An Effective Estimation of Distribution Algorithm for Multi-track Train Scheduling Problem

Shengyao Wang and Ling Wang

Tsinghua National Laboratory for Information Science and Technology (TNList),
Department of Automation, Tsinghua University, Beijing, 100084, P.R. China
wangshengyao@tsinghua.org.cn, wangling@mail.tsinghua.edu.cn

Abstract. In this paper, an effective estimation of distribution algorithm (EDA) is presented for solving the multi-track train scheduling problem (MTTSP). The individual of the EDA is represented as the permutation of train priority. With a proper track assignment rule, the individual is decoded into feasible schedule. In addition, the EDA builds a probability model for describing the distribution of the solution space. In every generation, it samples the promising region for generating new individuals and updates the probability model with the superior population. Moreover, the influence of parameter setting is investigated based on design-of-experiment method and a set of suitable parameter values is suggested. Simulation results based on some instances and comparisons with the existing algorithm demonstrate the effectiveness and efficiency of the EDA.

Keywords: train scheduling, assignment rule, estimation of distribution algorithm, probability model.

1 Introduction

Due to the increasing traffic of the railway, the scheduling for trains becomes more and more important to the transportation efficiency. As a complicated combinational optimization problem, the train scheduling problem is difficult to solve [1]. Ghoseiri et al. [2] developed a multi-objective optimization model for the passenger train-scheduling problem on a railroad network including single and multiple tracks as well as multiple platforms with different train capacities. Dorfman and Medanic [3] developed a local feedback-based travel advance strategy based on a discrete-event model of the traffic on the railway network. Zhou and Zhong [4] developed a branch-and-bound (B&B) algorithm with effective dominance rules and a beam search algorithm with utility evaluation rules for multi-objective planning applications of a double-track train scheduling problem. D'ariano et al. [5] developed a B&B algorithm by using the estimation of time separation among trains and modeling the scheduling problem with an alternative graph formulation.

Besides, the train scheduling problems have some similar features compared with the shop scheduling problems [6-9]. Burdett and Kozan [6] addressed the representation and construction of accurate train schedules by a solution algorithm for the hybrid

D.-S. Huang et al. (Eds.): ICIC 2014, LNAI 8589, pp. 697–708, 2014.

job shop problems with capacitated buffers. Liu and Kozan [7] modeled the train scheduling problem as a blocking parallel-machine job-shop scheduling problem and proposed a constructive heuristic algorithm to obtain the feasible schedule. Recently, Zhang and Chen [8-9] modeled the problem as the blocking hybrid flow shop scheduling problem and proposed a hybrid particle swarm optimization (HPSO) algorithm for solving it.

As a relatively new population-based evolutionary algorithm, the estimation of distribution algorithm (EDA) has gained increasing study during recent years [10]. Moreover, the EDA has already been successfully applied to solve a variety of engineering optimization problems, especially the scheduling problems [11-15]. However, to the best of our knowledge, there is no research work about the EDA for solving the train scheduling problem. Inspired by the successful applications of the EDA, this paper attempts to present an effective EDA for solving the multi-track train scheduling problem (MTTSP). To be specific, the individual of the population is represented as the permutation of the train priority and is decoded into feasible schedule by a track assignment rule. A probability model is built for describing the distribution of the solution space. It samples the promising region for generating new individuals and updates the probability model with the superior population in every generation. The influence of parameter setting is investigated based on design-of-experiment method and a set of suitable parameter values is suggested consequently. Simulation results based on some instances and comparisons with the existing algorithms demonstrate the effectiveness and efficiency of the EDA.

The remainder of the paper is organized as follows. The MTTSP is described in Section 2. In Section 3 the basic EDA is introduced, and the EDA for solving the MTTSP is presented in Section 4. Simulation results and comparisons are provided in Section 5. Finally, the paper ends with some conclusions and future work in Section 6.

2 Problem Description

2.1 Notation

n: the number of the trains;

S: the number of the railway segments;

$m(k)$: the number of the tracks in segment k;

X_{ijk}: a binary variable that is equal to 1 if train i is assigned to track j in segment k and is equal to 0 otherwise;

Y_{ijk}: a binary variable that is equal to 1 if train i precedes train j on the same track in segment k and is equal to 0 otherwise;

S_{ik}: the starting time of train i in segment k;

T_{ijk}: the travel time of train i on track j in segment k;

C_{ik}: the completion time of train i in segment k;

L: a large constant.

