J Ind Eng Int (2016) 12:299–310
DOI 10.1007/s40092-016-0147-9

ORIGINAL RESEARCH

JIT single machine scheduling problem with periodic preventive maintenance

Mohammadreza Shahriari[1] · Naghi Shoja[2] · Amir Ebrahimi Zade[3] · Sasan Barak[4] · Mani Sharifi[5]

Received: 18 February 2016 / Accepted: 26 February 2016 / Published online: 15 March 2016
© The Author(s) 2016. This article is published with open access at Springerlink.com

Abstract This article investigates a JIT single machine scheduling problem with a periodic preventive maintenance. Also to maintain the quality of the products, there is a limitation on the maximum number of allowable jobs in each period. The proposed bi-objective mixed integer model minimizes total earliness-tardiness and makespan simultaneously. Due to the computational complexity of the problem, multi-objective particle swarm optimization (MOPSO) algorithm is implemented. Also, as well as MOPSO, two other optimization algorithms are used for comparing the results. Eventually, Taguchi method with metrics analysis is presented to tune the algorithms' parameters and a multiple criterion decision making technique based on the technique for order of preference by similarity to ideal solution is applied to choose the best algorithm. Comparison results confirmed the supremacy of MOPSO to the other algorithms.

Keywords Scheduling · Single machine · Periodic maintenance · Total earliness-tardiness · Multi-objective optimization · MCDM

✉ Mohammadreza Shahriari
shahriari@iau.ae

[1] Faculty of Management, South Tehran Branch, Islamic Azad University, Tehran, Iran

[2] Department of Industrial Engineering, Firoozkooh Branch, Islamic Azad University, Firoozkooh, Iran

[3] Department of Industrial Engineering, Amirkabir University of Technology, Tehran, Iran

[4] Faculty of Economics, Technical University of Ostrava, Ostrava, Czech Republic

[5] Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran

Introduction

Most of the manufacturing organizations try to implement some of the ideas adopted by Just in Time (JIT) philosophy, like on time delivery or minimum inventory (Salameh and Ghattas 2001). In this paper we introduce and formulate a JIT single machine scheduling problem with a periodic preventive maintenance on the machine. In most of the scheduling problems it is assumed that the machine is available interruptedly while, in practice, it may be unavailable due to causes like breakdown or preventive maintenance. According to the British Standard Institute (BSI), "Maintenance is a combination of any actions to retain an item in, or restore it to an acceptable condition" (BSI 1984). Periodic preventive maintenance, which is a fundamental part of JIT production, consists of regular preventive measures to increase machine reliability and to decrease breakdown probability during manufacturing process. In some of the manufacturing processes, overuse of the tool might decrease quality of the work piece. Therefore, when the work piece is expensive or when the accuracy is necessary, we change the tool before it is amortized. A well-known example is the printed circuit board manufacturing process in which the drilling machine is one of the most important devices. Thus not only should the machine stop to maintain after a period of processing time, but also the machine should stop to change the micro-drill after fixed times of using. Accordingly, the proposed model holds a limitation on the maximum number of processed jobs during each period. The objective of the proposed bi-objective mixed integer model is minimizing total earliness-tardiness costs and makespan simultaneously.

Machine unavailability problem has been investigated in the literature due to causes like machine breakdown, tool change or preventive maintenance. Machine breakdown or

product quality loss is probable when a machine continues to work unceasingly for a long time. To avoid this situation, preventive maintenance is conducted on the machine which may be periodic or flexible (Xu et al. 2015). In a flexible maintenance the earliest and latest start time are determined and the maintenance process is allowed to start during this period. Yang et al. were the first to study a single machine scheduling problem with a flexible maintenance (Yang et al. 2002). They investigated the problem to minimize makespan and provided a proof for NP-Hardness of the problem. Qi et al. (1999) studied a problem in which multiple maintenance processes and jobs are to be scheduled on a single machine. They proposed heuristics as well as Branch and Bound based approaches to determine the sequence of jobs and maintenance start times while total completion time is minimized. Furthermore, Chen proposed a mixed binary integer programming and a heuristic to minimize mean flow time (Chen 2006). Also, Wan (Wan 2014) investigates on minimizing total earliness and tardiness in a single machine scheduling problem with a common due date for all jobs and a flexible maintenance. Luo et al. proposed polynomial algorithms for a single machine scheduling with flexible maintenance and various objective functions (Luo et al. 2015). Low et al. studied a single machine scheduling with flexible maintenance under two strategies, the first one was the flexible maintenance and the latter was changing the tool, after a predetermined number of jobs were conducted on the machine (Low et al. 2010). Their model was aimed at minimizing the makespan.

In addition, there are various scheduling researches addressing a periodic maintenance process. For example (Liao and Chen 2003) proposes a branch and bound based algorithm to minimize maximum tardiness or (Chen 2006) proposes a heuristic to minimize mean flow time in a problem with periodic maintenance. Benmansour et al. investigated on a JIT single machine scheduling problem with periodic maintenance in which the objective was to minimize maximum tardiness and maximum earliness (Benmansour et al. 2014), they also proposed a heuristic to solve the problem efficiently. On the other hand, in some cases like (Liao et al. 2007; Yang et al. 2008) it is assumed that the machine must stop for maintenance after a fixed number of processes. Hsu et al. studied a single machine scheduling problem, with a makespan minimizing objective, under two strategies; namely periodic maintenance and limited number of operations during each period (Hsu et al. 2010). They proposed a two stage binary integer programming and two efficient heuristics, best fit butterfly (BBF) and best fit decreasing (DBF), for solving large scale problems. Also Ebrahimyzade and Fakhrzad proposed a new mathematical model and dynamic genetic algorihtm (GA) to solve this problem (Ebrahimy Zade and Fakhrzad 2013). Computational results revealed that the solutions from the proposed dynamic GA had a better quality than the BBF and DBF.

