Int J Adv Manuf Technol
DOI 10.1007/s00170-010-2932-8

ORIGINAL ARTICLE

A hybrid algorithm based on particle swarm optimization and simulated annealing for a periodic job shop scheduling problem

Amin Jamili · Mohammad Ali Shafia · Reza Tavakkoli-Moghaddam

Received: 15 September 2009 / Accepted: 6 September 2010

Abstract Generating schedules such that all operations are repeated every constant period of time is as important as generating schedules with minimum delays in all cases where a known discipline is desired or obligated by stakeholders. In this paper, a periodic job shop scheduling problem (PJSSP) based on the periodic event scheduling problem (PESP) is presented, which deviates from the cyclic scheduling. The PESP schedules a number of recurring events as such that each pair of event fulfills certain constraints during a given fixed time period. To solve such a hard PJSS problem, we propose a hybrid algorithm, namely PSO-SA, based on particle swarm optimization (PSO) and simulated annealing (SA) algorithms. To evaluate this proposed PSO-SA, we carry out some randomly constructed instances by which the related results are compared with the proposed SA and PSO algorithms as well as a branch-and-bound algorithm. In addition, we compare the results with a hybrid algorithm embedded with electromagnetic-like mechanism and SA. Moreover, three lower bounds (LBs) are studied, and the gap between the found LBs and the best found solutions are reported. The outcomes prove that the proposed hybrid algorithm is an efficient and effective tool to solve the PJSSP.

A. Jamili (✉) · M. A. Shafia
Department of Industrial Engineering,
Iran University of Science and Technology,
Tehran, Iran
e-mail: a_jamili@iust.ac.ir

R. Tavakkoli-Moghaddam
Department of Industrial Engineering, College of Engineering,
University of Tehran,
Tehran, Iran

Keywords Periodic job shop scheduling · Periodic event scheduling problem · Particle swarm optimization · Simulated annealing

1 Introduction

The job shop scheduling problem (JSSP) is the most challenging one which has been the subject of many research studies during the recent decades. The problem is described simply as follows: given n jobs to be processed on m machines. Each consists of a predetermined sequence of task operations, each of which requires processing without interruption for a given period of time on a given machine. Tasks of the same job cannot be processed simultaneously and each job must at last meet each machine only once to complete its processing. A schedule is an assignment of operations to time slots on a machine. The objective is to find the sequence of all the jobs in such a way that an established criterion is optimized. The researchers have considered various objectives like minimizing the makespan, maximum tardiness/tardiness, total (weighted) completion time, and total (weighted) tardiness/tardiness. Since JSSP is not tractable to be solved especially in large-scale problems by conventional optimization techniques, heuristic algorithms are highly concentrated on in the literature. Amongst all solving methods, the use of meta-heuristic methods is well experienced. For example, Guo et al. [1] studied a multi-objective order scheduling problem where processing time, orders, and arrival times are considered uncertain. They presented a mathematical model and proposed a genetic algorithm to solve the given problem. Guo et al. [2] investigated the mathematical model for a scheduling problem in the flexible assembly line with parallel

Published online: 21 September 2010

machines and flexible operation assignment. To solve this problem, they developed a bi-level genetic algorithm. Gao et al. [3] proposed a new parallel genetic algorithm based on the vector group encoding method and the immune method to solve a multi-objective job shop scheduling problem. In addition, hybridization of algorithms can be considered as a way to develop a more effective and efficient searching strategy to overcome the weaknesses of a pure single algorithm. Zhang et al. [4] as an example, has introduced a hybrid algorithm combining a genetic algorithm with local search for the JSSP and used a new procedure to further reduce the search space. Naderi et al. [5] solved an extended job shop scheduling problem by the artificial immune algorithm hybridized with a simple and fast simulated annealing (SA). They considered sequence-dependent setup times and preventive maintenance operations minimizing the total completion time.

The convergence speed of evolutionary algorithms to the globally (or nearly globally) optimal results is better than that of traditional techniques. Therefore, evolutionary algorithms, such as genetic algorithms (GA), differential evolution algorithm, ant colony algorithm, immune algorithm, and particle swarm optimization algorithm (PSO) have been used to improve the solution of optimization problems further. Recently, PSO has accelerated gaining attention and applications by a number of researchers. It is a population-based searching technique which has a high search efficiency by combining local search (by self experience) and global one (by neighboring experience), whose development is based on the observations of social behavior of animals, such as bird flocking, fish schooling, and swarm theory. PSO embodies some attractive characteristics: (1) compared with other evolutionary algorithms (e.g., GA), it possesses memory (i.e., the characteristics of the good solutions are retained by all particles), whereas in GA, the previous characteristics of the problem are lost once the population alters. (2) PSO has constructive cooperation amongst the particles (i.e., particles in the swarm share their information). From the other point of view, similar to evolutionary algorithms, PSO is often easy to be a premature convergence so that exploration (i.e., searching for promising solutions within the entire region) and exploitation (i.e., searching for improved solutions in sub-regions) should be enhanced and well balanced to achieve better performance. On the other hand, SA is a stochastic searching algorithm with a jumping property (i.e., a worse solution has a probability to be accepted as the new solution). Thus, the authors of this paper employ the jumping mechanism of SA into PSO in order to achieve the results with higher quality.

Sha and Lin [6] constructed PSO for a multi-objective job shop scheduling problem that minimizes makespan, total tardiness, and total machine idle times. Xia and Wu [7] introduced a hybrid algorithm, the so-called HPSO, based on PSO and SA algorithms to solve the JSSP. The role of the PSO algorithm is limited to find an initial solution for SA during the hybrid search process. Such a hybrid algorithm can be converted to the general PSO by omitting the SA unit, and it can be converted to the traditional SA by setting the swarm size to one particle. The results showed that the PSO-based algorithm is a viable and effective approach for the JSSP. It is worth noting that as it is explained in the following sections, the PSO-SA algorithm proposed in this paper is benefited from the SA algorithm by modifying all swarms in each run. PSO has also been applied in other optimization problems. For instance, Mehdizadeh et al. [8] proposed a hybrid PSO and fuzzy c-means clustering algorithm for optimizing the fuzzy clustering criteria. In addition to the PSO algorithm, there have been many efforts conducted in applying other well-known approaches. Eswaramurthy and Tamilarasi [9] have introduced a hybrid tabu search and ant colony algorithms for classical job shop scheduling problems. Zhang et al. [4] and Wang et al. [10] applied genetic algorithm for JSSP. Pan and Huang [11] proposed a hybrid genetic algorithm to solve the no-wait job shop problems which utilizes an asymmetric traveling salesman problem formulation. Roshanaei et al. [12] have studied the electromagnetism like mechanism (EM) algorithm for the scheduling job shop problem with sequence-dependent setup times. The JSSP is classified as one of the most challenging NP-complete problems [13].

In the present paper, a new formulation for a periodic JSSP is presented where a definite number of jobs are to be scheduled repeatedly in a fixed time frame. The proposed formulation is based on the periodic event scheduling problem (PESP) introduced by Serafini and Ukovich [14]. It is worth noting that the studied periodic scheduling problem is somehow different from the known cyclic scheduling problem, which is deeply studied in the literature. To illustrate the issue, before introducing the PESP, the cyclic scheduling (CS) problem is shortly reviewed.

In this problem, a set of activities are to be repeated an indefinite number of times, and it is desired that the sequence be also repeated. Cyclic scheduling problems arise in domains, such as automated manufacturing systems, time-sharing of processors in embedded systems and in compilers for scheduling loop operations for parallel or pipelined architectures. The primary objective is to minimize the period length. The CS problem is generally decomposed into two categories, with and without resource constraints. The basic cyclic scheduling (BCS) problem falls in the second category. Brucker and Kampmeyer [15] have presented a tabu search method for the BCS problem. They also introduced a general basic cyclic scheduling

problem that is to minimize the cycle time [16]. On the other hand, the cyclic job shop scheduling problem (CJSSP) belongs to the first category (i.e., with resource constraints). Chrétienne [17] studied the BCS with deadlines. Munier [18] has considered the BCS with linear precedence constraints known as the BCSL problem.

