Advances in Engineering Software 47 (2012) 1–6

Contents lists available at SciVerse ScienceDirect

Advances in Engineering Software

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

Application of particle swarm optimization and simulated annealing algorithms in flow shop scheduling problem under linear deterioration

M. Bank [a], S.M.T. Fatemi Ghomi [a,*], F. Jolai [b], J. Behnamian [a]

[a] Department of Industrial Engineering, Amirkabir University of Technology, 424 Hafez Avenue, 1591634311 Tehran, Iran
[b] Department of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran

ARTICLE INFO

Article history:
Received 17 November 2010
Received in revised form 28 November 2011
Accepted 5 December 2011
Available online 29 December 2011

Keywords:
Flow shop
Scheduling
Deteriorating job
Simulated annealing
Particle swarm optimization
Total tardiness

ABSTRACT

This paper studies a permutation flow shop scheduling problem with deteriorating jobs. Deteriorating jobs are the jobs which the processing time depends on the waiting time before process starts. A particle swarm optimization algorithm with and without a proposed local search is developed to determine a job sequence with minimization of the total tardiness criterion. Furthermore, a simulated annealing is proposed to solve the problem. We compare the performance of these algorithms to achieve an optimal or near optimal solution. It is concluded that the particle swarm optimization algorithm with local search gives promising solutions. The quality of solution obtained by particle swarm optimization algorithm with local search is superior to that of the simulated annealing algorithm, but the simulated annealing algorithm takes shorter time to find a schedule solution.

1. Introduction

In classical scheduling problems, the job processing time is considered to be constant. However, scheduling problems with this restrictive assumption may correspond to a limited number of real-world cases. By omitting this assumption and considering a variable job processing time many of the real-world problems will be studied. In this way, Gupta and Gupta [1] proposed an interesting scheduling model in which the processing time of the jobs depends on the jobs' starting time with a polynomial function. This scheduling model could interest many researchers to continue that work. They named this scheduling problem as a deterioration scheduling model. For more information see Alidaee and Womer [2], and Cheng et al. [3].

Mosheiov [4] considered a simple linear deterioration function in a single machine scheduling problem. He studied the effect of deterioration on optimal schedule of the various classified single machine problem. Bachman and Janiak [5] minimized the maximum lateness criterion in a single machine scheduling problem with deteriorating jobs. They showed the NP-completeness of the problem and proposed two heuristic algorithms to solve the problem. Mosheiov [6] studied multi machine scheduling problem with simple linear deterioration to minimize the makespan. He proved that this problem is NP-hard for the two-machine case. Mhlanga and Hindi [7] studied scheduling deteriorating jobs on parallel machines. They developed both step descend search and simulated annealing to minimize the makespan. Kononov and Gawiejnowicz [8] also showed that the two machine flow shop under simple linear deterioration is strongly NP-hard and in three machine flow shop there not exists the polynomial-time approximation algorithm with the worst case ratio bounded by a constant. They also showed that optimal schedule in two and three machine flow shops with proportional processing time ($P_{ij}(t) = c_{ij}(a + bt)$) is the permutation schedule. Zhao et al. [9] proposed some optimal algorithm for single machine problem under simple linear deterioration. For two machine flow shop scheduling problem they proved that the optimal schedule can be obtained by Johnson's rule.

Lee et al. [10] considered a makespan flow shop problem with deteriorating jobs. They proposed an exact algorithm to solve most of the problems up to 32 jobs and a heuristic algorithm to derive the near-optimal solution. Davoudpour and Hadji Molana [11] investigated a flow shop scheduling problem with simple linear deterioration. They used an electromagnetic algorithm to find a near optimal solution. Alagheband et al. [12] studied m-machine scheduling problem and deployed a modified simulated annealing algorithm to solve that NP-hard problem. Layegh et al. [13] proposed a memetic algorithm to find a near optimal schedule in a single machine scheduling under step deterioration and compared the results with the those of complete enumeration.

Particle swarm optimization (PSO) algorithm was introduced firstly by Kennedy and Eberhart [14] in 1995 and applied in continuous optimization problem extendedly. Taşgetiren and Liang [15] in 2003 applied it in a lot sizing problem to minimize the total ordering and handling costs. Experimental results of that research show that the binary PSO algorithm is capable of finding optimal

* Corresponding author. Tel.: +98 21 64545381; fax: +98 21 66954569.
E-mail address: Fatemi@aut.ac.ir (S.M.T. Fatemi Ghomi).

0965-9978/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2011.12.001

results in almost all cases. Tasgetiren et al. [16] presented a PSO algorithm to solve the single machine problem with total weighted tardiness criterion. A heuristic rule named the *Smallest Position Value* (*SPV*) rule was developed to enable the continuous PSO algorithm to be applied to all classes of sequencing problems. In 2005 Liao et al. [17] proposed a PSO algorithm, extended from discrete PSO, for flow shop scheduling problems. In the algorithm, the particles and the velocities were redefined, and an efficient approach was developed to move a particle to the new sequence. Tasgetiren et al. [18] presented a PSO algorithm to solve the permutation flow shop sequencing problem (PFSP) with the objectives of minimizing makespan and the total flow time of jobs. They embedded a very efficient local search, called *variable neighborhood search* (VNS), in the PSO algorithm to solve the well known benchmark suites in the literature. Pan et al. [19] developed a discrete PSO (DPSO) algorithm to solve the no-wait flow shop scheduling problem with both makespan and total flow time criteria. In their algorithm the particles represent district job permutation.