On time delivery and minimum inventory costs are amongst the most important targets of a JIT manufactoring system. However, minimizing earliness and tardiness costs does not necessarily imply minimum inventory. Therefore, in some cases like Behnamian (2014), Gao et al. (2014), Behnamian and Fatemi Ghomi (2014), Gao (2010), and Eren (2007) total earliness–tardiness and makespan minimization are considered simultaneously, however, none of them mentioned the preventive maintenance, despite its substantial role in JIT philosophy. Table 1 delineates some properties of the proposed model with the most related articles in the literature.

Considering the above literature review, main contributions of the article are as follows:

1. Proposing a mathematical model for JIT single machine scheduling problem with periodic maintenance.
2. Simultaneous minimization of makespan, earliness and tardiness in a JIT scheduling problem with periodic maintenance.
3. Proposing three different multi-objective optimization algorithms to solve the problem.

The rest of this paper is organized as follows. The proposed mixed integer model is presented in the next section and considering computational complexity of the problem we propose three multi-objective optimization algorithms for solving the problems in the following section. Parameters tuning for each algorithms are described next, followed by section on computational results and finally conclusions and further research directions are provided.

Proposed model

In this section, we define the discussed problem formally. The problem is composed of n nonresumable jobs available at time zero to be scheduled on a single machine. It is assumed that the machine does not have ready time and the jobs will be delivered to customers immediately after they are completed. Each job has a unique due date. When a job is submitted to the customer before the due date it is called an early job and if it is delivered after the due date, it is a tardy job. It is assumed that both earliness and tardiness are costly, although when earliness is desired from the customer's point of view, a negative cost may be applied in the model. Preventive maintenance is conducted on the machine after a fixed period T. Moreover, to maintain the quality of the products, there is a limitation on the

Table 1 Some extensions to the scheduling problem with maintenance process

Article	Maintenance	Tool change	Minimizing	Solution approach
Yang et al. (2002)	Flexible	–	Makespan	Exact
Qi et al. (1999)	Flexible	–	Total completion time	Heuristics and branch and bound
Chen (2006)	Periodic/flexible	–	Mean flow time	Heuristic
Wan (2014)	Flexible	–	Total tardiness-earliness	Exact
Luo et al. (2015)	Flexible	✔	Makespan	Exact
Low et al. (2010)	Flexible	✔	Makespan	Heuristic
Liao et al. (2007)	–	✔	Makespan	Branch and bound
Yang et al. (2008)	–	✔	Makespan	Heuristic
Liao and Chen (2003)	Periodic	–	Maximum tardiness	Heuristic
Benmansour et al. (2014)	Periodic	–	Maximum earliness-tardiness	Heuristic
Hsu et al. (2010)	Periodic	✔	Makespan	Heuristic
Ebrahimy Zade and Fakhrzad (2013)	Periodic	✔	Makespan	GA
This article	Periodic	✔	(1) Total tardiness-earliness and (2) Makespan	MOPSO

maximum number of allowable jobs during each working period and after that, during the maintenance period, the tool will be changed.

The first objective in the proposed model minimizes total earliness-tardiness. On the other hand, machine/operator idleness and unnecessary work in process (WIP) are costly to the manufacturing system but the first objective does not encompass them. For example assume a problem in which the fixed processing period is 5, fixed maintenance duration is 2, and maximum number of doable jobs during each processing period are 3. Process time and due date for each jobs are presented in Table 2. Two different sequences, namely A and B, are demonstrated in Fig. 1. As evident, total earliness-tardiness for both sequences is 5, thus they are equivalent from the first objective's point of view. However, in sequence B we have less idle time and subsequently total working time for the machine and operator are less than sequence A which results in a less expensive manufacturing system. For this purpose, the second objective in the proposed model minimizes the makespan.

The set of problem parameters and indices in the proposed model are as follows:

i Index for the jobs; $i = \{1, 2,\ldots, n\}$.
j Index for the position of a job in a period; $j = \{1,2,\ldots, k\}$.
s Index for the periods.
p_i Process time for job i.
t Duration of a working period (between two consecutive periods).
m Fixed maintenance duration.
d_i Due date for job i.
α_i Earliness penalty coefficient for job i.
β_i Tardiness penalty coefficient for job i.

And the set of decision variables are as follows:

x_{ijs} Is a binary variable which is 1 if job i is in the jth position of period s; otherwise it is 0.
C_{ijs} Completion time for job i in the jth position of period s.
T_i Tardiness for job i.
E_i Earliness for job i.

Considering the fact that neither the number of periods nor the number of jobs in each period are predetermined, initially $\left(l = \max\left\{n, \left[\frac{\max d_i}{t+m}\right] + 1\right\}\right)$ maximum number of

Table 2 Process time and due dates

Job number	Process time	Due date	Completion time (A)	Completion time (B)
1	4	4	4	4
2	3	17	17	12
3	1	5	10	5
4	2	9	9	9