Kimbrel and Sviridenko [19] introduced a high-multiplicity CJSSP in which the jobs are all the same and the schedule should process them in a cyclic fashion. They considered bi-objectives, namely the cycle time and the flow time. Cavory et al. [20] focused on a CJSSP with linear precedence constraints as the main topic and presented a general approach based on the coupling of a genetic algorithm and a scheduler. Song and Lee [21] investigated the scheduling problem for general cyclic job shop with blocking where each machine has an input buffer of the finite capacity. They also developed Petri net models for the shops. In contrast with CS, the PESP is used for constant period lengths. In practice, there are many applications where the time frame should be constant, and violating the period length results in more charges. Some of the main applications of periodic scheduling with fixed period length are crew scheduling, train timetabling in railway networks, bus public transportation, and aircraft. Moreover, in the industrial engineering environment, particularly in manufacturing systems, providing the conditions where all jobs follow a known discipline results in the simplification of the tracking process which is desired by the managers. Periodic job shop scheduling problem (PJSSP) can also be used when one intends to schedule a definite number of jobs in each single working shift wherein all shifts consist of similar operations as well as the same scheduling. Along with CS, some papers consider the job shop scheduling in a periodic fashion; however, the introduced definition for the PJSSP is like the CS problem where the objective is to minimize the cycle time. In this regard, Tohme et al. [22], as an example, have exhibited an evolutionary computation approach to the PJSSP in which a Petri net model of a periodic job shop system is generated automatically and evolutionarily. The objective function of the problem is to obtain the shortest schedule. Song and Lee [23], as another example, have discussed a sequencing problem that finds the processing order at each machine which minimizes the cycle time, and they have solved the problem by tabu search. Lee and Posner [24] studied the PJSSP that minimizes the cycle time and maximizes makespan simultaneously.

To clarify the outcomes of applying the PJSSP instead of the classical JSSP, consider an example to schedule eight jobs on four machines. Each pair of job {1, 5}, {2, 6}, {3, 7}, and {4, 8} has the same characteristics. The release times for jobs 1, 2, 3 and 4 are 0 where the others are equal to the period length, say T=60 min. The due dates are equal to 0. Figures 1 and 2 depict the optimum schedules for the classical and periodic cases, respectively.

Comparing the solutions of classical and periodic JSSP, one can find that (1) both solutions are equal considering the total tardiness objective, (2) the periodic solution provides conditions where all operations can be controlled easier where every 60 min, jobs with the same characteristics start and end at the same time (e.g., the operation of jobs 4 and 8 on machine 4 starts at T+27 and ends at T+2), and (3) the periodic solution can easily extend to more periods without affecting the optimum solution. Figure 3 shows the extension of the periodic solution of Fig. 2 to three periods.

The novelty of this paper is summarized as follows:

- Modeling a new formulation for a periodic JSSP
- Applying a new hybrid algorithm based on PSO and SA with a new hybridization method
- Applying a branch-and-bound (B&B) algorithm for the given problem to validate the proposed hybrid algorithm
- Introducing two new lower bound generation methods to validate the proposed hybrid algorithm

The content of this paper is organized as follows: In Section 2, after providing a brief explanation of the PESP, the periodic JSSP is formulated. Section 3 presents the proposed particle swarm optimization (PSO) and simulated annealing (SA) algorithms, as well as the hybridization of these algorithms. The computational results are presented in Section 4, and the conclusion remarks are given ultimately to pin point the contribution of this paper.

2 Periodic job shop scheduling problem

Serafini and Ukovich [14] have introduced the PESP. This problem is to schedule a number of recurring events, such that each pair of event fulfills certain constraints. Given a time period T, a set of V events, and a set of constraints A, every constraint $a=(i, j)$ considers a pair of events (i, j) and defines a lower bound l_a and an upper one u_a.

It is possible to encode the set of periodic interval constraints imposed on the schedule in an event–activity graph $D=(V, A)$ with node set V and arc set A which represent events and activities, respectively.

A solution of a PESP instance is a node assignment π: $V \rightarrow [0, T)$ which satisfies inequality (1) by:

$$(\pi_j - \pi_i - l_a) mod\ T \leq u_a - l_a, \forall a = (i,j) \in A, \quad (1)$$

Where, π_i is the occurrence time of event i. It is worth noting that one can scale an instance such that $0 \leq l_a < T$ and for the span $d_a = u_a - l_a$, with $d_a < T$.

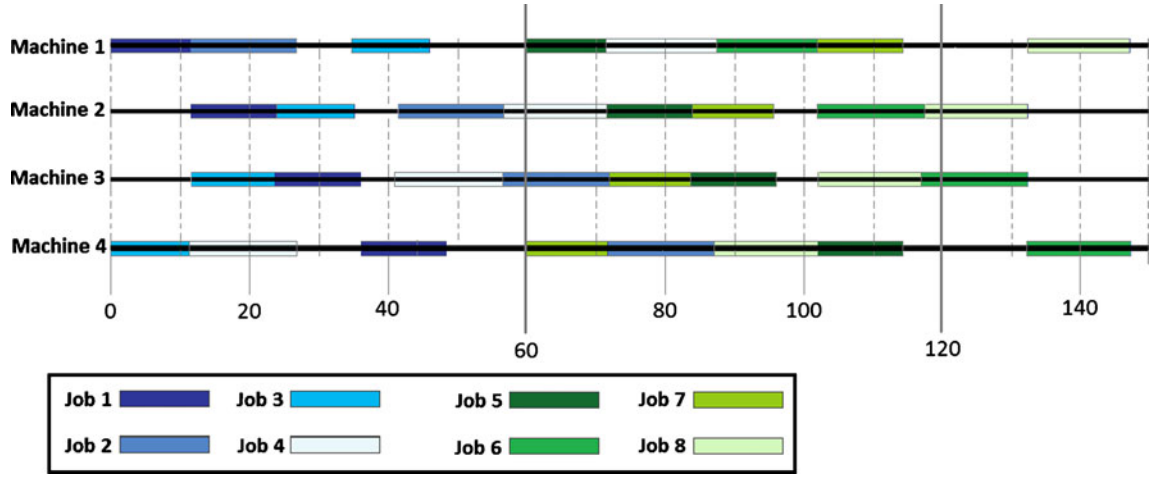

Fig. 1 Classical optimum schedule

Preposition 1 To apply integer programming techniques, the following reformulation of the problem can be utilized:

$$l_a \le \pi_j - \pi_i - T \times z_a \le u_a \tag{2}$$

Where, $z_a \in Z$, and is called a periodical offset of the activity a.

Proof Based on the definition of mod operator, we have:

$$(\pi_j - \pi_i - l_a) mod\ T = (\pi_j - \pi_i - l_a) - T \times max\{z \in Z | (\pi_j - \pi_i - l_a) - Tz \ge 0\}$$

Inequality 1 can be rewritten as follows:

$$0 \le (\pi_j - \pi_i - l_a) - T \times z_a \le u_a - l_a, \forall a = (i,j) \in A,$$

such that $z_a = max\{z \in Z | (\pi_j - \pi_i - l_a) - Tz \ge 0\}$.

Where the uniqueness of z_a follows from our assumption that $u_a - l_a < T$.

The proof follows immediately from adding l_a to the above inequality. For more details, the reader may refer to [25].

