Algorithm for Solving Parallel Machines Scheduling Problem to Minimize Earliness and Tardiness Costs

Pensiri Sompong*

Kasetsart University, Chalermphrakiat Sakon Nakhon Province Campus,
Sakon Nakhon, Thailand

Received: 24 December 2019, Revised: 6 February 2020, Accepted: 14 February 2020

Abstract

Algorithm for parallel machines scheduling problem to minimize the earliness and tardiness costs is proposed in this study. The problem is associated with the assignment of jobs to machines and determination of staring time for each job in a given sequence. Population-based incremental learning (PBIL) algorithm is used to allocate the jobs to machines. The optimal timing algorithm based on the minimum block cost function calculation is then employed to decide the starting time of jobs on each machine. To illustrate the performance of proposed algorithm, numerical examples generated randomly are tested. The numerical results obtained from PBIL combined with optimal timing algorithm called PBILOTA are compared to EDDPM (Earliest Due Date for Parallel Machines) to indicate the decrease in penalty cost. From the experimental results, it is shown that PBILOTA is an efficient algorithm for solving parallel machines scheduling problem with earliness-tardiness costs minimization.

Keywords: population-based incremental learning algorithm, scheduling, parallel machines, earliness, tardiness
DOI 10.14456/cast.2020.8

1. Introduction

Study related to minimization of earliness and tardiness costs plays an important role in production system due to the Just-in-Time (JIT) production philosophy. The problem associated with the objective to minimize earliness and tardiness penalties has been focused on algorithm design making the jobs finished exactly on their due dates or as close as possible. Lee and Choi [1] considered a job scheduling problem for a single machine to minimize early-tardy penalty costs. An optimal timing algorithm was used to decide the optimal starting time of each job in a given sequence generated by genetic algorithm. The optimal starting time was obtained by shifting a block to a point giving the minimum block cost function. Bauman and Józefowska [2] proposed an algorithm for solving a single machine scheduling problem with linear earliness and tardiness costs.

*Corresponding author: Tel.: +66 42 72 5033 Fax: +66 42 72 5034
E-mail: pensiri.so@ku.th

The objective was to find a vector of job completion time such that the total cost is minimum. Kedad-Sidhoum and Sourd [3] proposed fast neighborhood search based on a block representation of schedule. To get a larger neighborhood search, random swap and earliness-tardiness perturbation were performed. Feasible solutions were considered as a vector of the completion times of all jobs. Kianfar and Moslehi [4] developed a branch and bound algorithm for solving a single machine scheduling problem to minimize the weighted quadratic earliness and tardiness penalties such that no machine idle time was allowed. An arc-time-indexed formulation and a branch and bound algorithm were presented by Keshavarz *et al*. [5] to investigate a single machine sequence dependent group scheduling problem combined with earliness and tardiness considerations. The single machine scheduling problem with distinct time windows and sequence dependent setup time was addressed by Rosa *et al*. [6]. The problem involved the determination of job sequence and starting time of each job in the sequence. Implicit enumeration and general variable neighborhood search algorithms were proposed to determine the job sequence and idle time insertion algorithm was then used to determine the starting time for each job. In addition, parallel machines system has been considered. Kayvanfar *et al*. [7] studied scheduling problem on unrelated parallel machines with the objective to minimize earliness-tardiness costs and makespan simultaneously. Therefore, no inserted idle time was allowed. ISETP was used to assign the jobs on parallel machines and PNBC-NBE heuristic was then applied to acquire the optimal set of jobs compression and expansion processing time in a given sequence. Scheduling problem on unrelated parallel machines with sequence dependent setup time was studied by Zeidi and Mohammad Hosseini [8]. An integrated meta-heuristic algorithm consisting of genetic algorithm and simulated annealing method was proposed to solve the problem in which simulated annealing method was used as a local search to improve the quality of solutions. Alvarez-Valdes *et al*. [9] proposed hybrid heuristic algorithm combining priority rules for assigning jobs to machine and local search for solving the one-machine subproblem. Path relinking and scatter search were also applied to obtain high quality of solutions. Hung *et al.* [10] addressed the scheduling jobs with time windows on unrelated parallel machines with sequence dependent setup time. Machine and job dependent processing times were also considered. Three solution methods based on mixed integer programming (MIP) called HCMIP, ETMIP and HIMIP were proposed for solving the problem. Wu *et al.* [11] investigated unrelated parallel machine scheduling with consideration of job rejection and earliness-tardiness penalties. Hybrid algorithm combining genetic algorithm and tabu search were presented to solve the problem.