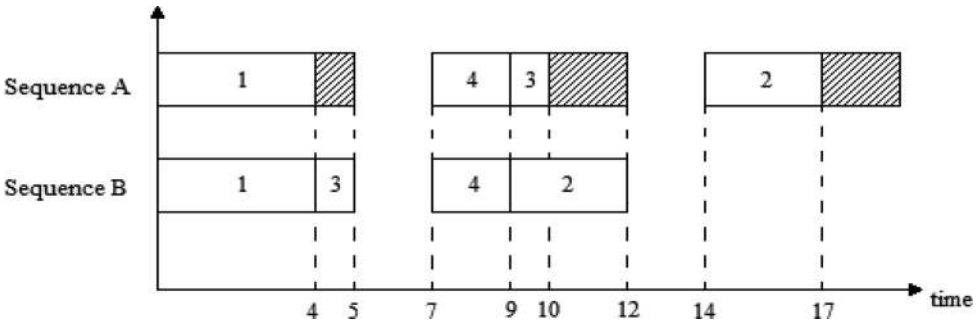

Fig. 1 Two different sequences

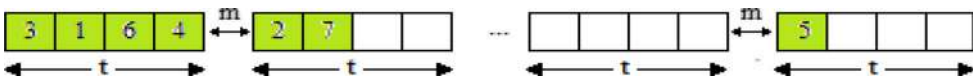

Fig. 2 A hypothetical solution

possible periods are assumed, each of which containing k positions for the jobs. Thereafter considering the duration of each period, maximum number of applicable jobs in each period, and the objective functions; the jobs are arranged in the periods. Figure 2 presents a possible solution for a problem with $n = 7$, $k = 4$.

The proposed mixed integer nonlinear model is as follows:

$$\min \sum_i \alpha_i E_i + \beta_i T_i \tag{1}$$

$$\min \max_{i,j,k} \{c_{ijs}\} \tag{2}$$

Subjects to:

$$\sum_j \sum_s x_{ijs} = 1 \quad \forall i \tag{3}$$

$$\sum_i x_{ijs} \leq 1 \ \forall j, s \tag{4}$$

$$\sum_j \sum_i p_i x_{ijs} \leq t \quad \forall s \tag{5}$$

$$\sum_i \sum_j x_{ijs} \leq k \quad \forall s \tag{6}$$

$$c_{ijs} = (t+m)(k-1)x_{ijs} + \left(\sum_{i'=1}^{n} \sum_{j'=1}^{j} p_{i'} x_{i'j's} \right) x_{ijs} \quad \forall i, j, s \tag{7}$$

$$T_i \geq \sum_j \sum_s c_{ijs} - d_i \quad \forall i \tag{8}$$

$$E_i \geq d_i - \sum_j \sum_s c_{ijs} \quad \forall i \tag{9}$$

$$x_{ijs} \in \{0, 1\}, \quad c_{ijs}, T_i, E_i \geq 0 \tag{10}$$

In the above equations, Eq. (1) is the first objective of the proposed model that minimizes total earliness-tardiness and Eq. (2) is the second objective that minimizes the makespan. Equation (3) guarantees that each job appears in a unique position of a period. According to constraints Eq. (4), it is guaranteed that at most one job will be located in each position of a period. Constraints Eq. (5) ensure that the sum of processing time for the jobs being in the same period is less than t. According to Eq. (6), maximum number of doable jobs in each period is less than k. Equation (7) calculates the completion time for job i in the jth position of period s. Right side of this equation totalizes duration of the previous periods with the process time of the jobs prior to i (including i) in the jth period. For a given job i, tardiness and earliness are $\max\{0, C_i - d_i\}$ and $\max\{0, d_i - C_i\}$ respectively. Constraints Eqs. (8) and (9) provide lower bounds for tardiness (T_i) and earliness (E_i) respectively. According to Eq. (10), x_{ijs} is a binary variable and C_{ijs}, Ti, E_i are nonnegative variables.

Proposed solving algorithm

Due to the computational complexity of the problem, multi-objective particle swarm optimization (MOPSO) algorithm is implemented. In addition to the MOPSO, two other multi-objective optimization algorithms are used for comparing the results.

MOPSO algorithm

Optimization algorithm of particle swarm which was first introduced in 1995 (Yousefi et al. 2013), is a population based stochastic optimization technique inspired by social

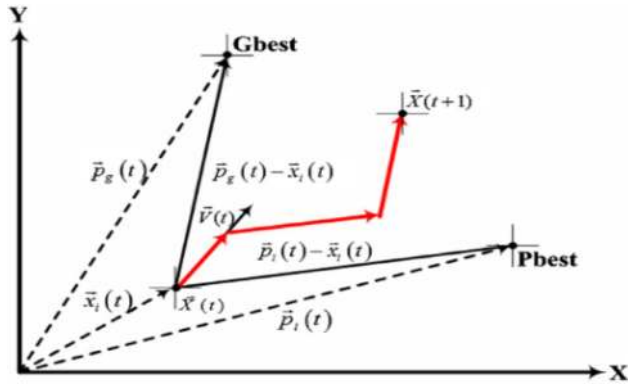

Fig. 3 Decoding and encoding solution

behavior of the swarm members such as bird flocking, fish training, etc. PSO's capability of storing the previous solution helps to reduce memory usage and speed up the CPU. In addition, a few parameters which should be adjusted in PSO and obtaining remarkable results in literature make PSO a useful method to apply. A brief description of how the algorithm works is as follows: Initially, a particle is identified as the best particle in a neighborhood of particles, based on its fitness. All particles are then accelerated in the direction of this particle as well as the direction of their own best solutions that they have covered previously.

