Proceedings of the 10th
World Congress on Intelligent Control and Automation
July 6-8, 2012, Beijing, China

A Compact Estimation of Distribution Algorithm for Solving Hybrid Flow-shop Scheduling Problem*

Shengyao Wang, Ling Wang and Ye Xu

Tsinghua National Laboratory for Information Science and Technology (TNList)
Department of Automation, Tsinghua University, Beijing, P.R. China

wangshengyao@tsinghua.org.cn; wangling@mail.tsinghua.edu.cn; xuye05@mails.tsinghua.edu.cn

***Abstract* - According to the characteristics of the hybrid flow-shop scheduling problem (HFSP), the permutation based encoding and decoding schemes are designed and a probability model for describing the distribution of the solution space is built to propose a compact estimation of distribution algorithm (cEDA) in this paper. The algorithm uses only two individuals by sampling based on the probability model and updates the parameters of the probability model with the selected individual. The cEDA is efficient and easy to implement due to its low complexity and comparatively few parameters. Simulation results based on some widely-used instances and comparisons with some existing algorithms demonstrate the effectiveness and efficiency of the proposed compact estimation of distribution algorithm. The influence of the key parameter on the performance is investigated as well.**

Index Terms* - *Hybrid flow-shop scheduling, estimation of distribution algorithm, probability model, compact algorithm

I. INTRODUCTION

The hybrid flow-shop scheduling problem (HFSP) [1] is of wide industrial background, and many real industrial manufacturing problems in fields of textile, paper and electronics industries can be modeled as the HFSP [2]. Meanwhile, the HFSP with multiple stages and jobs is strongly NP-hard [3]. So, it is of strong academic significance and engineering application value to study the HFSP, especially to develop effective and efficient solution algorithms.

Since the HFSP was proposed in 1954 [4], many approaches have been proposed, including exact methods, heuristics and meta-heuristics. Due to the complexity of the HFSP, exact algorithms such as branch and bound (B&B) [5] can only be used to solve small-scaled problems. Although the heuristics such as dispatching rules [6] and divide-and-conquer strategy [7] are very efficient, the quality of the obtained solution is often not satisfactory, especially for the problems with a large number of jobs or machines or stages.

During the past two decades, meta-heuristics have gained increasing research, especially in the field of manufacturing and scheduling, which makes it possible to achieve satisfactory schedules with reasonable computational effort for the large-scaled problems. Among meta-heuristics, genetic algorithm (GA) [8-9], particle swarm optimization (PSO) [10], ant colony optimization (ACO) [11], tabu search (TS) [12], differential evolution (DE) [13], and shuffled frog leaping algorithm (SFLA) [14] have already been applied to solve the HFSP, while it still needs to develop other kinds of meta-heuristics for the HFSP.

As a relatively new population-based evolutionary algorithm, estimation of distribution algorithm (EDA) [15] has gained increasing study and applications in several fields during recent years. The earliest model of the EDA is population-based incremental learning (PBIL). According to the complexity of the model, the EDA can be classified as univariate model, bivariate model and multivariate model. The PBIL, univariate marginal distribution algorithm (UMDA) and compact GA (CGA) are univariate models, while mutual information maximization for input clustering (MIMIC), combining optimizers with mutual information trees (COMIT) and bivariate marginal distribution algorithm (BMDA) are bivariate models. The factorized distribution algorithm (FDA), extended compact GA (ECGA) and Bayesian optimization algorithm (BOA) are multivariate models. Please refer [15] for more details about the EDA.

So far the EDA has been applied to a variety of academic and engineering optimization problems, such as feature selection, cancer classification, quadratic assignment problem, machinery structure design, nurse rostering, and etc [16]. However, to the best of our knowledge, there is no research work about the EDA for solving the HFSP. In this paper, we will propose a compact EDA (cEDA) for solving the HFSP. The cEDA uses only two individuals in each generation, which are generated by sampling a probability model. In addition, an updating mechanism is proposed to update the probability model. The influence of the key parameter on the performance is investigated. Simulation results based on some typical instances and the statistical comparisons with some existing methods demonstrate the effectiveness of the proposed algorithm.