In the literature, the exact method and heuristic algorithm were proposed for solving scheduling problem. This study is associated with the job assignment to parallel machines and determination of starting time for each job in a given sequence. Population-based incremental learning (PBIL) algorithm explored by Baluja [12] is used to create feasible solution indicating a sequence of jobs on each machine. PBIL is an evaluation algorithm combining the mechanism of genetic algorithm and competitive learning. Probability vector is applied to define a population representing solution of problem. To obtain high quality of solution, high values of probability in probability vector are updated based on the best solution at each iteration. Once a job sequence is given, optimal timing algorithm is applied to decide optimal starting time for each job in the sequence as one-machine subproblem.

The remainder of this paper is organized as follows. Section 2 describes the problem description and some notations used throughout in this paper. Three algorithms for assigning and sequencing jobs to parallel machines, dispatching rule based on earliest due date, optimal timing and population-based incremental learning algorithms are also explained in this section. Section 3 presents the experiment results obtained from the proposed algorithm. Finally, the conclusions are provided in section 4.

2. Materials and Methods

2.1 Problem description

Given n jobs and m parallel machines, the job i for $i=1,2,...,n$ can be processed on any machines k for $k=1,2,...,m$ without interruption. A machine can perform only one job at a time and the processing time of job *i*, denoted by p_i, is identical for all machines. Let d_i and c_i be due date and completion time of job *i*, respectively. Job *i* is said to be early if $c_i < d_i$ and the earliness cost is $\alpha_i(d_i - c_i)$ where α_i is earliness weight while it is said to be tardy if $c_i > d_i$ and the tardiness cost is $\beta_i(c_i - d_i)$ where β_i is tardiness weight. Job *i* is on time if $c_i = d_i$. All jobs are available to process at time zero and can be started immediately after the predecessor job is completed. The inserted idle time is allowed to determine the starting time of each job in which the completion time is close to its due date. The cost of job *i* processed completely before or after its due date is represented by equation (1).

$$f(c_i) = \alpha_i E_i + \beta_i T_i \tag{1}$$

Where $E_i = \max\{0, d_i - c_i\}$ and $T_i = \max\{0, c_i - d_i\}$. The objective of the study is to determine the job schedule that minimize the total cost calculated by the following equation.

$$\text{Total cost} = \sum_{i=1}^{n} f(c_i) \tag{2}$$

2.2 Dispatching rule

Definition 2.1 A set of jobs $\{J_1, J_2, ..., J_n\}$ is in the order of earliest due date (EDD) if the jobs are sequenced according to the non-decreasing due date, i.e., $d_{J_1} \le d_{J_2} \le ... \le d_{J_n}$.

In order to obtain the job sequence on each machine, the jobs in EDD order are assigned to the machines depending on total completion time. The steps for the job assignment called Earliest Due Date for Parallel Machines (EDDPM) are as follows.

Algorithm 1: EDDPM

Step 1: Let $S = \{J_1, J_2, ..., J_n\}$ be a set of jobs sequenced by EDD rule and $S' = \varnothing$ be a set of assigned jobs.

Step 2: Let $TC_k = 0$ be the total completion time on machine k, $k = 1, 2, ..., m$. Assign the first job J_1 to the first machine and update $TC_1 = p_{J_1}$. Put J_1 in S to S', $S' = S' \cup \{J_1\}$ and $S = S - \{J_1\}$

Step 3: Assign the next job J_i in S to all machines and calculate temporary $TC_k^{'} = TC_k + p_{J_i}$. Job J_i is processed on a machine giving minimum $TC_k^{'}$, in the case that $TC_k^{'} = TC_p^{'}$, where $k \ne p$, a machine with minimum index is chosen. Update $TC_k = TC_k^{'}$, $S' = S' \cup \{J_i\}$ and $S = S - \{J_i\}$.

Step 4: Continue step 3 until $S' = \{J_1, J_2, ..., J_n\}$ and $S = \varnothing$.

To present the job sequence and penalty cost resulting from applying EDDPM, the input data for 10 jobs given in Table 1 are used for the calculation and the results are shown in example 2.1.