In this paper a PSO algorithm and a simulated annealing algorithm are developed to solve flow shop deterioration scheduling problem (FDSP) with minimization of the total tardiness criterion. The paper is organized as follows. Next section is problem definition in which the formulation and notations used in the paper are described. In Section 3 a PSO algorithm is proposed. A heuristic rule called SPV rule proposed by Tasgetiren et al. [16] is used to adapt the algorithm to discrete space. In subsequent section, a simulated annealing is developed to evaluate the performance of PSO algorithm. In Section 5, a comparison between the results of two proposed algorithms is made and the performance of the algorithms is examined. Finally, Section 8 is devoted to conclusions and recommendation for future studies.

2. Problem formulation

There is a set of n jobs $\{J_1,J_2,\ldots,J_N\}$ which are going to be processed on m-machines flow shop $\{M_1,M_2,\ldots,M_m\}$. Job's processing time on each machine depends on its starting time by a linear increasing function considered by Kononov and Gawiejnowicz [8] and Ng et al. [20]:

$$P_{ij}(t) = c_{ij}(a + bt), \tag{1}$$

where t is the time when the job J_j starts processing on machine M_i, ac_{ij} stands in place of its normal processing time and bc_{ij} makes the deterioration rate of that job where $a, b \geqslant 0$ .We assume that there is an unlimited storage space between each two-machine, all jobs are available for processing at time zero, each machine can handle only one job at a time and a job can be processed only on one machine at a time. No breakdown is allowed in the machines and there is no precedence among jobs. All of the parameters are known and deterministic. The objective is to find the optimal sequence that minimizes total tardiness in this permutation flow shop scheduling problem.

3. Particle swarm optimization algorithm for FDSP

Particle swarm optimization algorithm is one of the latest evolutionary optimization techniques. This algorithm is based on communication and interaction between the members. It means that each member of the PSO algorithm which is named as *particle* determines its position by combining the history of its own best location with those of other members of the swarm. The behavior of the members in PSO is a simulation of bird flock or fish school movements.

The initial PSO algorithm proposed by Kennedy and Eberhart [14] in 1995, only can be used in continuous spaces. Although there are several researches that modified this algorithm to be used in discrete spaces and combinatorial optimization, the application of this method is still limited in this type of problems. In this paper, a PSO algorithm is developed which uses the *smallest position value* (SPV) heuristic rule proposed by Tesgetiren et al. [16] to solve the flow shop deterioration scheduling problem. The main components in the PSO algorithm are introduced as follows:

- *Particle*: X_i^t represents ith particle in tth iteration. It has n dimensions that X_{ij}^t stands for its jth dimension. The set of NP particles is denoted by X_t and named as population. NP is population size.
- *Velocity*: each particle in PSO algorithm has a velocity that determines its direction represented by V_i^t. Same as the particles V_i^t has n-dimensions symbolized by V_{ij}^t.
- *Personal best:* in this algorithm each particle saves its best position and best total tardiness value experiments in PB_i^t and fpb_i respectively. PB_i^t has n-dimensions symbolized by PB_{ij}^t.
- *Global best:* the best position which is experienced by all of the population members during each iteration is denoted as GB^t. And fgb^t stands for its total tardiness value. GB^t has n-dimensions symbolized by GB_j^t.
- *Inertia weight:* ω^t determines the impact of previous velocity on current velocity.

The position of each dimension of ith particle in the algorithm can be obtained from the following relation:

$$X_{ij}^t = V_{ij}^t + X_{ij}^{t-1} \tag{2}$$

where V_{ij}^t can be achieved from the following equation:

$$V_{ij}^t = \omega^t V_{ij}^{t-1} + C_1 r_1 \left(PB_{ij}^{t-1} - X_{ij}^{t-1}\right) + C_2 r_2 \left(GB_j^{t-1} - X_{ij}^{t-1}\right) \tag{3}$$

The inertia weight ω^t is an important parameter in PSO. This parameter determines the search scope. It means that global search scope can be obtained by large values for this parameter and small values may cause local search. Therefore, it is important to select the value of this weight in a way to create balance between local and global searches. In this algorithm the inertia weight decreases linearly in the range of [0.9,0.4]. These values for inertia weight are obtained from the literature and the function shape selection is based on our experiments and pilot tests. C_1 and C_2 are social and cognitive parameters. According to the literature the values of these two parameters are assumed equal to 2 [21]. r_1 and r_2 are uniform random numbers that have to be reproduced for each dimension of each particle in each iteration.

Initial population and initial velocities are created randomly. The uniform distribution was used in this random creation. The velocity has to be in the range of $[V_{min}, V_{max}]$ where $V_{max} = -V_{min} = 1$.

By applying the SPV rule, the position value of particles would be changed to a permutation to calculate the total tardiness value. The number of dimensions of each particle is equal to the number of jobs to be scheduled. Therefore, this heuristic rule changes the position values to a job sequence. The steps of this rule can be summarized as follows:

Step 1: Consider a n-dimension particle X_i^t obtained in each iteration of PSO algorithm and S as a job sequence. Set $k = 1$.
Step 2: Find the dimension j that has the kth smallest position value in X_i^t , and place job J_j in kth position of schedule S.
Step 3: Put $k = k+1$.
Step 4: If $k \leqslant n$ go to step 2; otherwise stop the algorithm and report schedule S.

3.1. Local search

There are two types of reaching to a neighbor in PSO algorithm. In the first type, a neighborhood search is applied on the particle position and then the related schedule will be achieved. In the