Consider a group of N particles that are searching a global optimal solution in a D dimensions space. Vectors $X_i = (x_{1i}, x_{2i},\ldots, x_{Di})$, $V_i = (v_{1i}, v_{2i},\ldots, v_{Di})$ and $Pbest = (Pbest_{i1}, Pbest_{i2}, \ldots, Pbest_{iD})$ Shows position, velocity, and the best personal position of each particle respectively. *Gbest* is also a *leader* position visited by the whole particle swarm population. In any iteration of PSO, the position and velocity of particles are updated according to Eqs. (11) and (12) and Fig. 3.

$$v_{ij}^{t+1} = w \times v_{ij}^{t} + C_1 \times r_1 \times (Pbest_{ij}^{t} - x_{ij}^{t}) + C_2 \times r_2 \times (leader_{ij}^{t} - x_{ij}^{t}) \quad (11)$$

$$x^{t+1}ij = x^{t}ij + v^{t+1}ij \quad (12)$$

Based on Eqs. (11) and (12), v_j^t and v_{ij}^{t+1} are the velocity of particle i in jth dimension and iteration t and $t + 1$, x_{ij}^t and x_{ij}^{t+1} are the position of particle i in jth dimension and iteration t and $t + 1$, $Pbest_{ij}^t$ is the best previous position of particle i in jth dimension and iteration t, and $leader_{ij}^t$ (best global position) is the leader position for particle i in jth dimension and iteration t. C_1 and C_2 are the personal learning coefficient and the social learning factor respectively. r_1 and r_2 are random numbers between 0 and 1 and ω is an inertia factor usually in the range [0.8, 1.2].

Multi-objective PSO (MOPSO) is an extension of PSO proposed by (Coello et al. 2004) for multi-objective optimization problems. MOPSO stores the non-dominated solutions in 'repository', an external archive for solutions. The members of repository are not dominating each other and they provide an approximation of real Pareto frontier of the optimization problem. Repository members are updated by region based selection (grid index). Furthermore, in MOPSO, each particle chooses a solution in the repository as its leader with region based selection, instead of a unique global best for all particles. The Pseudo Code of MOPSO is presented as follows:

```
Start
Input nParticle, C1, C2, W, Max iteration;
Generate initial particle;
Evaluation fitness value of initial particle;
Calculate best personal memory;
Create best leader memory;
Create grid index for solution dimension;
Determine repository member;
Determine grid for repository members;
        For it=1to Max iteration
                For j=1to nParticle
                        Update particle position;
                        Evaluate fitness value of
                    particle;
                        Update best personal memory;
                        Update best leader memory;
                End for
                Determine repository members;
                Combine new repository members with
                old repository members rep = {rep ∪ rep_new};
                Update repository members by non-
                dominance sorting algorithm;
                Update grid index for solution
                dimension;
                Find grid for repository members;
                Delete extra repository members;
        End for
 Output: Extract the repository front;
End
```

For decoding process of MOPSO algorithm, N random numbers between [0, 1] is used, where N is the number of works. Encoding process is implemented by ordering these numbers increasingly while the minimum and maximum numbers are given to the first and the last works, respectively. Figure 4 illustrates the process.

Fig. 4 Decoding and encoding solution

Non-dominated sorting genetic algorithm II (NSGA-II)

NSGA-II algorithm was proposed by Deb, Pratap, Agarwal and Meyarivan (Deb et al. 2002). It uses a ranking scheme called the fast non-dominated sorting approach which requires a computational complexity of at most to rank the individuals, where M is the number of objectives and N is population size.

The Pseudo Code of NSGA-II is presented as follows:

```
Begin
Input nPop, Pc, Pm, Max iteration;
Generate initial population;
Evaluation fitness value of initial population;
Assign rank base on Pareto non-dominance sort;
For i=1 to Max iteration do
    For j=1 to 2×round((Pc×nPop)/2) ;
        Select parent by binary tournament selection;
        Apply crossover;
    End for
    Combine offspring and population;
    For j=1 to round(Pm×nPop);
        Select Chromosome by random selection;
        Apply mutation;
    End for
  Combine mutation members and population;
  Assign rank based on Pareto non-dominance sorting
  algorithm;
  Calculated the crowding distance of individuals in each
   front;
  Select the best nPop individual base on rank and
   crowded distance;
  End for
Output: Extract the best Pareto front;
End
```

where P_m is percentage of mutation and P_c is percentage of cross over.

Non-dominated ranking genetic algorithms (NRGA)

Al Jadaan et al. (2008) presented NRGA by exchanging the selection strategy of NSGA-II from the tournament selection to the roulette wheel. The Pseudo Code of NRGA is presented as follows:

```
Begin
Input nPop, Pc, Pm, Max iteration;
Generate initial population;
Evaluation fitness value of initial population;
Assign rank base on Pareto non-dominance sort;
For i=1 to Max iteration do
    For j=1 to 2×round((Pc×nPop)/2) ;
        Select parents by Pareto roulette wheel selection;
        Apply tournament crossover;
    End for
    Combine offspring and population;
    For j=1 to round(Pm×nPop) ;
        Select Chromosome by random selection;
        Apply mutation;
    End for
  Combine mutation members and population;
  Assign rank based on Pareto non-dominance sorting
  algorithm;
  Calculated the crowding distance of individuals in each
   front;
  Select the best nPop individual base on rank and
   crowded distance;
  End for
Output: Extract the best Pareto front;
End
```

PSO algorithm was developed for continuous searching space optimization problems. Since we encode the searching space of this problem continuously, therefore, PSO algorithm is used appropriately in this paper. Moreover, regarding to population based property of the MOPSO, we use two well-known population based algorithms namely NRGA and NSGAII to verify and validate the MOPSO results.