Table 1. Input data for 10 jobs

Job	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
p_i (day)	6	5	1	4	3	10	3	3	4	4
d_i (day)	11	10	12	15	15	8	13	14	14	10
α_i ($/day)	1.9	1.8	1.8	1.1	2.3	1.4	1.9	1.2	1.8	0.9
β_i ($/day)	3.8	3.7	3.4	1.4	2.7	1	0.9	0.8	1.8	3.3

Example 2.1 Consider the input data given in Table 1. A set of job ordered by EDD rule is $S=\{6,2,10,1,3,7,8,9,4,5\}$. By applying EDDPM, job sequence for each machine is shown in Figure 1.

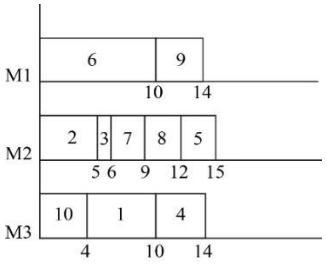


Figure 1. Job sequences for each machine resulting from applying EDDPM algorithm

It can be seen from Figure 1 that job 6 is completed after the due date for two days, i.e. $E_6=\max\{0,d_6-c_6\}=0$ and $T_6=\max\{0,c_6-d_6\}=2$ while job 9 is completed on its due date, i.e. $E_9=\max\{0,d_9-c_9\}=0$ and $T_9=\max\{0,c_9-d_9\}=0$. The costs of job 6 and job 9 calculated by equation (1) are as follows:

$$f(c_6)=(1.4)(0)+(1)(2)=2$$
$$f(c_9)=(1.8)(0)+(1.8)(0)=0$$

Therefore, the penalty cost of machine 1 calculated by equation (2) is \$2. For machine 2, jobs 2, 3, 7 and 8 are finished before the due date for 5, 6, 4 and 2 days, respectively, and job 5 is finished on its due date. Then,

$$E_2=\max\{0,d_2-c_2\}=5,\ T_2=\max\{0,c_2-d_2\}=0,$$
$$E_3=\max\{0,d_3-c_3\}=6,\ T_3=\max\{0,c_3-d_3\}=0,$$
$$E_7=\max\{0,d_7-c_7\}=4,\ T_7=\max\{0,c_7-d_7\}=0,$$
$$E_8=\max\{0,d_8-c_8\}=2,\ T_8=\max\{0,c_8-d_8\}=0,$$
$$E_5=\max\{0,d_5-c_5\}=0,\ T_5=\max\{0,c_5-d_5\}=0,$$

and the costs for all jobs are as follows:

$$f(c_2) = (1.8)(5) + (3.7)(0) = 9$$
$$f(c_3) = (1.8)(6) + (3.4)(0) = 10.8$$
$$f(c_7) = (1.9)(4) + (0.9)(0) = 7.6$$
$$f(c_8) = (1.2)(2) + (0.8)(0) = 2.4$$
$$f(c_5) = (2.3)(0) + (2.7)(0) = 0$$

Thus, the penalty cost of machine 2 is \$29.8. Similarly, the penalty cost of machine 3 is \$8.4 and the total cost for all machines is \$40.2.

2.3 Optimal timing algorithm

Optimal timing algorithm is an algorithm used to decide the optimal starting time of each job. Lee and Choi [1] proposed an optimal timing algorithm to determine the starting time of the jobs in a given sequence without total cost evaluation. The minimum block cost function is calculated to determine the extreme point which decides the starting time of the first job in the block. The extreme point is a point where the slope begins to be greater than or equal to zero. Due to the earliness and tardiness penalties, idle time can be inserted between blocks to shift the entire block toward the minimum point. In order to apply the optimal algorithm proposed by Lee and Choi [1] to this study, the steps of the algorithm can be concluded as follows.

Algorithm 2: Optimal timing algorithm (OTA)

Step 1: Let $S = \{J_1, J_2, ..., J_n\}$ be a set of jobs sequenced by EDD rule and $S' = \varnothing$ be a set of assigned jobs.

Step 2: Put the first job J_1 in block B_1, $s_{J_1} = \max(0, d_{J_1} - p_{J_1})$ and $c_{J_1} = \max(d_{J_1}, p_{J_1})$, where s_{J_1} and c_{J_1} are starting and completion times of job J_1, respectively. $S = S - \{J_1\}$ and $S' = \{J_1\}$.