As an example of a PESP, with three events, three constraints and time period T=60, consider the constraint graph shown in Fig. 4. The problem is to find π_i and z_i, $\forall i$= 0,1,2, such that:

$$6 \le \pi_1 - \pi_0 - 60z_0 \le 18,\ -8 \le \pi_1 - \pi_2 - 60z_1 \le 9,\ -48 \le \pi_0 - \pi_2 - 60z_2 \le 30$$

For more details, please refer to Kinder [26]. Liebchen and Peeters [27], Odijk [28], Nachtigall [29], and Serafini and Ukovich [14] showed that the PESP is NP-completeness for fixed $T \ge 3$.

In the PJSSP, events are defined as operations. Therefore, to reach the formulation of the PJSSP, the constraints involved with the scheduling of operations should be

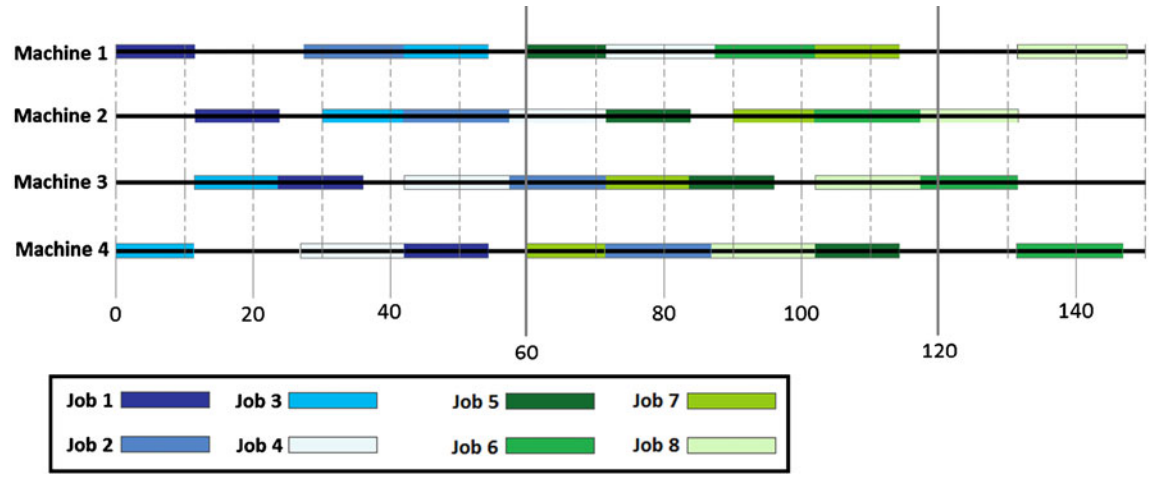

Fig. 2 Periodic optimum schedule

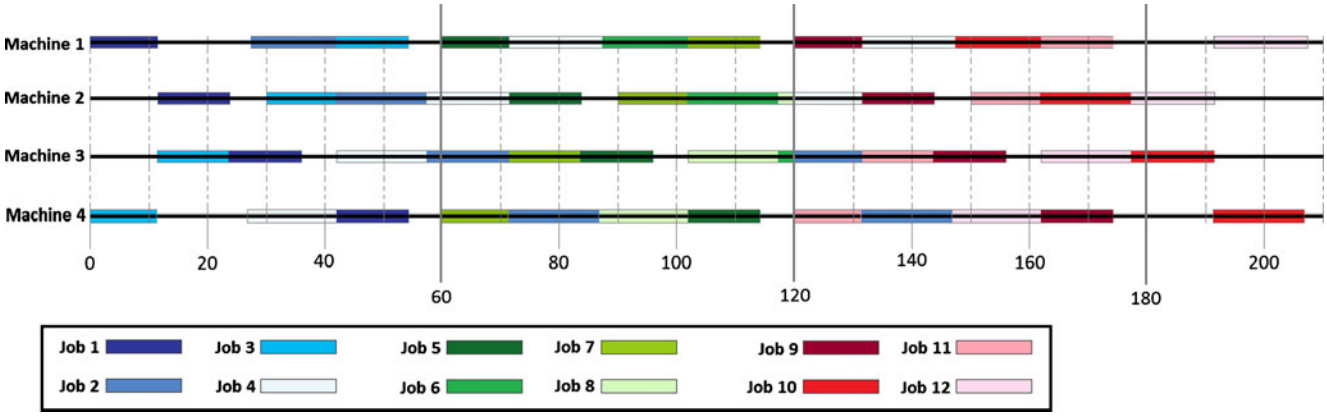

Fig. 3 Extension of the periodic optimum schedule

modified based on the PESP approach. To formulate the PJSSP, the following notations are used in the mathematical model.

M Set of machines
P Set of jobs
e_i The last machine which is met by job i
t_{ijm} Required time to process operation j of job i on machine m
T Time period
x_{ik} Completion time of job i on machine k

The mathematical model of the PJSSP, called Model P1, is as follows:

$$\min \sum_{i \in P} x_{ie_i}$$

$$t_{iuk} \le x_{ik} - x_{ih} \le T - 1, \ (i,u-1,h) << (i,u,k), \ \forall i \in P \ \forall k,h \in M \quad (3)$$

$$t_{iuk} \le x_{ik} - x_{jk} - z_{ijk} \times T \le T - t_{ju'k}, \quad \forall i,j \in P \ \forall k \in M \quad (4)$$

where, Z_{ijk} are integer variables ($\forall i, j \in P$; $\forall k \in M$).

The objective function of the above model (P1) is to minimize the completion time of jobs. Constraint 3 represents the precedence of each job, where the uth operation of job i should be performed on machine k. Moreover, $(i,u-1,h) << (i,u,k)$ shows the routings of jobs and expresses that job i visits machine k immediately after finishing the process on machine h. Constraint 4 assures that no two operations are processed simultaneously by the same machine.

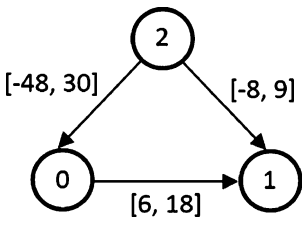

Fig. 4 A small PESP instance

Remark 1 The presented objective function can be considered as the so-called total weighted tardiness (TWT) where all weights of jobs are equal and the due dates are equal to zero. Furthermore, the release times are all supposed to be equal to zero.

3 Proposed hybrid algorithm

This section describes the proposed hybrid algorithm consisting of particle swarm optimization (PSO) and simulated annealing (SA) algorithms for the PJSSP. This problem is very complex in nature and difficult to solve large-scale problems by optimization techniques. In this case, it is well experienced that meta-heuristic algorithms can often outperform conventional optimization methods when applied to difficult real-world problems. An encoding scheme is first presented in order to generate schedules. Then SA and PSO are separately reviewed and designed. Ultimately, the hybridization procedure of these algorithms is explained.

3.1 Encoding scheme

To represent the candidate schedules, random keys (RKs) are selected as the encoding scheme, which is well experienced and is easy to adjust to the PSO and SA algorithms [30–35].

In this paper, the permutation of jobs is shown through random keys. Each job has a random number between 0 and 1, and these random keys show the relative order of the

jobs. For example, consider a problem with three jobs and two machines. Each job consists of two operations and is thereby repeated twice so that for this problem there are six operations on hand {1 1 2 2 3 3}. For each operation, a random number is randomly generated from a uniform distribution between 0 and 1, as shown in Table 1. These RKs are then sorted to find a relative order of operations, as illustrated in Table 2.

3.2 Simulated annealing

Simulated annealing (SA) is one of the most popular meta-heuristics providing a means to escape local optima by considering moves which worsen the objective function value known as jumping mechanism. Towards the end of computation, when the *temperature* or probability of accepting a worse solution, is nearly zero, this simply seeks the bottom of the local optima. The chance of getting a good solution can be traded off with the computational time by slowing down the cooling schedule. One can express that the slower the cooling, the higher the chance of finding the optimum solution, but the longer the run time.