Parameter tuning

Undoubtedly, the results of a MOPSO, NRGA, and NSGA II algorithms to attain better fitness function value significantly depends on their parameters. The main parameters of a PSO algorithm that should be calibrated at the best level are: cognitive factor (C1), social factor (C2), swarm size (*N*-Particle), number of iterations (*N*-It), number of repository (*N* Rep), number of grid (*N* Grid), and inertia factor (w). Also, the main parameters of NRGA and NSGA

Tables 3 S/N for three repetitions in for MOPSO with orthogonal array L^{27}

C1	C2	W	Max It	N Particle	N Rep	N Grid	MOPSO			
							$MOCV_1$	$MOCV_2$	$MOCV_3$	S/N
1	1	1	1	1	1	1	1.003	1.154	2.044	−3.36837
1	1	1	1	2	2	2	1.354	1.837	1.593	−4.1193
1	1	1	1	3	3	3	2.364	1.015	3.427	−7.86824
1	2	2	2	1	1	1	2.781	1.141	1.014	−5.25651
1	2	2	2	2	2	2	1.019	1.011	1.205	−0.68496
1	2	2	2	3	3	3	1.223	2.826	4.379	−9.80119
1	3	3	3	1	1	1	1.014	1.338	1.345	−1.88222
1	3	3	3	2	2	2	1.027	2.011	1.339	−3.6121
1	3	3	3	3	3	3	1.009	1.019	1.145	−0.50182
2	1	2	3	1	2	3	1.622	5.844	1.417	−11.1161
2	1	2	3	2	3	1	3.494	1.884	1.852	−8.05895
2	1	2	3	3	1	2	2.750	5.420	7.217	−14.7239
2	2	3	1	1	2	3	1.013	1.034	1.009	−0.16115
2	2	3	1	2	3	1	1.004	1.024	1.027	−0.15824
2	2	3	1	3	1	2	1.019	1.026	1.028	−0.20889
2	3	1	2	1	2	3	1.025	1.030	1.009	−0.18368
2	3	1	2	2	3	1	1.017	1.012	1.018	−0.13505
2	3	1	2	3	1	2	1.006	1.012	1.026	−0.12676
3	1	3	2	1	3	2	1.184	1.018	1.015	−0.63009
3	1	3	2	2	1	3	1.261	1.614	1.024	−2.42516
3	1	3	2	3	2	1	2.211	1.016	1.029	−3.6671
3	2	1	3	1	3	2	1.034	1.018	1.219	−0.78147
3	2	1	3	2	1	3	1.018	1.029	1.228	−0.79561
3	2	1	3	3	2	1	1.158	1.014	1.180	−0.98247
3	3	2	1	1	3	2	1.014	1.021	1.020	−0.15784
3	3	2	1	2	1	3	1.012	1.011	1.025	−0.13805
3	3	2	1	3	2	1	1.014	1.021	1.021	−0.16069

II algorithms are: percentage of crossover (P_c), percentage of mutation (P_m), maximum iteration (Max It), and population size (N pop).

Therefore, in this section, the parameters of all algorithms are calibrated by using Taguchi method (Fazel Zarandi et al. 2013). Instead of the Fisher factorial designs, Taguchi developed fractional factorial experiments (FFEs) to reduce complexity of experiments in the full factorial designs. Taguchi method analyses the results in two ways: (1) analysis of variance for experiments with a single replicate, (2) the signal to noise ratio (S/N) for experiments with multiple replications where (N) is noise factor and (S) is controllable factors. Since the one with multiple replications has a better performance, the S/N is applied in this research to analyze the solutions. For more information regarding the Taguchi method see Taguchi et al. (2005).

For tuning of the proposed algorithm's parameters with Taguchi model properly, a new response which is representing different quality of a solution is considered for the experiments. In Pareto based algorithms, two main goals are interesting (1) convergence and (2) diversity. Mean ideal distance (MID) is the one that measure the convergence rate of the algorithms. MID measures the convergence rate of Pareto frontier members to a certain point (0, 0). Also, diversity measures the extension of the Pareto frontier. Spacing is the one that measure the diversity rate of the algorithms. It measures the standard deviation of the distances among solutions of the Pareto frontier. The Diversity and the MID metrics, as representatives of the multi-objective goals, are used to define multi-objective coefficient of variation (MOCV) in Eq. (13) as follows:

$$\text{MOCV} = \frac{\text{MID}}{\text{Diversity}} \tag{13}$$

By this definition, the two goals of the Pareto-based algorithms are considered simultaneously as a single response and more reliable outputs can be expected.

To conduct the Taguchi method more comprehensively, we have implemented a three stages orthogonal array experiments with MOCV. The purpose of conducting these