Step 3: For any block B_t and job J_i, $i = 2, ..., n$, consider the following three cases.

(1) If $c_{J_{i-1}} + p_{J_i} < d_{J_i}$, then $s_{J_i} = d_{J_i} - p_{J_i}$, $c_{J_i} = d_{J_i}$, $t = t + 1$, $B_t = \{J_i\}$, $S = S - \{J_i\}$ and $S' = S' \cup \{J_i\}$.

(2) If $c_{J_{i-1}} + p_{J_i} = d_{J_i}$, then $s_{J_i} = c_{J_{i-1}}, c_{J_i} = d_{J_i}$, $B_t = B_t \cup \{J_i\}$, $S = S - \{J_i\}$ and $S' = S' \cup \{J_i\}$.

(3) If $c_{J_{i-1}} + p_{J_i} > d_{J_i}$, then $s_{J_i} = c_{J_{i-1}}$, $c_{J_i} = s_{J_i} + p_{J_i}$ and $B_t = B_t \cup \{J_i\}$, $S = S - \{J_i\}$ and $S' = S' \cup \{J_i\}$.

Step 4: Determine the minimum block cost function by comparing the slopes of block which can be obtained from the earliness and tardiness weights and shift the entire block B_t toward the minimum point until one of the following cases occurs.

(1) s_{J_1} in block B_1 is zero.

(2) The minimum point is reached.

(3) s_{J_i} in block B_t equals to $c_{J_{i-1}}$ in block B_{t-1}. In this case, blocks B_t and B_{t-1} are concatenated and the new minimum block cost function has to be calculated.

Step 5: Continue steps 3 and 4 until $S' = \{J_1, J_2, ..., J_n\}$ and $S = \varnothing$.

Example 2.2 To determine the optimal starting time of the jobs on each machine as shown in Figure 1 with the given input data in Table 1, algorithm 2 is utilized. In this example, only block cost function of machine 2 is calculated.

Job 2:

Step 1: Let $S = \{2,3,7,8,5\}$ be the job sequence of machine 2 and $S' = \varnothing$.

Step 2: Set $B_1 = \{2\}$, $s_2 = 5$, $c_2 = 10$, $S = \{3,7,8,5\}$ and $S' = \{2\}$. In this case, job 2 has no penalty.

Job 3:

Step 3: Since $c_2 + p_3 = 11 < d_3$, $s_3 = 11$, $c_3 = 12$, $B_2 = \{3\}$, $S = \{7,8,5\}$ and $S' = \{2,3\}$. Job 3 has no penalty.

Job 7:

Step 3: Since $c_3 + p_7 = 15 > d_7$, $s_7 = 12$, $c_7 = 15$, $B_2 = \{3,7\}$, $S = \{8,5\}$ and $S' = \{2,3,7\}$.

Step 4: Compare the slope of block B_2, $-\alpha_3 - \alpha_7 = -3.7$, $-\alpha_3 + \beta_7 = -0.9$ and $\beta_3 + \beta_7 = 4.3$. Thus, job 3 is still completed on its due date and block B_2 is not shifted.

Job 8:

Step 3: Since $c_7 + p_8 = 18 > d_8$, $B_2 = \{3,7,8\}$, $s_8 = 15$, $c_8 = 18$, $S = \{5\}$ and $S' = \{2,3,7,8\}$.

Step 4: Compare the slope of block B_2 as follows.

$$-\alpha_3 - \alpha_7 - \alpha_8 = -4.9$$
$$-\alpha_3 - \alpha_7 + \beta_8 = -2.9$$
$$-\alpha_3 + \beta_7 + \beta_8 = -0.1$$
$$\beta_3 + \beta_7 + \beta_8 = 5.1$$

Job 3 is finished on its due date and block B_2 is placed at the present position.

Job 5:

Step 3 : Since $c_8 + p_5 = 21 > d_5$, $B_2 = \{3,7,8,5\}$, $s_5 = 18$, $c_8 = 21$, $S = \varnothing$ and $S' = \{2,3,7,8,5\}$.