2.2 The MTTSP

The MTTSP is described as follows: A set of n trains are going to pass S segments, where each segment has at least one track and some segments have multiple tracks. Tracks in each same segment are unrelated. One track can be used for only one train at a time. When a train completes a segment, if the planning track in the next segment is occupied by another train, it waits on the current track until the planning track is available. Due to these features, the MTTSP can be modeled as the blocking hybrid flow shop scheduling problem (BHFSP) [8-9]. The corresponding relationship between the MTTSP and the BHFSP is listed in Table 1.

Table 1. Corresponding relationship between the MTTSP and the BHFSP

Problem	Properties				
MTTSP	Train	Track	Segment	Travel time	Waiting on the track
BHFSP	Job	Machine	Stage	Processing time	Blocking on the machine

The MTTSP is supposed that: All the n trains are independent and ready at the initial time; All the tracks are available at time 0; For all the n trains, the travel times on each track are deterministic and known in advance; The transportation times between different segments are negligible. An example of MTTSP with 5 segments is illustrated in Fig. 1. The configuration of track numbers is {1, 3, 1, 2, 1}.

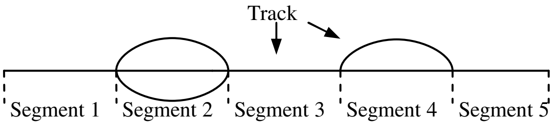

Fig. 1. An example of the MTTSP

The MTTSP can be formulated as follows:

$$\text{Minimize: } C_{\max} \tag{1}$$

$$\text{Subject to: } C_{\max} = \max C_{iS}, i = 1,2,\cdots,n \tag{2}$$

$$\sum_{j=1}^{m(k)} X_{ijk} = 1, i = 1,2,\cdots,n; k = 1,2,\cdots,S \tag{3}$$

$$C_{ik} = S_{ik} + \sum_{j=1}^{m(k)} X_{ijk} T_{ijk}, i = 1,2,\cdots,n; k = 1,2,\cdots,S \tag{4}$$

$$S_{ik} \geq C_{i,k-1}, i = 1,2,\cdots,n; k = 2,3,\ldots,S \tag{5}$$

$$S_{jk} \geq C_{ik} - L(1 - Y_{ijk}) \text{, for all the pairs } (i, j) \tag{6}$$

$$S_{jk} \geq S_{i,k+1} - L(1 - Y_{ijk}) \text{, for all the pairs } (i, j) \tag{7}$$

As for the above formulation: Eq. (1) denotes the objective function to minimize the completion time of all the trains as shown in (2); Eq. (3) ensures that a train can use exactly one track in each segment; Eq. (4) describes the computation of C_{ik}; Constraint (5) means that the trains pass all the segments from the first one to the last one in order; Constraint (6) means that a train can use the track until all its preceding train passes the track; Constraint (7) means that the trains will wait on the current track if the next segment is busy.

3 Estimation of Distribution Algorithm

The EDA [18] is a relatively new member of the evolutionary algorithms. The EDA employs explicit probability distributions to perform optimization procedure. It establishes a probability model of the most promising area by statistical information. Based on the probability model, it samples the search space to generate new solutions. To trace the more promising search area, it adjusts the probability model in each generation by using the information of some superior individuals of the new population.

The key content of the EDA is to estimate the probability distribution. Since the probability model is use to describe the distribution of the solution among the space, it is crucial to build a proper probability model. In addition, the probability model should be well adjusted to make the search procedure tract the promising search area. As a consequence, a reasonable mechanism should be design to adjust the model. Since different problems have different properties, both the probability model and the updating mechanism should be well adopted for a specific problem. Nevertheless, the EDA pays more attention to global exploration while its exploitation capability is relatively limited. Therefore, an effective EDA should balance the exploration and the exploitation abilities [13].

4 EDA for MTTSP

In this section, an effective EDA is presented for solving the MTTSP. First, the encoding, decoding, probability model, updating mechanism and local search scheme are introduced. Then, the flowchart of the EDA is illustrated.

4.1 Encoding and Decoding

Every individual of the population denotes a solution of the MTTSP. A solution is expressed by a permutation of the train priority for scheduling. In other words, an integer number sequence with the length of n determines the scheduling order of the trains. For example, a solution {2, 5, 3, 4, 1} represents that train 2 is scheduled first, and next are train 5, train 3 and train 4 in sequence. Train 1 is the last one to be scheduled.

In the decoding procedure, it assigns tracks in each segment for a certain train. After all the segments are assigned, the scheduling of a certain train completes and the next train in the solution sequence is considered. To be specific, for a given train i, the track of each segment is selected according to the track assignment rule as follows.