The notations used in the proposed SA are as follows:

K	Iteration counter
T_k	Temperature in the kth iteration
s_k	kth schedule
T_0	Initial temperature
s_{best}	Best found solution
$f(s)$	Objective function value for schedule s
α	Cooling factor
$P_{\text{accept}}(s,s',T)$	Probability function to accept non-improving solution s'

The main steps of the proposed SA algorithm for the PJSSP are as follows:

Step 1 Set T_0 and α. Let $k \leftarrow 0$.
Step 2 Select an initial solution, s. Let $s_{\text{best}} \leftarrow s$.
Step 3 If s is infeasible, go to Step 2; otherwise, go to Step 4.
Step 4 Generate a neighborhood solution, s', using schedule s.
Step 5 If s' is infeasible go to Step 4; otherwise, go to Step 6.
Step 6 If $f(s') \le f(s)$ or random[0, 1] $< P_{\text{accept}}$ then $s \leftarrow s'$. If $f(s') \le f(s_{\text{best}})$ then $s_{\text{best}} \leftarrow s$.

Table 1 Representation of a schedule using the non-sorted random key scheme

Random key	0.45	0.67	0.92	0.13	0.89	0.21
Operations	1	1	2	2	3	3

Table 2 Representation of a schedule using the sorted random key scheme

Sort (random key)	0.13	0.21	0.45	0.66	0.89	0.92
Operations	2	3	1	1	3	2

Step 7 If the termination criterion is not satisfied then $T_k = \alpha \times T_{k-1}$, $k \leftarrow k+1$ and go to Step 4; else, stop and return s_{best}.

where,

$$P_{\text{accept}}(s, s', T_k) = \begin{cases} 1 & \text{if } f(s') < f(s) \\ \exp\left(\frac{f(s)-f(s')}{T_k}\right) & \text{otherwise} \end{cases} \quad (5)$$

3.2.1 Feasible solution

Schedules are generated based on the orders which are given to the operations. This method always leads to a feasible solution in classical but not in periodic job shop scheduling.

Remark 2 If $\exists k \in M$, such that $\sum_{i \in P} t_{ijk} > T$, where the jth operation of job i should be performed on machine k, there exists no feasible solution for the given problem.

Remark 3 For each machine, all assigned operations should be performed in a time interval less than period length. Figure 5 depicts an infeasible PJSSP in order to clarify the issue. It is intended to schedule five operations in the period length of T in a machine. As shown in this figure, the first time allocation is infeasible because there is no interval to assign job 5 in the period length, while the second one is feasible.

Remark 4 For each instance, if one assumes a very big value for the period length, the PJSSP will change to a classical JSSP.

As a result of this remark, any heuristic algorithm is capable of finding feasible solutions by increasing the period length.

3.2.2 Initial solution generation

To obtain the initial solution, three approaches are employed in this paper:

1. Based on the first-leave first-served (FLFS) rule. The procedure is as follows: When two or more jobs compete for the same machine, the precedence is given to the one which leaves the machine first.

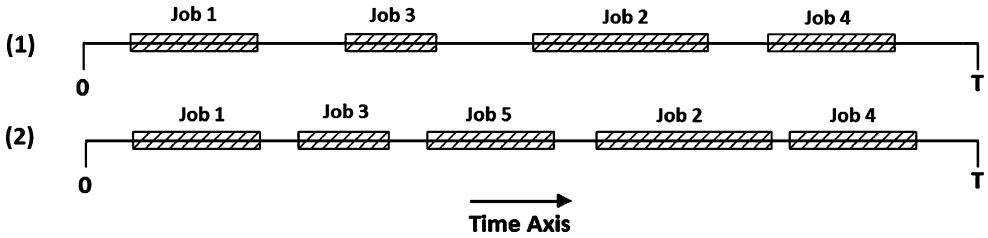

Fig. 5 Infeasible PJSSP

2. Based on first-in first-out (FIFO) rule. The procedure is as follows: When two or more jobs compete for the same machine, the precedence is given to the one which waits more.
3. Based on generating random keys.

The idea behind defining three approaches is the fact that there exists no assurance to find a feasible solution using the FLFS and FIFO rules.

3.2.3 Neighborhood solution generation

To generate a neighborhood candidate, the single point operator method is employed. In this method, the RK of one randomly selected job from schedule s is randomly regenerated.

3.2.4 Parameter tuning

It is well-known that the quality of algorithms is significantly influenced by the values of parameters. Those of the proposed SA are limited to T_0 and α. To tune them, a full factorial design in the design of experiment (DOE) approach is applied. As it is shown in Table 3, three levels for the parameters are considered, and therefore a 3^2 design is applied. Moreover, 15 different instances are randomly generated and solved by assuming each of nine different combinations of (T_0, α). The stopping criterion is to perform the proposed SA by 3,000 iterations.

The relative deviation index (RDI) is used for the objective function value (OFV) of the given problem as a common performance measure to compare the instances. This index is obtained by:

$$\mathrm{RDI}_k = \frac{F_k - \mathrm{Min}_k}{\mathrm{Max}_k - \mathrm{Min}_k} \times 100 \tag{6}$$

where, F_k is the OFV obtained for the kth instance. Min_k and Max_k are the best and worst solutions obtained for each instance.

Table 3 Three levels of SA parameters

	Level 1	Level 2	Level 3
T_0	100	200	300
α	0.9	0.94	0.98

Fifteen instances are randomly generated in different combinations of the number of jobs and the number of machines. Each instance is solved considering one of the combinations of T_0 and α. Therefore, (15×3×3=) 135 instances are totally solved. The related results are analyzed by means of the analysis of variance technique. The normality and homogeneity of variance and independence of residuals do not show any particular pattern in the experiments. Figure 6 depicts the interaction plot for parameters T_0 and α. It is concluded that the combination T_0=150 and α=0.97 results in a statistically better output than other evaluated combinations.

3.3 Particle swarm optimization

Particle swarm optimization (PSO), which was first developed by Kennedy and Eberhart [36], is an evolutionary algorithm that is initialized with a population of random candidate solutions known as particles. Similar to a bird that flies to the food, one particle moves its position to a better solution with a velocity which is dynamically adjusted according to its own flying experience and its companions' flying experience. Each particle is updated iteratively by:

$$v_t = w \times v_t + c_1 \times \mathrm{Random}\,[0,1] \times \left(P_l^t - s^t\right) + c_2 \times \mathrm{Random}\,[0,1] \times \left(P_g - s^t\right) \tag{7}$$

$$s^t = s^t + v_t \tag{8}$$

where, s^t is the tth particle; v_t is the rate of the position change for x_t. P_l^t is the best local solution that the tth

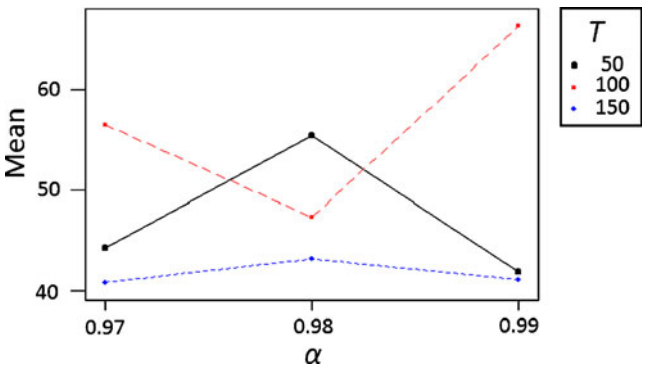

Fig. 6 Interaction between SA parameters

particle has achieved; P_g is the best solution obtained in the swarm; w, c_1 and c_2, are positive constants which represent the weight of previous velocity, the weight of the stochastic acceleration terms that pull each particle toward P_l^t and P_g, respectively.

The proposed PSO algorithm is applied for PJSSP.

Step 1 Generate *pop* initial schedules: s^t, $t=1,\ldots$, pop. If each of the generated schedules is infeasible regenerate it until all t schedules are feasible. Find the objective value for each particle. Update P_l^t and P_g.