Step 4: By comparing the slope of block B_2, job 7 is finished on its due date and it has to be shifted for 2 time units left. Because $c_7 - d_7 > s_3 - c_2$, block B_2 can be shifted only one time unit. Now, block B_1 and B_2 are concatenated such that $s_3 = 10$, $c_3 = 11$, $s_7 = 11$, $c_7 = 14$, $s_8 = 14$, $c_8 = 17$, $s_5 = 17$, $c_5 = 20$ and $B_1 = \{2,3,7,8,5\}$. Due to the concatenation of block, the new minimum block cost function of B_1 is calculated and it is found that job 7 is finished on its due date. Thus, block B_1 is shifted for one time unit left. Now, $S = \varnothing$ and $S' = \{2,3,7,8,5\}$, therefore, the algorithm is terminated.

The optimal starting time of jobs on each machine is shown in Figure 2. The earliness and tardiness costs of machine 2 is reduced to \$17.8 and the total cost for all machines is \$24.3.

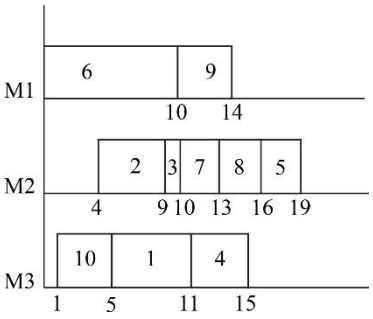


Figure 2. Optimal starting time for each job after applying algorithm 2

2.4 Population-based incremental learning algorithm

Population-based incremental learning (PBIL) algorithm is an evaluation algorithm using a probability vector to describe the population representing the solution of the problem. Each position of solution is represented by either 0 or 1 which can be obtained by the probability vector. Suppose that probability generating 1 for *k*-th position is 0.6, the probability generating 0 for *k*-th position is 0.4 resulting from subtracting 0.6 from 1. The following definition describes the job assignment to the machines used in this study.

Definition 2.2 Let $A=[a_{ki}]$ be an $m\times n$ matrix such that $a_{ki}\in\{0,1\}$. A is satisfied by the properties that $\sum_{k=1}^{m} a_{ki}=1$ and $\sum_{k=1}^{m}\sum_{i=1}^{n} a_{ki}=n$. If $a_{ki}=1$, then machine k operates job i and A is called population.

Example 2.3 Given population matrix A as follow.

$$A=\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Population matrix A is 3×10 matrix and it indicates that there are 3 machines and 10 jobs such that machine 1 operates jobs 6 and 9, machine 2 operates jobs 2, 3, 5, 7 and 8 and the last machine operates the remaining jobs satisfying job assignment as shown in Figure 1. It can be seen that population matrix A only specifies the jobs which will be processed by a machine. Thus, EDD rule is used to determine the sequence of jobs.

The procedure of PBIL combined with optimal timing algorithm called PBILOTA applied to scheduling problem to minimize earliness and tardiness costs is as follows.

Algorithm 3: PBILOTA

Step 1: Let $1\times n$ matrix P be the initial probability vector.
Step 2: Create a set of feasible solution called population according to probability vector P representing the job assignment on machines.
Step 3: Sort the jobs by EDD rule to determine the job sequence on each machine. The optimal timing algorithm is applied to find the starting time of jobs in each block and total cost is then

calculated for all solutions in step 2 based on eq. (2). The best solution is a population giving the minimum total cost.

Step 4: Update probability vector P by using the following equation.

$$prob_i^{(t)} = prob_i^{(t-1)}\left(1-LR\right)+Best_i^{(t-1)}\left(LR\right) \tag{3}$$

For $i=1,2,...,n$ and $Best_i^{(t-1)}$ is bit integer (0 or 1) of the best solution at iteration $t-1$ and LR is learning rate.

Step 5: Continue step 2 to step 4 until stopping criterion is satisfied. The maximum number of iteration, 500 iterations, is set to be stopping criteria. When the algorithm is terminated, probability values in probability vector are approached to either 0 or 1.

Example 2.4 In order to reduce earliness and tardiness costs by using PBILOTA, the initial probability vector is set based on the solution obtained from EDDPM combined with optimal timing algorithm in example 2.2.

After applying PBILOTA with the input data given in table 1 to determine optimal starting time for each job, the total penalty cost is reduced from \$24.3 to \$12.3 and job schedule for all machines is shown in Figure 3.