Step 1: $j = 1$. Choose track s if $T_{i1s} + a_s$ has the smallest value, where a_s denotes the available time of track s. Record s as the selected track for segment 1 and denote it as s^1. Update the completion time of train i (i.e., $C_{i,1}$) and a_s.

Step 2: $j = j + 1$. Decide the earliest starting time according to the completion time at the previous segment ($C_{i,j\text{-}1}$) and the earliest available time of each track (a_k). Obtain $\max\{a_k, C_{i,j\text{-}1}\}$.

Step 3: Choose track s if $T_{ijs} + \max\{a_s, C_{i,j\text{-}1}\}$ has the smallest value. Record s as the selected track for segment j, which is denoted as s^j.

Step 4: Update the completion time of train i (i.e., $C_{i,j}$) and the available time of track s (i.e., a_s). Update the available time of the track $s^{j\text{-}1}$, i.e., $a_{s^{j-1}} = S_{ij}$.

Step 5: Go to Step 2 until all the segments are assigned.

From step 4, it can be seen that the available time of the previous track is set as the starting time of the current track. The reason is that a train has to wait on the current track until the next track is available. With the above assignment rule, a feasible schedule is formed and the completion time of the schedule is obtained.

4.2 Probability Model and Updating Mechanism

The EDA produces the new individuals by sampling a probability model. In this paper, the probability model is designed as a probability matrix P. The element $p_{ij}(l)$ of the probability matrix P represents the probability that train j appears before or in position i of the solution sequence at generation l. The value of p_{ij} implies the priority of a train when deciding the scheduling order. For all i and j, p_{ij} is initialized as $p_{ij}(0) = 1/n$, which ensures that the whole solution space can be sampled uniformly.

In each generation of the EDA, the new individuals are generated via sampling the probability matrix P. For every position i, train j is selected with a probability p_{ij}. If train j has already appeared, it means the scheduling procedure of train j is completed. Then, the whole j-th column of probability matrix P will be set as zero and all the elements of P will be normalized to maintain that each row sums up to 1. In such a way, an individual is constructed until all the trains appear in the sequence. Finally, *P_Size* individuals are generated.

Next, it determines the superior sub-population that consists of the best *SP_Size* solutions, where $SP\_Size = \eta\% \cdot P\_Size$. The probability matrix P is updated according to the following equation:

$$p_{ij}(l+1)=(1-\alpha)p_{ij}(l)+\frac{\alpha}{i\times SP\_Size}\sum_{k=1}^{SP\_Size}I_{ij}^{k},\forall i,j. \tag{8}$$

where $\alpha\in(0,1)$ is the learning rate of *P* and I_{ij}^{k} is the following indicator function of the *k*-th individual in the superior sub-population.

$$I_{ij}^{k}=\begin{cases}1, \text{if train } j \text{ appears before or in position } i\\ 0, \text{else}\end{cases} \tag{9}$$

4.3 Local Search Scheme

To enhance the exploitation of the EDA, several different local search operators for the shop scheduling problems [17] are employed, which are described as follows:

Swap: Randomly select two different elements from a sequence and then swap them.

Insert: Randomly choose two different elements from a sequence and then insert the back one before the front one.

Inverse: Invert the subsequence between two different random positions of a sequence.

In each local search step, the above three operators are performed in sequence and the new individual will replace the old one if it has a smaller completion time. The above procedure is applied on the best individual of the current population *ls* times in every generation, where *ls* represents the intensity of the local search.

4.4 Procedure of EDA for MTTSP

With the design above, the procedure of EDA for solving the MTTSP is illustrated in Fig. 2.

From the above procedure, it can be seen that the EDA samples the probability model to generate the new individuals and learns the information of the superior ones to update the probability model. Meanwhile, the local search strategy is performed to enhance local exploitation capability. The algorithm stops when the maximum decoding time *MAX_D* is satisfied.

4.5 Computation Complexity Analysis

For each generation of the EDA, its computational complexity can be roughly analyzed as follows.