Step 2 Update the position and velocity of the particles according to Eqs. 7 and 8.

Step 3 If each of the updated schedules is infeasible set $s^t \leftarrow P_l^t$.

Step 4 Find the objective value for each particle. Update P_l^t and P_g. If termination criterion is not met, then go to Step 2; otherwise, return P_g.

Where, pop is referred as the population number. The procedure to find the initial and neighborhood solutions are similar to the proposed SA algorithm. The parameters of the proposed PSO are tuned in the following subsection.

3.3.1 Parameter tuning

The parameters of PSO which must be tuned are pop, w, c_1, and c_2. Similar to the proposed SA, a full factorial design in the DOE approach is applied. As illustrated in Table 4, two levels for the parameters are considered, so a 2^4 design should be performed. Moreover, 15 different instances are randomly generated and solved by assuming each of the 16 different combinations of (pop, w, c_1, c_2). The stopping criterion is to perform $\frac{10{,}000}{\text{pop}}$ iterations.

The RDI also used for the OFV as the performance measure to compare the instances. As illustrated in Fig. 7, the interaction plot for the PSO parameters does not demonstrate any significant interactions between parameters. Figure 8 depicts the main effects of the investigated parameters.

It is concluded that the combination of w=0.65, c_1= 0.65, c_2=0.35, and pop=20, results in a statistically better output than other evaluated combinations.

Table 4 Two levels of PSO parameters

	Level 1	Level 2
Pop	10	20
W	0.35	0.65
C_1	0.35	0.65
C_2	0.35	0.65

3.4 Hybrid PSO-SA algorithm

The idea of the proposed hybrid algorithm, called PSO-SA, based on PSO and SA algorithms for job shop problems, has been widely exploited in the literature [6, 7, 34, 35]. PSO possesses high search efficiency by combining local search (by self experience) and global search (by neighboring experience). Moreover, SA is meta-heuristic, that is, designed for finding a near optimal solution of combinatorial optimization problems. Therefore, the PSO and SA algorithms are combined which can omit the concrete velocity–displacement updating method in the traditional PSO for the PJSS problem.

The proposed hybrid algorithm includes two phases: (1) the initial solutions are randomly generated and (2) the PSO algorithm combined with the SA algorithm is run. The general outline of the hybrid algorithm is summarized as follows:

Step 1 Generate $t=1$, …, pop initial random solutions. If any of the generated schedules is infeasible, reconstruct them until all the initial solutions are feasible. Set $P_l^t \leftarrow s^t$ and update P_g.

Step 2 For each particle of swarm, run the SA algorithm. In each iteration, if the new schedule is infeasible set $s^t \leftarrow P_l^t$, update P_l^t and P_g.

Step 3 Update the position and velocity of the particles according to Eqs. 7 and 8.

Step 4 If each of the updated schedules is infeasible, set $s^t \leftarrow P_l^t$.

Step 5 Find the objective value for each particle. Update P_l^t and P_g. If termination criterion is not met, go to Step 2; otherwise, return P_g.

The general outline of the hybrid algorithm is summarized in Fig. 9.

3.4.1 Parameter tuning

As explained in the proposed steps of PSO-SA algorithm, there remain two more parameters which are to be tuned, i.e., pop and SA iteration. The latter refers to the number of

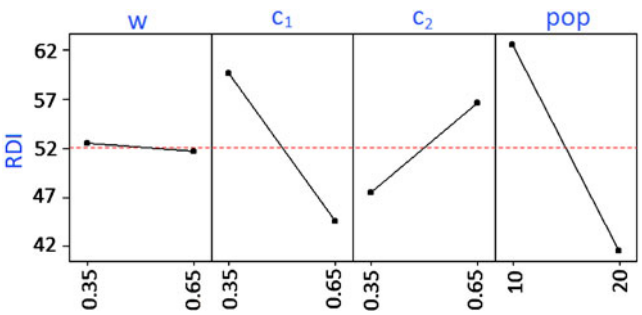

Fig. 7 The interaction among PSO parameters

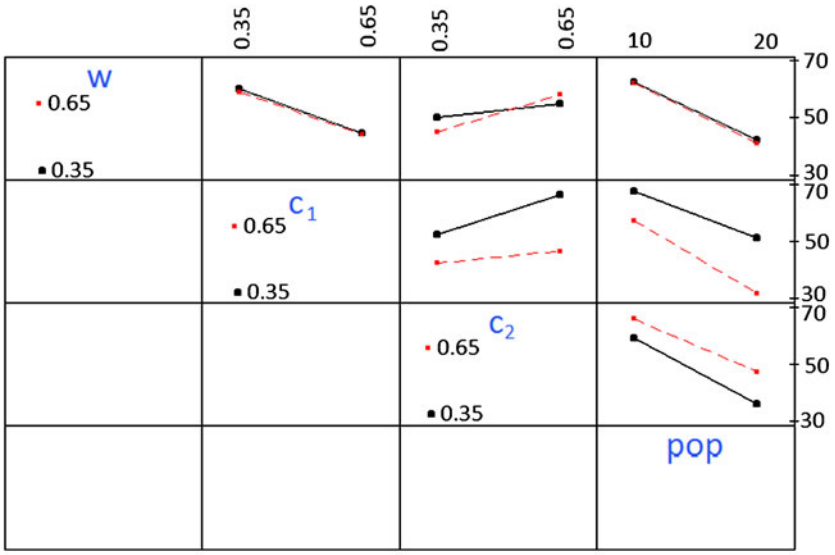

Fig. 8 The main effects of PSO parameters

times the SA algorithm is to be run in each step of the PSO-SA algorithm. To that end, a full factorial design in the DOE method is applied. As presented in Table 5, three levels for the parameters are considered, so a 3^2 design should be performed. Similar to previous sections, 15 different instances are randomly generated and solved by assuming each of the nine different combinations of (pop, SA iteration). The stopping criterion is to perform $\frac{30{,}000}{\text{pop}\times\text{SA iteration}}$ iterations. For the levels of the SA iteration parameter, we set a fixed initial temperature to 150 and the cooling factor to 0.73, 0.85, and 0.90 for levels 20, 40, and 60, respectively.

The RDI also used for the OFV as the performance measure to compare the instances. The interaction plot is depicted in Fig. 10.

It is concluded that the combination of pop=30 and SA iteration=20 results in statistically better output than other evaluated combinations.

4 Algorithm validation

Since there is no guarantee that the PSO-SA algorithm leads to an optimal solution in order to validate the proposed algorithm, a B&B algorithm and some efficient lower bounds are provided in the following subsections.

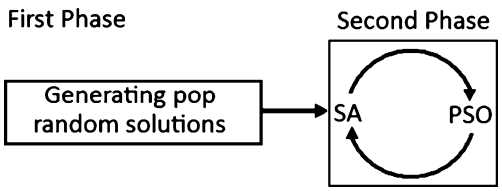

Fig. 9 General outline of the hybrid algorithm

4.1 Proposed B&B algorithm

The proposed B&B algorithm is relied on generating all the active schedules. A feasible schedule is called active if no operation can be completed earlier by altering the processing sequence on machines and not delaying any other operation. The employed notations are as follows:

Ω_v	Set of all operations of whose predecessors have already been scheduled in node ν
(i, j)	Operation of job i on machine j
r_{ij}^v	Earliest possible starting time of operation $(i, j) \in \Omega_v$
θ_v	Set of all scheduled operations in node ν
d_{ij}	Processing time of operation (i, j)
$P(\Omega_v)$	Earliest possible completion time of all operations belonging to Ω_v
Ω'_v	A subset of Ω_v

The main steps of the proposed algorithm for generating all active schedules are as follows:

Step 1) (Initial condition)

$v \leftarrow 0$
$\theta_v \leftarrow \emptyset$
$\Omega_v \leftarrow$ {First operation of each job}
$r_{ij}^v \leftarrow 0$ for all $(i, j) \in \Omega_v$