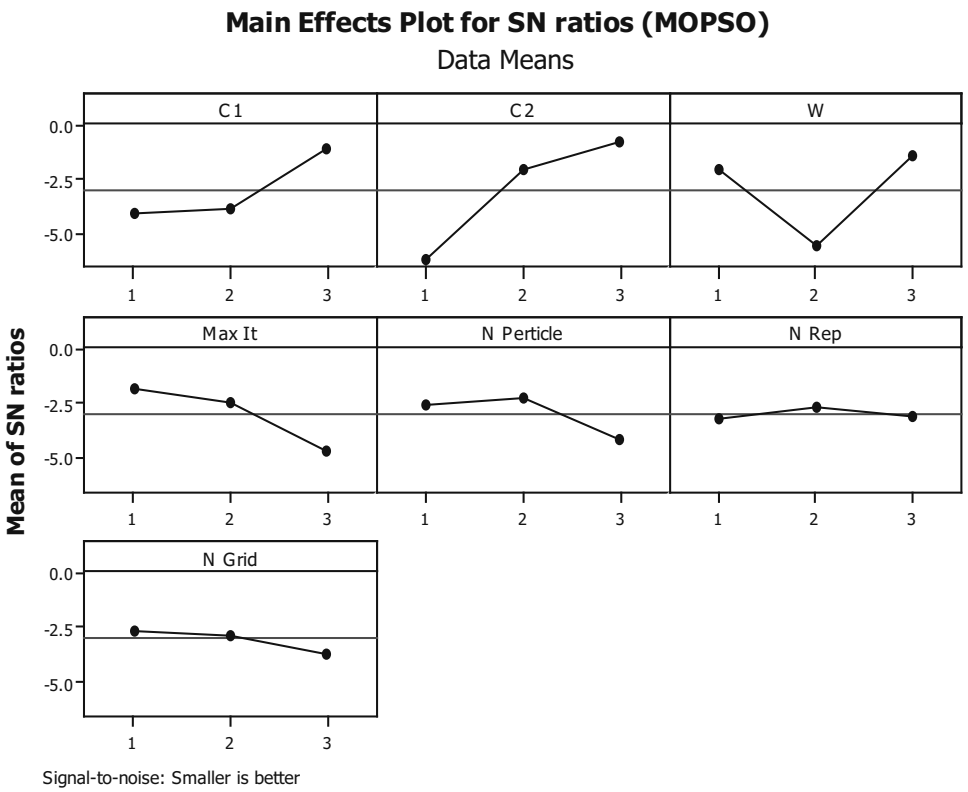

Fig. 5 The mean S/N plot for different levels of the MOPSO parameters

Table 4 Optimal parameters for MOPSO

Parameter	MOPSO			
	Level 1	Level 2	Level 3	Optimal value
Personal learning coefficient (C1)	1	1.4962	2	2
Global learning coefficient (C2)	1	1.4962	2	2
Inertia weight (W)	0.6	0.7298	0.9	0.9
Maximum iteration (Max It)	$5 \times n$	$10 \times n$	$15 \times n$	$5 \times n$
Particle size (N particle)	100	150	200	150
Repository size (N Rep)	75	100	100	100
Number of grids (N Grid)	5	8	10	5

arrays is to define the optimum level for each controllable parameter which cooperates to get the better fitness function value. From orthogonal arrays, each problem is run three times and since a solution with the highest MOCV is desired, the aim is to find minimize *S/N* calculated by Eq. 14 (Sadeghi et al. 2014).

$$S/N = -10 \log\left(\frac{1}{n}\sum_{i=1}^{n} MOCV_i^2\right) \quad (14)$$

where, MOCV_i, $i = 1, 2, 3$ is the solution in ith replication of the Taguchi method and $n = 3$ is the number of replications in experiments.

Table 3 shows the experimental results of MOPSO with L^{27} orthogonal array under different scenarios of the parameters combinations, respectively where "1", "2", and "3" refer to the first, the second, and the third level of the parameters. Regarding Eq. (14), these tables present S/Ns as well. In addition, it can be seen from Fig. 5 that the highest mean of S/N is the best. Therefore, Table 4 contains the optimal parameter values of the MOPSO algorithm. The same calculation is done for NRGA and NSGA II algorithms and the optimal parameters along with their levels are presented in Table 5.

Results comparison and discussion

In this section, we study the ability of algorithms (MOPSO, NRGA, and NSGAII) on test problems which is implemented in Matlab Software R2013a. It should be noted

Table 5 Optimal parameters for NSGA-II and NRGA

Parameter	NSGA-II				NRGA-II			
	Level 1	Level 2	Level 3	Optimal value	Level 1	Level 2	Level 3	Optimal value
Percentage of crossover (Pc)	0.7	0.8	0.9	0.7	0.7	0.8	0.9	0.9
Percentage of mutation (Pc)	0.2	0.15	0.1	0.15	0.2	0.15	0.1	0.1
Maximum iteration (Max It)	$5 \times n$	$10 \times n$	$15 \times n$	$5 \times n$	$5 \times n$	$10 \times n$	$15 \times n$	$15 \times n$
Population size (N pop)	100	150	200	100	100	150	200	100

Table 6 The result metric for NSGA-II

	MID	Max spread	SNS	NPS	RAS	Spacing
Problem 1	8,188,090	81,755	65,535	1	0	0
Problem 2	3,115,196	61,820	4,316,702	2	0.0173	0
Problem 3	1,514,438	105,448	1,520,510	7	0.0170	472
Problem 4	477,649	169,720	343,502	5	0.0064	250
Problem 5	922,707	212,320	868,632	3	0.0098	0
Problem 6	365,532	271,606	99,730	6	0.0289	4504
Problem 7	1,234,825	289,389	1,087,554	4	0.0148	2831
Problem 8	888,024	351,010	583,274	6	0.0147	2992
Problem 9	2,405,870	425,846	2,131,479	7	0.0203	3172
Problem 10	889,989	489,836	441,929	5	0.0150	3155
Problem 11	902,871	540,905	442,327	3	0.0013	0
Problem 12	835,618	621,949	256,050	3	0.0096	986
Problem 13	1,554,134	662,422	1,084,839	3	0.0092	3872
Problem 14	1,699,642	726,679	1,189,341	3	0.0020	762
Problem 15	5,726,711	771,869	5,521,994	5	0.0208	10,168
Problem 16	1,495,514	888,459	735,868	3	0.0107	7939
Problem 17	1,872,152	932,262	1,147,962	3	0.0033	2370
Problem 18	101,626,317	1,015,117	65,535	1	0	0
Problem 19	3,054,732	1,158,626	2,016,031	8	0.0131	6669
Problem 20	1,185,397	1,168,536	22,055	2	0.0136	0

that, since there is no executive library for this problem, all data in this paper have been generated randomly. However, these data are chosen in such a way which mirrors the real condition of the company and can serve as proxy for real cases. Therefore, 20 samples are generated and six well-known metrics including: MID, Max Spread, SNS, NPS, RAS, and Spacing (see Coello et al. 2007) are calculated for each algorithm and sample (see Tables 6, 7, 8). After obtaining the results of metrics, we evaluate and rank the algorithms with MCDM model. Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) method is one of well-known multi-criteria decision making modeling (Akhavan et al. 2015).