For the updating process, first it is with the computational complexity *O*(*P_Size*log*P_Size*) by using the quick sorting method to select the best *SP_Size* individuals from population; Then, it is with the complexity *O*[*n*(*SP_Size*+*n*)] to update

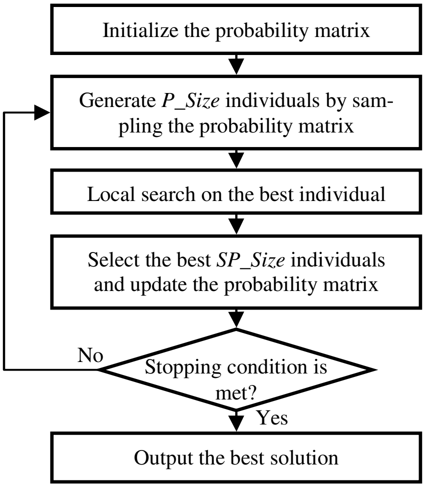
Initialize the probability matrix
Generate *P_Size* individuals by sampling the probability matrix
Local search on the best individual
Select the best *SP_Size* individuals and update the probability matrix
No
Stopping condition is met?
Yes
Output the best solution

Fig. 2. Procedure of the EDA for solving the MTTSP

all the n×n elements of P. Besides, local search is performed on the best individual of the population ls times with the complexity $O(ls{\times}n)$. For the sampling process, the element for every position is generated with the roulette strategy by sampling the probability matrix P to obtain a new individual. To generate a train priority sequence, it is with the complexity $O(n^2)$. Thus, the computational complexity for generating *P_Size* individuals is $O[n^2P\_Size]$. It can be seen that the complexity of the EDA is not large and the algorithm could be efficient.

5 Simulation and Comparisons

5.1 Benchmark Instances

In this section, three instances with different scales taken from literatures [8-9] are used to test the performance of the EDA and for comparison. The parameters of the instances are provided in Table 2. Besides, the travel times of each train on every track are listed in Tables 3-5 for each instance.

Table 2. Parameters of the instances

Instance	Train	Track	Segment	Track configuration
1	12	9	3	{3, 2, 4}
2	10	14	9	{1, 2, 1, 2, 1, 3, 1, 2, 1}
3	12	10	4	{3, 3, 2, 2}

Table 3. Travel time of instance 1

Train	Track								
	1	2	3	4	5	6	7	8	9
	Segment								
	1			2		3			
1	2	2	3	4	5	2	3	2	3
2	4	5	4	3	4	3	4	5	4
3	6	5	4	4	2	3	4	2	5
4	4	3	4	6	5	3	6	5	8
5	4	5	3	3	1	3	4	6	5
6	6	5	4	2	3	4	3	9	5
7	5	2	4	4	6	3	4	3	5
8	3	5	4	7	5	3	3	6	4
9	2	5	4	1	2	7	8	6	5
10	3	6	4	3	4	4	8	6	7
11	5	2	4	3	5	6	7	6	5
12	6	5	4	5	4	3	4	7	5

Table 4. Travel time of instance 2

Train	Track													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	Segment													
	1	2		3	4		5	6			7	8		9
1	3	1	2	2	1	3	4	2	3	5	4	6	9	5
2	3	2	4	4	2	3	3	3	4	2	5	4	6	4
3	4	2	3	2	2	1	2	3	6	9	2	4	5	4
4	2	2	5	5	4	2	3	1	3	6	2	3	7	7
5	3	1	2	3	1	4	5	4	3	4	5	6	4	5
6	2	3	6	1	3	5	2	6	2	3	2	5	6	3
7	1	3	3	1	3	2	5	3	4	6	3	4	8	7
8	1	4	2	3	4	3	6	3	2	7	5	2	4	4
9	4	1	2	3	2	3	4	2	3	5	4	6	7	5
10	2	2	3	4	2	3	3	2	4	4	3	5	6	6

5.2 Parameter Setting

The EDA has four parameters: *P_Size* (the population size), η (the parameter associated with the superior sub-population), α (the learning rate of *P*), and *ls* (the intensity of local search). To investigate the influence of these parameters on the performance of the EDA, the Taguchi method of design-of-experiment (DOE) [18] is implemented base on instance 1. The combinations of different values of these parameters are listed in Table 6.