Table 5 Three levels of hybrid PSO-SA parameters

	Level 1	Level 2	Level 3
Pop	10	20	30
SA iteration	20	40	60

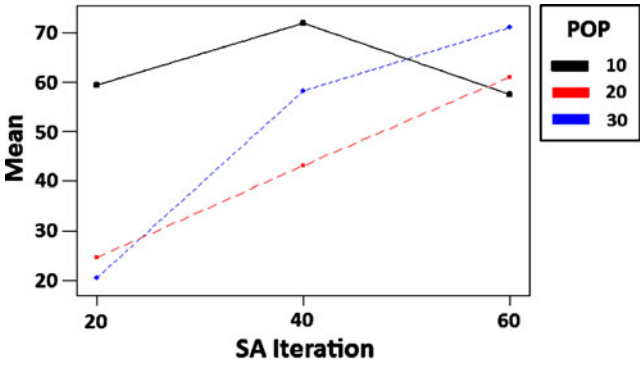

Fig. 10 Interaction plot for the hybrid SA-PSO parameters

Step 2) (Machine selection)

$$P(\Omega_v) \leftarrow \min_{(i,j)\in\Omega_v}\left\{r_{ij}^{v} + d_{ij}\right\}$$

$j^{*} \leftarrow$ the machine on which the minimum is achieved.

Step 3) (Branching)

$$\Omega'_v \leftarrow \left\{\left(i,j^{*}\right)\middle| r_{ij^{*}}^{v} < P(\Omega_v)\right\}$$

Sub step 3-1) For all $(i, j^{*}) \in \Omega'_v$ generate a new branch.

$$\Omega_v = \Omega_v - \left(i,j^{*}\right)$$

Add job successor of (i, j^{*}) to Ω_v.

$$\theta_v = \theta_v \cup \left(i, j^{*}\right)$$

Sub step 3-2) For all $(i, j^{*}) \in \Omega'_v$, if $(i, j^{*}) > T$ then for all $(i', j^{*}) \in \theta_v$, if there exists a conflict between (i, j^{*}) and (i', j^{*}) then generate a new branch. Let $(i, j^{*}) \in \theta_v$ and eliminate (i', j^{*}) and all its successors from θ_v.

Eliminate any repetitive branches.
$v \leftarrow v+1$ and go to Step 2.

The nodes of the branching tree are corresponding to the partial schedules. Step 3 branches from the node corresponding to the current partial schedule. The lower bound (LB) in each node equals the sum of the total delays plus the sum of all processing times.

4.2 Lower bound generation

Let S denote the set of all feasible schedules, $s \in S$, of a given job shop scheduling problem, P, where the objective function is $f(s)$. Furthermore, LB(f) indicates a lower bound on the P where, LB(f)$\leq f(s)$, $\forall s \in S$. Clearly, it is desired to determine lower bounds which are as close as possible to the optimum solution.

As it is explained in remark 1, the objective function is a special case of the TWT. It is known that due to the characteristics of this objective function, lower bounds are much more difficult to derive than for the classical

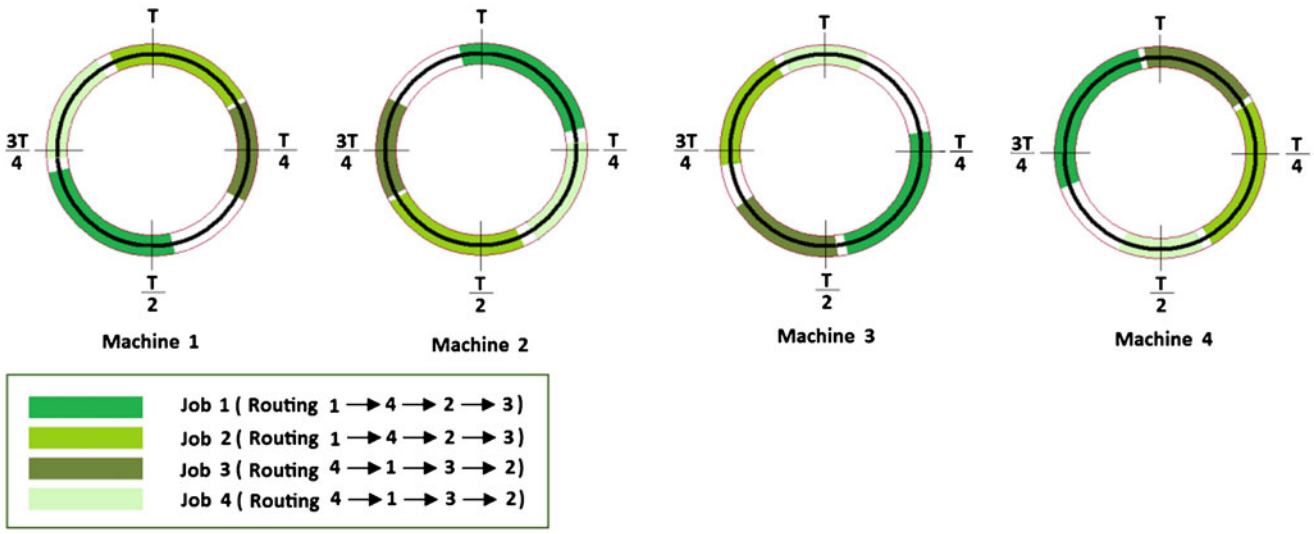

Fig. 11 Re-definition of the PJSSP

makespan [37]. Braune et al. [37] performed a computational study of lower bounding schemes for job shop scheduling problems with the TWT objective. Hoitomt et al. [38] applied the Lagrangian relaxation technique to job shop scheduling problems. Lancia et al. [39] introduced a time-indexed formulation for the general job shop problem with the column generation and provided an LB through LP relaxation.

In this paper, to further illustrate the effectiveness and performance of the algorithm, three procedures are studied to find efficient lower bounds for the PJSSP. At first it should be noted that the PJSSP can be redefined as the problem of ordering operations in the circles scaled from 1 to *T*, where each circle corresponds with a machine. Figure 11 shows a sample consisting of scheduling four jobs that are processed on four machines.

In the first proposed LB procedure, machines, circles in the new definition, are decomposed into a definite number of clusters (e.g., the problem shown in Fig. 11 can be decomposed into two clusters including {1, 4} and {2, 3}). In the next step, each cluster is solved under the condition of relaxing the release times. The summation of delays plus total processing times indicates the lower bound. It is worth noting that clusters should be defined by considering the routing of jobs (i.e., each cluster should contain the machines that provide a consecutive operations of jobs).

Table 6 Comparison results of the PSO-SA algorithm with SA, PSO, EM, EM-SA, and B&B algorithms