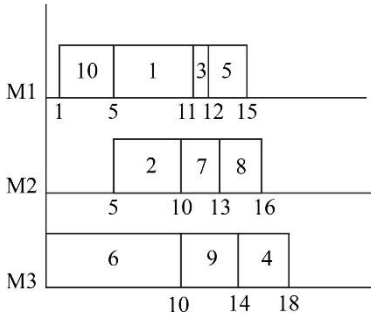


Figure 3. Job schedule obtained from applying PBILOTA

3. Results and Discussion

3.1 Validation

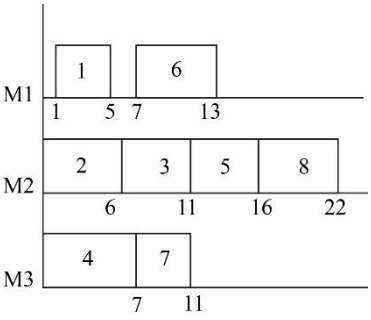
To validate and indicate that the proposed PBILOTA can reduce total cost, the job information presented in Kayvanfar *et al.* [7] is employed to test the algorithms. The information is shown in Table 2 consisting of processing time (day), due date (day), earliness and tardiness weights (\$/day) for each job. This information is used to illustrate ISETP performance for 8 job with 3 machines proposed by Kayvanfar *et al.* [7]. The results obtained from using ISETP, EDDPM and PBILOTA with the given information are shown in Figure 4.

Figure 4 shows the jobs schedule and total cost obtained from applying ISETP, EDDPM and PBILOTA. ISETP is a procedure proposed by Kayvanfar *et al.* [7] for assigning the jobs on parallel machines to minimize earliness-tardiness penalties and makespan simultaneously, thus no inserted idle time is allowed. To satisfy the objective of this study, optimal timing algorithm is utilized to find the starting time of each job to reduce the cost for comparison. As seen in Figure 4(a), the cost of ISETP after applying optimal timing algorithm is \$3.5. The solution of EDDPM shown in Figure 4(b) is set to be one of the populations in PBILOTA such that the cost is set to be

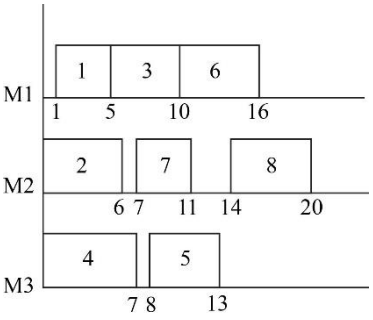
upper bound. Although the cost of EDDPM is more than the cost of ISETP, it is decreased from \$5 to \$2.5 after PBILOTA is performed as shown in Figure 4(c).

Table 2. Job information presented in Kayvanfar *et al.* [7].

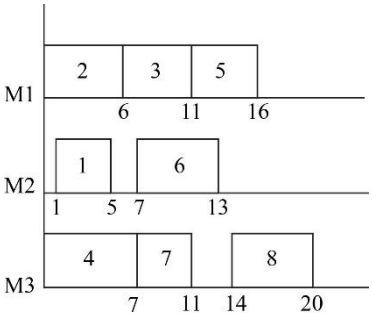
Job	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
p_i	4	6	5	7	5	6	4	6
d_i	5	5	11	6	13	13	11	20
α_i	0.5	1	1	1.25	1.5	1	1.5	0.5
β_i	0.5	0.5	1.25	0.5	0.5	1	3	0.5



(a) ISEPT: cost \$3.5



(b) EDDPM: cost \$5



(c) PBILOTA: cost \$2.5

Figure 4. The results obtained from (a) ISETP, (b) EDDPM and (c) PBILOTA with job information in Table 2

3.2 Parameter Setting

To illustrate the performance of the proposed method, numerical examples are generated. In order to generate the parameters used for the random problems, the procedures found in the literatures are applied. The proposed algorithm is tested on the problem instances consisting of 20, 40, 60, 80 and 100 jobs. Due to parallel machines, the number of machines is related to the number of jobs