Table 5. Travel time of instance 3

Train	Track									
	1	2	3	4	5	6	7	8	9	10
	Segment									
	1			2			3		4	
1	45	48	50	35	35	30	30	35	25	26
2	45	50	45	35	36	35	35	34	25	30
3	50	45	46	35	36	36	31	34	30	31
4	50	48	48	34	38	35	32	33	27	31
5	45	46	48	30	35	50	34	32	28	31
6	45	45	45	30	35	50	33	32	30	26
7	47	50	47	31	30	35	35	31	29	25
8	50	45	48	32	30	34	34	30	24	27
9	48	46	46	33	34	30	34	30	25	25
10	45	47	47	33	33	30	35	34	32	26
11	46	50	45	34	30	50	30	35	31	25
12	48	50	47	35	31	35	32	30	25	30

Table 6. Combinations of parameter values

Parameters	Factor level			
	1	2	3	4
P_Size	20	30	40	50
η	10	20	30	40
α	0.1	0.2	0.3	0.4
ls	5	10	15	20

In all the experiments, it sets *Max_D* = 10000 as the stopping condition. For each parameter combination, the EDA is run 20 times independently and the average result obtained is calculated as the response variable (RV) value. Besides, the number of factor levels is set as 4. Accordingly, the orthogonal array $L_{16}(4^4)$ is used. The orthogonal array and the obtained RV values are listed in Table 7.

According to the orthogonal table, the trend of each factor level is illustrated in Fig. 3. Then, the average response value of each parameter is figured out in Table 8 to analyze the significance rank.

From Table 8, it can be seen that the intensity of local search *ls* is the most significant one among the four parameters. A large value of *ls* may help the EDA enhance the searching capability. In addition, the significance of the population size *P_Size* ranks the second. With a fixed maximum number of evaluation times, a small population size allows more generations so that it may provide a deeper search. According to above analysis, a better choice of the parameter combination is suggested as *P_Size*=20, η=30, α=0.3, *ls*=20 and *MAX_D*=10000, which is also used for the following experiments.

Table 7. Orthogonal array and ARV values

Experiment Number	Factor				RV
	P_Size	*η*	*α*	*ls*	
1	1	1	1	1	24.1
2	1	2	2	2	24.0
3	1	3	3	3	23.6
4	1	4	4	4	23.9
5	2	1	2	3	24.0
6	2	2	1	4	23.8
7	2	3	4	1	24.1
8	2	4	3	2	24.0
9	3	1	3	4	23.9
10	3	2	4	3	23.9
11	3	3	1	2	24.0
12	3	4	2	1	24.1
13	4	1	4	2	24.0
14	4	2	3	1	24.2
15	4	3	2	4	23.9
16	4	4	1	3	24.0

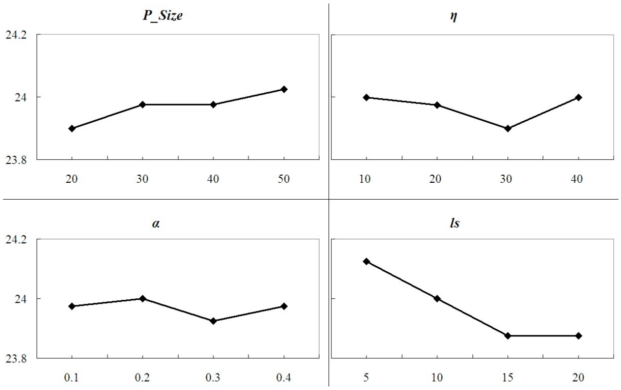

Fig. 3. Factor level trends of the parameters

Table 8. Average response values

Level	*P_Size*	*η*	*α*	*ls*
1	**23.9**	24	23.975	24.125
2	23.975	23.975	24	24
3	23.975	**23.9**	**23.925**	**23.875**
4	24.025	24	23.975	**23.875**
Delta	0.125	0.1	0.075	0.25
Rank	2	3	4	1

5.3 Simulation Results and Comparisons

The EDA is compared with the HPSO [8-9] for solving the MTTSP. For each instance, the algorithm runs 10 times as in literatures [8-9]. The best and average completion times as well as the CPU time are used as the performance measure. The results are listed in Table 9, where the results of the comparative algorithm are directly taken from the literatures [8-9].

Table 9. Simulation results

Instance	HPSO			EDA		
	Best	Average	CPU(s)[a]	Best	Average	CPU(s)[b]
1	25	26.2	95	**23**	**23.5**	**0.047**
2	72	72	<1	**70**	**70**	**0.094**
3	301	N/A	190	**297**	**297**	**0.062**

a. Pentium 1.6 GHz CPU.

b. Core i5 2.3 GHz CPU.

From Table 9, it can be seen that the EDA outperforms the HPSO in solving all the three instances. The best and average results obtained by the EDA are better than that by the HPSO, which means that the EDA is more effective for solving the MTTSP. Besides, the CPU time employed by the EDA is less than 0.1 second for every instance. It demonstrates that the EDA is much more efficient than the HPSO.

