scheduling method with fewer parameters decreases attempts to train agents in the learning process in the new so called RLVNS approach. In fact, the proposed method for DJSS problem or RLVNS uses RL to determine value of effective parameters of the VNS according to problem situation dynamically. This determination causes an automatic coordination between problem situation and the VNS setting over time. The proposed method is tested in a challenging simulated environment and it is compared to some benchmark methods reported in the literature for DJSS problem.

The rest of the paper is organized as follows: In Section 2 the dynamic job shop scheduling problem is defined in detail and then VNS is explained in Section 3. Reinforcement learning and Q-factor algorithm is presented in Section 4. A framework for the proposed method is introduced in Section 5. Simulation results are presented in Section 6 and the conclusion is given in Section 7.

2. Dynamic job shop scheduling problem

In the general job shop scheduling problem, n jobs should be processed on m machines while minimizing a function of completion time of jobs is considered along with the following technological constraints and assumptions:

(1) each machine can perform only one operation at a time on any job,
(2) an operation of a job can be performed by only one machine at a time,
(3) once an operation has begun on a machine it must not be interrupted,
(4) an operation of a job cannot be performed until its preceding operations are completed,

Initialization: Select the set of neighborhood structures $N_k\,(k = 1,2...,k_{\max})$, which will

be used in the search; find an initial solution x; choose a stopping

condition;

Repeat the following until the stopping condition ($N =$ number of iteration) is met:

(1) Set $k \leftarrow 1$;

(2) Until $k = k_{\max}$, repeat the following steps:

(a) *Shaking*. Generate a point x' at random from the k^{th} neighborhood of x

($x' \in N_k(x)$);

(b) *Local search*. Apply some local search method with x' as initial

solution; denote with x'' the so obtained local optimum;

(c) More or not. If this local optimum is better than the incumbent, move

there ($x \leftarrow x''$), and continue the search with $N_1\,(k \leftarrow 1)$; otherwise,

set $k \leftarrow k+1$;

Fig. 1. Steps of the basic VNS.