Problem characteristics			Algorithms						Lower bounds (%)			
Job	Machine	Period	SA	PSO	PSO-SA	EM	EM-SA	B&B	LB1	LB2 (1-M)	LB2 (2-M)	LB3
6	6	105	Inf.	Inf.	Inf.	Inf.	Inf.	Inf.	–	–	–	–
6	6	120	0.33	1.00	0.27	0.41	0.33	0.00	60	86	83	54
6	6	150	0.35	1.00	0.00	0.21	0.00	0.00	24	73	68	11
6	6	300	0.88	0.88	0.00	1.00	0.13	0.00	24	73	68	11
6	7	150	1.00	0.70	0.52	0.52	0.52	0.00	48	80	77	35
6	7	180	0.00	1.00	0.00	0.79	0.00	0.00	39	76	73	24
6	7	300	0.64	1.00	0.00	0.71	0.14	0.00	39	76	73	24
6	8	150	0.61	1.00	0.15	0.26	0.27	0.00	52	86	82	47
6	8	180	0.00	1.00	0.00	0.14	0.00	0.00	34	80	75	26
6	8	300	0.00	1.00	0.00	0.14	0.00	0.00	34	80	75	26
6	9	150	0.24	1.00	0.24	0.51	0.24	0.00	60	82	66	54
6	9	180	0.21	0.72	0.21	1.00	0.21	0.00	47	83	80	39
6	9	300	0.00	1.00	0.00	0.16	0.00	0.00	37	80	76	27
6	10	150	0.83	1.00	0.31	0.47	0.34	0.00	59	66	78	63
6	10	210	1.00	0.57	0.00	0.49	0.32	0.00	17	79	72	24
6	10	300	0.33	1.00	0.00	0.55	0.00	0.00	17	79	72	24
8	8	150	0.00	1.00	0.22	0.76	0.00	0.00	66	74	74	79
8	8	210	0.00	1.00	0.01	0.81	0.00	NA	35	67	66	60
8	8	300	0.24	1.00	0.25	0.86	0.24	0.00	35	67	66	60
8	10	150	0.42	1.00	0.35	0.77	0.42	0.00	70	100	80	81
8	10	210	0.32	1.00	0.00	0.91	0.32	NA	47	77	77	66
8	10	300	0.00	1.00	0.08	0.42	0.00	NA	30	70	70	56
10	10	210	0.39	1.00	0.00	0.62	0.19	NA	49	78	68	73
10	10	300	0.00	1.00	0.35	0.97	0.00	NA	26	65	63	62
12	12	210	0.23	0.30	0.00	1.00	0.23	NA	73	82	80	88
12	12	240	0.00	0.69	0.06	1.00	0.00	NA	54	74	73	80
12	16	240	0.08	1.00	0.00	0.71	0.08	NA	68	65	68	87
12	16	270	0.44	1.00	0.00	0.64	0.44	NA	61	67	61	83
14	20	270	0.16	0.92	0.00	1.00	0.16	NA	76	86	84	92
14	20	300	0.00	1.00	0.47	0.64	0.00	NA	70	75	73	90
Sum:			8.32	26.65	3.91	18.54	4.39	0	–	–	–	–
Average:			0.29	0.92	0.13	0.64	0.15	0	47	77	73	53

Inf. infeasible, *NA* not achievable

For example, in Fig. 11, defining clusters {1, 2} and {3, 4} are unacceptable.

The second proposed LB is based on the method introduced by Singer and Pinedo [40]. This technique is enforced by relaxing the capacity constraints of all machines except one. This remaining machine is then sequenced optimally by solving a derived single machine scheduling problem which is specified by defining a release date and a local due date for each operation on the respective machine.

The third proposed LB is based on the B&B algorithm. In this method, instead of selecting the most promising solution to branch or performing a deep exploration known as last-in first-out (LIFO) rule, first-in first-out (FIFO) rule is executed (i.e., no child nodes are surveyed unless all nodes that belong to the higher level are investigated). In this method, if an example with five machines and five jobs contains exactly two branches for each node, 2^{25} nodes should be investigated so that the found LB equals the optimum solution.

4.3 Experimental results

To illustrate the effectiveness and performance of the hybrid algorithm (i.e., PSO-SA) proposed in this paper, it is implemented in VB on a laptop with Pentium IV Core 2 Duo 2.53 GHz CPU. The outputs of the hybrid PSO-SA are compared with that achieved by the SA, PSO, and EM proposed by Jamili et al. [41] as well as the B&B algorithm proposed in section 4.1. Each instance can be characterized by a number of parameters, such as number of jobs, number of machines, and operation routings of jobs, the delivery time of jobs, processing times, and the period length. All the generated instances are based on the following assumptions:

- All jobs visit all machines only once, and the number of operations of each job equals the number of machines.
- The routings are randomly generated.
- Delivery times of jobs are considered zero.
- Processing times are all integer numbers between the interval [10, 20] generated at random.
- The objective is to minimize the completion time of all jobs.

The randomly generated instances are solved by all mentioned algorithms, and the related results are reported in terms of the RDI in Table 6. In this table, the last four columns present the gap among the lower bounds computed based on the LBs introduced in Section 4.2 and the best found solutions. The second LB is solved under the condition of reducing the problem to single and double machine problems, and the results are reported in two individual columns in Table 6. Figure 12 shows the means plot and Tukey intervals for the type of the algorithm. The outputs demonstrate that the hybrid PSO-SA can clearly result better solutions than using SA, PSO, and EM. The results show that there is no significant priority between the PSO-SA and EM-SA; however, since the EM-SA has the SA solution as one of the initial solution, one can find the proposed PSO-SA more effective than the EM-SA. Finally, Fig. 13 depicts the convergence rate of the PSO-SA algorithm improving the solutions in terms of the OFV.

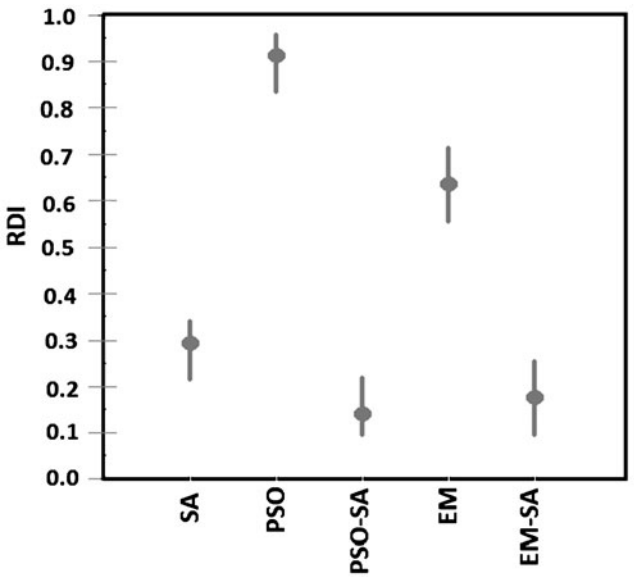

Fig. 12 Means plot and Tukey intervals (at 95% confidence level) for the type of the algorithm factor

5 Conclusion

In this paper, an effective hybrid PSO-SA based on particle swarm optimization (PSO) and simulated annealing (SA) algorithms has been proposed in order to solve the PJSSP.

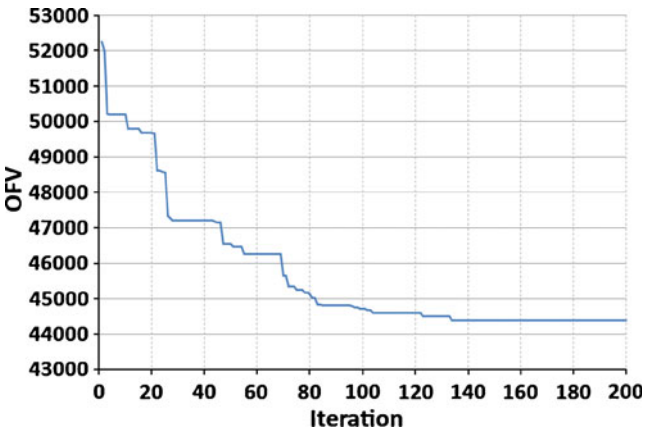

Fig. 13 The convergence rate

The performance of the proposed algorithm has been evaluated in comparison with the results obtained by the SA and PSO algorithms alone as well as the EM-SA algorithm. Moreover, a B&B algorithm was proposed to find optimum solutions of small instances. Beyond the LB proposition [40], two new added LBs are also proposed and investigated in this paper. Futhermore, the gap among the LBs and the best found solutions are reported.

The achieved results demonstrated the effectiveness of the proposed hybrid PSO-SA algorithm. Future research directions suggested are as follows: (1) solving the PJSSP under an uncertain environment, such as looking for an appropriate schedule where it is robust against some predefined interruptions, (2) utilizing other well-known metaheuristics to compare the results with the ones reported by the proposed PSO-SA algorithm, (3) considering other objective functions at different practical constraints, (4) extending the proposed algorithm to other similar scheduling problems, such as train scheduling, considered as an application of the JSSP, and (5) finally, investigating the application of other neighborhood methods.