6 Conclusions

In this paper, an effective estimation of distribution algorithm is presented for solving the multi-track train scheduling problem. The permutation of train priority is employed to represent the individual of the EDA. A track assignment rule is designed for decoding the individual into a feasible schedule. A probability model with its updating mechanism is presented. Based on the design-of-experiment method, a set of suitable parameter values is suggested. Simulation results based on some instances and comparisons with the existing algorithms demonstrate the effectiveness and efficiency of the EDA. As for the future work, it is worth to design EDA-based algorithm for other scheduling problems, such as the semi-conductor manufacturing scheduling.

Acknowledgments. This work was supported by the National Key Basic Research and Development Program of China (2013CB329503), the National Science Foundation of China (61174189) and the Doctoral Program Foundation of Institutions of Higher Education of China (20130002110057).

References

1. Gholami, O., Sotskov, Y.N., Werner, F.: Job-Shop Problems with Objectives Appropriate to Train Scheduling in a Single-Track Railway. In: Proceedings of 2nd International Conference on Simulation and Modeling Methodologies, Technologies and Applications, Rome, pp. 425–430 (2012)
2. Ghoseiri, K., Szidarovszky, F., Asgharpour, M.J.: A Multi-Objective Train Scheduling Model and Solution. Transportation Research Part B: Methodological 38(10), 927–952 (2004)
3. Dorfman, M.J., Medanic, J.: Scheduling Trains on a Railway Network Using a Discrete Event Model of Railway Traffic. Transportation Research Part B: Methodological 38(1), 81–98 (2004)
4. Zhou, X., Zhong, M.: Bicriteria Train Scheduling for High-Speed Passenger Railroad Planning Applications. European Journal of Operational Research 167(3), 752–771 (2005)
5. D'ariano, A., Pacciarelli, D., Pranzo, M.: A Branch and Bound Algorithm for Scheduling Trains in a Railway Network. European Journal of Operational Research 183(2), 643–657 (2007)
6. Burdett, R.L., Kozan, E.: A Disjunctive Graph Model and Framework for Constructing New Train Schedules. European Journal of Operational Research 200(1), 85–98 (2010)
7. Liu, S.Q., Kozan, E.: Scheduling Trains as a Blocking Parallel-Machine Job Shop Scheduling Problem. Computers & Operations Research 36(10), 2840–2852 (2009)
8. Zhang, Q.L., Chen, Y.S.: Train Scheduling Problem and Its Solution Based on Hybrid Particle Swarm Optimization Algorithm. China Mechanical Engineering 24(14), 1916–1922 (2013)
9. Zhang, Q.L., Chen, Y.S.: Hybrid Particle Swarm Optimization Algorithm for Hybrid Flow Shop Scheduling Problem with Blocking. Information and Control 42(2), 252–257 (2013)
10. Wang, S.Y., Wang, L., Fang, C., Xu, Y.: Advances in Estimation of Distribution Algorithms. Control and Decision 27(7), 961–966 (2012)
11. Wang, L., Fang, C.: An Effective Estimation of Distribution Algorithm for the Multi-Mode Resource-Constrained Project Scheduling Problem. Computer & Operations Research 39(2), 449–460 (2012)
12. Wang, L., Wang, S.Y., Liu, M.: A Pareto-Based Estimation of Distribution Algorithm for the Multi-Objective Flexible Job-Shop Scheduling Problem. International Journal of Production Research 51(12), 3574–3592 (2013)
13. Wang, L., Wang, S.Y., Xu, Y., Zhou, G., Liu, M.: A Bi-Population Based Estimation of Distribution Algorithm for the Flexible Job-Shop Scheduling Problem. Computers & Industrial Engineering 62(4), 917–926 (2012)
14. Wang, S.Y., Wang, L., Liu, M., Xu, Y.: An Effective Estimation of Distribution Algorithm for the Flexible Job-Shop Scheduling Problem with Fuzzy Processing Time. International Journal of Production Research 51(12), 3778–3793 (2013)
15. Wang, S.Y., Wang, L., Liu, M., Xu, Y.: An Effective Estimation of Distribution Algorithm for Solving the Distributed Permutation Flow-Shop Scheduling Problem. International Journal of Production Economics 145(1), 387–396 (2013)
16. Larranaga, P., Lozano, J.A.: Estimation of distribution algorithms: A new tool for evolutionary computation. Springer, Netherlands (2002)
17. Wang, L.: Shop scheduling with genetic algorithms. Tsinghua University & Springer Press, Beijing (2003)
18. Montgomery, D.C.: Design and analysis of experiments. John Wiley & Sons, Arizona (2005)