In this research, the metrics are used as criteria and algorithms are considered as alternatives. The goal is to prioritize alternatives based on criteria and select the algorithm which has the best performance.

In this model, first of all, the average of six metrics (criteria) in all problems is calculated then this matrix is normalized. After calculating the normalized decision matrix, the Euclidean distance of alternatives from positive and negative ideal solutions (d_i^+, d_i^-) are calculated using Eqs. (15–16).

$$d_i^+ = \sqrt{\sum_{j=1}^{n} (v_{ij} - v_j^+)^2} \quad i = 1, 2, \ldots, m \tag{15}$$

$$d_i^- = \sqrt{\sum_{j=1}^{n} (v_{ij} - v_j^-)^2} \quad i = 1, 2, \ldots, m \tag{16}$$

Finally, relative closeness value for each alternative calculated using Eq. 17. The alternative which has larger relative closeness value should be selected as the best one.

$$Cl_i = \frac{d_i^-}{d_i^- + d_i^+} \quad i = 1, 2, \ldots, m \tag{17}$$

Table 7 The result metric for NRGA

	MID	Max spread	SNS	NPS	RAS	Spacing
Problem 1	8,188,090	81,755	65,535	1	0	0
Problem 2	3,141,348	61,820	4,353,686	2	0.017	0
Problem 3	2,654,742	104,720	2,943,465	4	0.019	61
Problem 4	1,908,975	170,280	1,842,959	9	0.009	922
Problem 5	5,337,288	211,018	5,916,110	4	0.016	1640
Problem 6	8,707,682	260,742	10,344,769	3	0.007	152
Problem 7	7,372,236	292,630	8,172,759	4	0.011	557
Problem 8	11,490,043	343,836	13,649,436	3	0.006	766
Problem 9	8,468,223	416,458	8,997,759	5	0.009	2104
Problem 10	9,741,373	485,164	10,342,254	5	0.017	2362
Problem 11	18,018,400	529,442	21,413,375	3	0.011	6501
Problem 12	14,945,207	596,653	16,567,010	4	0.001	249
Problem 13	31,992,291	638,213	44,339,500	2	0.001	0
Problem 14	18,291,967	725,449	20,279,436	4	0.006	2840
Problem 15	19,658,874	770,875	21,796,666	4	0.015	7084
Problem 16	13,764,059	822,929	14,172,228	6	0.005	4625
Problem 17	30,127,022	899,187	35,791,283	3	0.005	403
Problem 18	9,870,254	964,115	9,748,874	6	0.010	3563
Problem 19	36,327,763	1,073,741	43,165,207	3	0.012	0
Problem 20	22,699,998	1,129,400	24,111,918	5	0.005	1651

Table 8 The result metric for MOPSO

	MID	Max spread	SNS	NPS	RAS	Spacing
Problem 1	8,188,090	81,755	65,535	1	0	0
Problem 2	3,141,348	61,820	4,353,686	2	0.017	0
Problem 3	2,654,742	104,720	2,943,465	4	0.019	61
Problem 4	1,908,975	170,280	1,842,959	9	0.009	922
Problem 5	5,337,288	211,018	5,916,110	4	0.016	1640
Problem 6	8,707,682	260,742	10,344,769	3	0.007	152
Problem 7	7,372,236	292,630	8,172,759	4	0.011	557
Problem 8	11,490,043	343,836	13,649,436	3	0.006	766
Problem 9	8,468,223	416,458	8,997,759	5	0.009	2104
Problem 10	9,741,373	485,164	10,342,254	5	0.017	2362
Problem 11	18,018,400	529,442	21,413,375	3	0.011	6501
Problem 12	14,945,207	596,653	16,567,010	4	0.001	249
Problem 13	31,992,291	638,213	44,339,500	2	0.001	0
Problem 14	18,291,967	725,449	20,279,436	4	0.006	2840
Problem 15	19,658,874	770,875	21,796,666	4	0.015	7084
Problem 16	13,764,059	822,929	14,172,228	6	0.005	4625
Problem 17	30,127,022	899,187	35,791,283	3	0.005	403
Problem 18	9,870,254	964,115	9,748,874	6	0.010	3563
Problem 19	36,327,763	1,073,741	43,165,207	3	0.012	0
Problem 20	22,699,998	1,129,400	24,111,918	5	0.005	1651

The average decision matrix, normalized weighted decision matrix, Euclidean distance of alternatives, and relative closeness values of alternatives for MOPSO, NRGA, and NSGAII are shown in Table 9. The results show that MOPSO algorithm's performance in solving problems is better than others.

Table 9 The result of TOPSIS method for problem

	Average decision matrix						
	MID	Max spread	SNS	NPS	RAS	Spacing	MID
NSGA-II	6,997,770.5	547,278.8	1,197,042.5	4	0.01139	2507.1	0.032
NRGA	14,135,291.8	528,921.5	15,900,711.5	4	0.009131	1774.0	0.016
MOPSO	1,037,890.2	622,772.9	12,296.3	4.15	0.024065	2206.4	0.213

	Normalize decision matrix				
	Max spread	SNS	NPS	RAS	Spacing
NSGA-II	0.055	0.015	0.015	0.028	0.067
NRGA	0.054	0.193	0.015	0.035	0.095
MOPSO	0.063	0.000	0.016	0.013	0.076

	d_i^+	d_i^-	CL	Rank
NSGA-II	0.180	0.184	0.494	2
NRGA	0.011	0.276	0.038	3
MOPSO	0.276	0.030	0.902	1

Conclusion

This paper presented a Bi-objective model for scheduling a JIT single machine with a periodic preventive maintenance while total earliness-tardiness and makespan are minimized simultaneously. Furthermore, the proposed mixed integer model takes the maximum number of allowable jobs in each period which helps to maintain quality of the products.