such that $m = \lfloor n/\gamma + 0.5 \rfloor$, where $\gamma = 4, 5, 6$ and $\lfloor x \rfloor$ is the greatest integer less than x [10]. Processing time of job i is calculated based on the product of based processing time and machine adjusting factor, $b_i \times \delta_i$ [13]. In this study, b_i and δ_i are randomly generated from the uniform distribution with the range of [1,7] and [0.5,1.5], respectively. An integer due date is generated from the uniform distribution $\lfloor (1-T-R/2)P, (1-T+R/2)P \rfloor$ [4] for $P = \left(\sum_{i=1}^{n} p_i\right) \Big/ m$. Tardiness factor T is a parameter indicating the opportunity that a job is tardy and parameter R is due date range. To avoid crash due date in this study, normal due date can be randomly generated by setting $T = 0.2$ and $R = 0.5$. The earliness and tardiness weights are randomly generated from the range of [0.5,2.5] and [0.5,4.5], respectively [7]. For PBILOTA, population size is set to 100 and the initial probability vector is set based on the solution obtained from EDDPM. The learning rate represents the speed of convergence. As the learning rate is increased, the speed of convergence is increased while the search portion is decreased. To increase the chance for finding feasible solution, $LR = 0.05$ is used in this study and the maximum number of iterations, 500 iterations, is set to be the stopping criteria.

3.3 Results

By the parameter setting addressed in previous section, 5 random problems are generated for each level of γ and 5 run times of PBILOTA are performed for each problem. Thus, total of 375 problem combinations are tested. Since no exact numerical results are available, the results obtained from PBILOTA are compared to EDDPM combined with optimal timing algorithm to evaluate the performance.

Table 3 shows the average relative percentage deviation resulting from the comparison of PBILOTA and EDDPM algorithms mentioned in section 2. The first column shows the number of jobs with up to 100 jobs. The number of jobs divided by the different three levels of γ in the second column results in the different number of machines shown in the third column. To indicate that the application of PBILOTA can minimize the earliness and tardiness penalties, the relative percentage deviation (RPD) for all instances is calculated as follow:

$$RPD = \frac{sol_{EDDPM} - sol_{PBILOTA}}{sol_{EDDPM}} \times 100 \qquad (4)$$

where sol_{EDDPM} and $sol_{PBILOTA}$ are the solutions obtained from EDDPM and PBILOTA algorithms, respectively. Because five problems are randomly generated for each level of γ and PBILOTA is applied for five run times in each problem, the average RPD for each level is computed as shown in the last column of Table 3. The average RPD indicates the decrease in total cost compared to EDDPM. For 20 jobs with $\gamma = 4$ and $\gamma = 5$, total cost is reduced more than the other cases because the jobs can normally be allocated to machine resulting in the increase in the chance for shifting the job near to the due date. When the number of machines is decreased by the increase of γ, the average RPD is decreased because the different cost between EDDPM and PBILOTA is lower indicating that EDDPM creates good initial population. As seen from Table 3, the average RPD trends to be decreased except for the case of 60 jobs. In the case that the number of machines is decreased and the number of jobs is increased simultaneously, the number of jobs per machine is increased causing extended block size of job. Therefore, jobs at the beginning and at the end parts of block are completed far from their due dates. However, the better solution can be obtained

by PBILOTA based on the improvement of probability values in probability vector. Overall, PBILOTA can reduce the cost for all instances.

Figure 5 shows the convergence of proposed PBILOTA method for some problems of 20 to 100 jobs with different γ. It can be seen that the solutions are converged to a single point that the probability values in probability vector are approached to either 0 or 1.

Table 3. Average RPD obtained from the comparison of EDDPM and PBIL algorithms.

Jobs	γ	Machines	Average RPD (%)
$n = 20$	4	$m = 5$	42.23
	5	$m = 4$	44.34
	6	$m = 3$	24.73
$n = 40$	4	$m = 10$	29.29
	5	$m = 8$	39.07
	6	$m = 7$	31.81
$n = 60$	4	$m = 15$	27.08
	5	$m = 12$	31.78
	6	$m = 10$	32.11
$n = 80$	4	$m = 20$	33.28
	5	$m = 16$	31.73
	6	$m = 13$	24.90
$n = 100$	4	$m = 25$	28.29
	5	$m = 20$	24.94
	6	$m = 17$	23.85