Acknowledgment The authors would like to thank the anonymous referees for their constructive comments on the earlier version of this paper.

References

1. Guo ZX, Wong WK, Leung SYS, Fan JT, Chan SF (2008) Genetic optimization of order scheduling with multiple uncertainties. Expert Syst Appl 35:1788–1801
2. Guo ZX, Wong WK, Leung SYS, Fan JT, Chan SF (2008) A genetic-algorithm-based optimization model for scheduling flexible assembly lines. Int J Adv Manuf Technol 36:156–168
3. Gao J, He G, Wang Y (2009) A new parallel genetic algorithm for solving multiobjective scheduling problems subjected to special process constraint. Int J Adv Manuf Technol 43:151–160
4. Zhang C, Rao Y, Li P (2008) An effective hybrid genetic algorithm for the job shop scheduling problem. Int J Adv Manuf Technol 39:965–974
5. Naderi B, Khalili M, Tavakkoli-Moghaddam R (2009) A hybrid artificial immune algorithm for a realistic variant of job shops to minimize the total completion time. Comput Ind Eng 56(4):1494–1501
6. Sha DY, Lin HH (2009) A multi-objective PSO for job-shop scheduling problems. Expert Syst Appl. doi:10.1016/j.eswa.2009.06.041
7. Xia WJ, Wu ZM (2006) A hybrid particle swarm optimization approach for the job-shop scheduling problem. Int J Adv Manuf Technol 29:360–366
8. Mehdizadeh E, Sadi-Nezhad S, Tavakkoli-Moghaddam R (2008) Optimization of fuzzy clustering criteria by a hybrid PSO and fuzzy *c*-means clustering algorithm. Iran J Fuzzy Syst 5(3):1–14
9. Eswaramurthy VP, Tamilarasi A (2009) Hybridizing tabu search with ant colony optimization for solving job shop scheduling problems. Int J Adv Manuf Technol 40:1004–1015
10. Wang YM, Xiao NF, Yin HL, Hu EL, Zhao CG, Jiang YR (2008) A two-stage genetic algorithm for large size job shop scheduling problems. Int J Adv Manuf Technol 39:813–820
11. Pan CH, Huang HC (2009) A hybrid genetic algorithm for no-wait job shop scheduling problems. Expert Syst Appl 36:5800–5806
12. Roshanaei V, Khaleghei A, Balagh G, Esfahani MMS, Vahdani B (2009) A mixed-integer linear programming model along with an electromagnetism-like algorithm for scheduling job shop production system with sequence-dependent set-up times. Int J Adv Manuf Technol. doi:10.1007/s00170-009-2210-9
13. Garey MR, Johnson DS, Sethi R (1976) The complexity of flowshop and job-shop scheduling. Math Oper Res 1:117–129
14. Serafini P, Ukovich W (1989) A mathematical model for periodic scheduling problems. SIAM J Discrete Math 2(4):550–581
15. Brucker P, Kampmeyer T (2005) Tabu search algorithms for cyclic machine scheduling problems. J Sched 8:303–322
16. Brucker P, Kampmeyer T (2008) A general model for cyclic machine scheduling problems. Discrete Appl Math 156:2561–2572
17. Chrétienne P (1991) The basic cyclic scheduling problem with deadlines. Discrete Appl Math 30:109–123
18. Munier A (1996) The basic cyclic scheduling problem with linear precedence constraints. Discrete Appl Math 64:219–238
19. Kimbrel T, Sviridenko M (2008) High-multiplicity cyclic job shop scheduling. Oper Res Lett 36:574–578
20. Cavory G, Dupas R, Goncalves G (2005) A genetic approach to solving the problem of cyclic job shop scheduling with linear constraints. Eur J Oper Res 161:73–85
21. Song JS, Lee TE (1998) Petri net modeling and scheduling for cyclic job shops with blocking. Comput Ind Eng 34 (2):281–295
22. Tohme H, Nakamura M, Hachiman K, Onaga K (1999) Evolutionary Petri net approach to periodic job-shop scheduling. In: Proc IEEE Int Conf Syst Man Cybern (SMC'99) 12–15 October 1999, Tokyo, Japan, vol. 4:441–446
23. Song JS, Lee TE (1996) A tabu search procedure for periodic job shop scheduling. Comput Ind Eng 30(3):321–577
24. Lee TE, Posner ME (1997) Performance measures and schedules in periodic job shops. Oper Res 45(1):72–91
25. Liebchen C (2006) Periodic timetable optimization in public transport. PhD Thesis, Technische Universität Berlin
26. Kinder M (2008) Models for periodic timetabling. Master's thesis, Technische Universität Berlin
27. Liebchen C, Peeters L (2002) Some practical aspects of periodic timetabling. In: Cha-moni P, Leisten R, Martin A, Minnemann J, Stadtler H (eds) Operations research proceeding 2001. Springer, Berlin
28. Odijk MA (1994) Construction of periodic timetables. Part I: a cutting plane algorithm. Technical Report, DUT-TWI-94-61, Delft, The Netherlands
29. Nachtigall K (1996) Cutting planes for a polyhedron associated with a periodic network. DLR Interner Bericht, 112-96/17
30. Chang PC, Chen SH, Fan CY (2009) A hybrid electromagnetism-like algorithm for single machine scheduling problem. Expert Syst Appl 36:1259–1267
31. Naderi B, Tavakkoli-Moghaddam R, Khalili M (2009) Electromagnetism-like mechanism and simulated annealing algorithms for flowshop scheduling problems minimizing the total weighted tardiness and makespan. Knowl-Based Syst. doi:10.1016/j.knosys.2009.06.002
32. Tavakkoli-Moghaddam R, Khalili M, Naderi B (2009) A hybridization of simulated annealing and electromagnetic-like mechanism for job shop problems with machine availability and sequence-dependent setup times to minimize total weighted tardiness. Soft Comput 13:995–1006
33. Lin TL, Horng SJ, Kao TW, Chen YH, Run RS, Chen RJ, Lai JL, Kuo IH (2009) An efficient job-shop scheduling algorithm based

on particle swarm optimization. Expert Syst Appl. doi:10.1016/j.eswa.2009.08.015
34. Liu B, Wang L, Jin YH (2007) An effective hybrid particle swarm optimization for no-wait flow shop scheduling. Int J Adv Manuf Technol 31:1001–1011
35. Zhang G, Shao X, Li P, Gao L (2009) An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. Comput Ind Eng 56:1309–1318
36. Kennedy J, Eberhart RC (1995) Particle swarm optimization. In: Proc IEEE International Conference on Neural Networks, IEEE service Center, Piscataway, N.J., 4:1942–1948
37. Braune R, Zapfel G, Affenzeller M (2009) A computational study of lower bounding schemes for total weighted tardiness job shops. In: Proceedings of the 2nd International Symposium on Logistics and Industrial Informatics, LINDI 2009. pp 1–6
38. Hoitomt DJ, Luh PB, Pattipati KR (1993) A practical approach to job-shop scheduling problems. IEEE Trans Robot Automation 9 (1):1–13
39. Lancia G, Rinaldi F, Serafini P (2007) A compact optimization approach for job-shop problems. In: Baptiste P, Kendall G, Munier-Kordon A, Sourd F (eds) Proceedings of the 3rd Multidisciplinary International Conference on Scheduling: theory and applications (MISTA), pp 293–300
40. Singer M, Pinedo M (1998) A computational study of branch and bound techniques for minimizing the total weighted tardiness in job shops. IIE Trans 30(2):109–118
41. Jamili A, Shafia MA, Tavakkoli-Moghaddam R (2009) A hybridization of simulated annealing and electromagnetism-like mechanism for a periodic job shop scheduling problem. Working paper