To solve the model, we propose NSGA-II, NRGA, and MOPSO algorithms. The parameters of these algorithms are tuned by Taguchi method, and finally, six performance metrics are used to analyze the diversity and convergence of proposed algorithms. Based on MADM analysis of these metrics, we have shown that MOPSO has better metric performance to other algorithms and has better uniformity within the solutions of their Pareto curves.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

Akhavan P, Barak S, Maghsoudlou H, Antuchevičienė J (2015) FQSPM-SWOT for strategic alliance planning and partner selection; case study in a holding car manufacturer company. Technol Econ Dev Econ 21(2):165–185

Al Jadaan O, Rajamani L, Rao C (2008) Non-dominated ranked genetic algorithm for solving multiobjective optimization problems. J Theor Appl Inf Technol 15(5):60–67

Behnamian J (2014) A parallel competitive colonial algorithm for JIT flowshop scheduling. J Comput Sci 5(5):777–783

Behnamian J, Fatemi Ghomi SMT (2014) Multi-objective fuzzy multiprocessor flowshop scheduling. Appl Soft Comput 21:139–148

Benmansour R, Allaoui H, Artiba A, Hanafi S (2014) Minimizing the weighted sum of maximum earliness and maximum tardiness costs on a single machine with periodic preventive maintenance. Comput Oper Res 47:106–113

BSI (1984) Glossary of maintenance management terms in terotechnology, BS 3811

Chen JS (2006) Single-machine scheduling with flexible and periodic maintenance. J Oper Res Soc 57(6):703–710

Coello CAC, Pulido GT, Lechuga MS (2004) Handling multiple objectives with particle swarm optimization. Evolut Comput IEEE Trans 8(3):256–279

Coello CC, Lamont GB, Van Veldhuizen DA (2007) Evolutionary algorithms for solving multi-objective problems. Springer, Berlin

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. Evolut Comput IEEE Trans 6(2):182–197

Ebrahimy Zade A, Fakhrzad MB (2013) A dynamic genetic algorithm for solving a single machine scheduling problem with periodic maintenance. ISRN Ind Eng 2013:11

Eren T (2007) A multicriteria flowshop scheduling problem with setup times. J Mater Process Technol 186(1–3):60–65

Fazel Zarandi MH, Mosadegh H, Fattahi M (2013) Two-machine robotic cell scheduling problem with sequence-dependent setup times. Comput Oper Res 40(5):1420–1434

Gao J (2010) A novel artificial immune system for solving multiobjective scheduling problems subject to special process constraint. Comput Ind Eng 58(4):602–609

Gao KZ, Suganthan PN, Pan QK, Chua TJ, Cai TX, Chong CS (2014) Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling. Inf Sci 289:76–90

Hsu C-J, Low C, Su C-T (2010) A single-machine scheduling problem with maintenance activities to minimize makespan. Appl Math Comput 215(11):3929–3935

Liao CJ, Chen WJ (2003) Single-machine scheduling with periodic maintenance and nonresumable jobs. Comput Oper Res 30(9):1335–1347

Liao CJ, Chen CM, Lin CH (2007) Minimizing makespan for two parallel machines with job limit on each availability interval. J Oper Res Soc 58(7):938–947

Low C, Ji M, Hsu C-J, Su C-T (2010) Minimizing the makespan in a single machine scheduling problems with flexible and periodic maintenance. Appl Math Model 34(2):334–342

Luo W, Cheng TCE, Ji M (2015) Single-machine scheduling with a variable maintenance activity. Comput Ind Eng 79:168–174

Qi X, Chen T, Tu F (1999) Scheduling the maintenance on a single machine. J Oper Res Soc 50(10):1071–1078

Sadeghi J, Sadeghi S, Niaki STA (2014) A hybrid vendor managed inventory and redundancy allocation optimization problem in supply chain management: an NSGA-II with tuned parameters. Comput Oper Res 41:53–64

Salameh MK, Ghattas RE (2001) Optimal just-in-time buffer inventory for regular preventive maintenance. Int J Prod Econ 74(1–3):157–161

Taguchi G, Chowdhury S, Wu Y (2005) Taguchi's quality engineering handbook. Wiley, New York

Wan L (2014) Scheduling jobs and a variable maintenance on a single machine with common due-date assignment. Sci World J 2014:5

Xu D, Wan L, Liu A, Yang D-L (2015) Single machine total completion time scheduling problem with workload-dependent maintenance duration. Omega 52:101–106

Yang D-L, Hung C-L, Hsu C-J, Chern M-S (2002) Minimizing the makespan in a single machine scheduling problem with a flexible maintenance. J Chin Inst Ind Eng 19(1):63–66

Yang D-L, Hsu C-J, Kuo W-H (2008) A two-machine flowshop scheduling problem with a separated maintenance constraint. Comput Oper Res 35(3):876–883

Yousefi M, Omid M, Rafiee S, Ghaderi S (2013) Strategic planning for minimizing CO_2 emissions using LP model based on forecasted energy demand by PSO Algorithm and ANN. J Homepage www. IJEE.IEEFoundation.org 4(6):1041–1052