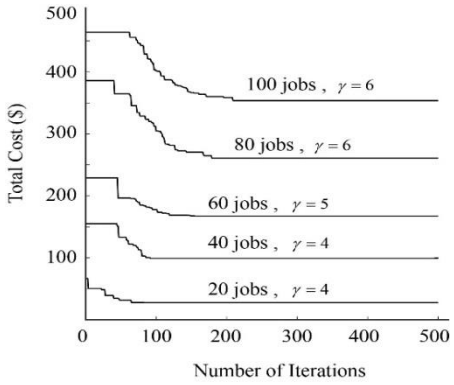


Figure 5. Convergence of solutions obtained from PBILOTA

4. Conclusions

Parallel machine scheduling problem to minimize earliness and tardiness cost is studied. Population-based incremental learning (PBIL) algorithm combined with optimal timing algorithm called PBILOTA is proposed for solving this problem. Computational results demonstrate that the better solution for each iteration can be obtained by the improvement of probability values. In the case that the number of machines is decreased and the number of jobs is increased simultaneously, PBILOTA is still able to find higher quality of solution. The results also show that EDDPM

creates good initial solution resulting in faster improvement of PBILOTA solution. It can be concluded that PBILOTA is an efficient algorithm and suitable for solving parallel machines scheduling problem with earliness and tardiness cost minimization. Developing other dispatching rules to create initial population or integrating heuristic algorithms in the proposed method may be performed for the extension of the study to improve the solution quality. Furthermore, additional conditions such as job sequence dependent setup time and machine dependent processing time are the options for further research.

References

[1] Lee, C.Y. and Choi, J.Y., 1995. A genetic algorithm for job sequencing problems with distinct due dates and general early-tardy penalty weights. *Computers & Operations Research*, 22(8), 857-869.

[2] Bauman, J. and Józefowska, J., 2006. Minimizing the earliness-tardiness costs on a single machine. *Computers & Operations Research*, 33(11), 3219-3230.

[3] Kedad-Sidhoum, S. and Sourd, F., 2010. Fast neighborhood search for the single machine earliness-tardiness scheduling problem. *Computers & Operations Research*, 37(8), 1464-1471.

[4] Kianfar, K. and Moslehi, G., 2012. A branch-and-bound algorithm for single machine scheduling with quadratic earliness and tardiness penalties. *Computers & Operations Research*, 39(12), 2978-2990.

[5] Keshavarz, T., Savelsbergh, M. and Salmasi, N., 2015. A branch-and-bound algorithm for the single machine sequence-dependent group scheduling problem with earliness and tardiness penalties. *Applied Mathematical Modelling*, 39(20), 6410-6424.

[6] Rosa, B.F., Souza, M.J.F., de Souza, S.R., de França Filho, M.F., Ales, Z. and Michelon, P., 2017. Algorithms for job scheduling problems with distinct time windows and general earliness/tardiness penalties. *Computers & Operations Research*, 88, 203-215.

[7] Kayvanfar, V., Komaki, GH.M., Aalaei, A. and Zandieh, M., 2014. Minimizing total tardiness and earliness on unrelated parallel machines with controllable processing time. *Computers & Operations Research*, 41, 31-43.

[8] Zeidi, J.R. and Mohammad Hosseini, S., 2015. Scheduling unrelated parallel machines with sequence-dependent setup times. *International Journal of Advanced Manufacturing Technology*, 81(9-12), 1487-1496.

[9] Alvarez-Valdes, R., Tamarit, J.M. and Villa, F., 2015. Minimizing weighted earliness-tardiness on parallel machines using hybrid metaheuristic. *Computers & Operations Research*, 54, 1-11.

[10] Hung, Y.-F., Bao, J.-S. and Chen, Y.-E., 2017. Minimizing earliness and tardiness costs in scheduling jobs with time windows. *Computers & Industrial Engineering*, 113, 871-890.

[11] Wu, R., Guo, S. and Li, X., 2017, Unrelated parallel machine scheduling with job rejection and earliness-tardiness penalties. *Proceedings of the 36th Chinese Control Conference*, Dalian, China, July 26-28, 2017, 2846-2851.

[12] Baluja, S., 1994. Population-based Incremental Learning: A Method forIintegrating Genetic Search Based Function Optimization and Competitive Learning. [online] Available at: https://www.ri.cmu.edu/pub_files/pub1/baluja_shumeet_1994_2/baluja_shumeet_1994_2.pdf

[13] Joo, C.M. and Kim, B.S., 2015. Hybrid genetic algorithms with dispatching rules for unrelated parallel machines scheduling with setup time and production availability. *Computers & Industrial Engineering*, 85, 102-109.