The remainder of the paper is organized as follows. The HFSP is formulated in section II. In Section III the basic EDA is introduced, and the framework of cEDA for solving the HFSP is proposed in Section IV. Simulation results and comparisons are provided in Section V, and the influence of the key parameter on the performance is investigated as well. Finally, we end the paper with some conclusions in Section VI.

* This research is partially supported by National Science Foundation of China (61174189, 60834004 and 61025018), Doctoral Program Foundation of Institutions of Higher Education of China (20100002110014), the National Key Basic Research and Development Program of China (No.2009CB320602) and National Science and Technology Major Project of China (No.2011ZX02504-008).

II. Hybrid Flow-shop Scheduling Problem

Generally, the HFSP consists of a set of n jobs that are to be processed in s stages, where each stage has at least one machine and some stages have multiple machines. Machines in each same stage are unrelated. Each job should be processed in all the stages, and each job can be processed by any one of machines in each stage.

Typically, the HFSP is supposed that: All the n jobs are independent and available to be processed at the initial time; Buffers between stages are unlimited; one machine can process only one operation and one job can be processed at only one machine at a time; the releasing time of all machines is not considered or set as 0; For all the n jobs, the processing times in each machine are deterministic and known in advance; The time between different machines for transportation is negligible; Once an operation is started, it cannot be interrupted. In Fig. 1, it illustrates an example of HFSP with 3 stages.

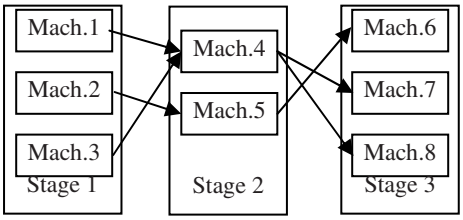

Fig. 1 An example of HFSP.

The HFSP can be formulated as follows [17]:

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

Subject to:

$$C_{\max} = \max C_{ik}, k = 1,2,\ldots,s; i = 1,2,\ldots,n \tag{2}$$

$$C_{ik} = S_{ik} + P_{ik} \tag{3}$$

$$\sum_{i=1}^{m(k)} X_{jik} = 1 \tag{4}$$

$$S_{ik} \geq C_{i,k-1}, k = 2,3,\ldots,s \tag{5}$$

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

$$S_{jk} \geq C_{ik} - (1 - M)Y_{ijk} \text{ , for all the pairs } (i, j) \tag{7}$$

$$S_{i1} \geq R_i, i = 1,2,\ldots,n \tag{8}$$

where, n is the total number of jobs, s is the total number of stages, $m(k)$ is the number of the machines at stage k, X_{ijk} is a binary variable (it is equal to 1 if job j is assigned to machine i at stage k; otherwise, it is equal to 0), Y_{ijk} is a binary variable (it is equal to 1 if job i precedes job j at stage k; otherwise, it is equal to 0), R_i denotes the releasing time of job i, S_{ik} denotes the starting time of job i at stage k, P_{ik} denotes the processing time of job i at stage k, C_{ik} denotes the completing time of job i, and M is a large constant.

As for the above formulation: Eqn. (1) denotes the objective function to minimize the makespan of the HFS, i.e., the maximum completion time of all the jobs as shown in Eqn. (2); Eqn. (3) describes the computation of C_{ik}; Eqn. (4) ensures that one job can be processed exactly on one machine at each stage; Constraints (5) and (6) ensure that one machine can process only one job at one time; Constraint (7) means that one job cannot be processed until its preceding job is finished; Constraint (8) bounds the starting time of a job.

III. Estimation of Distribution Algorithm

Estimation of distribution algorithms (EDA) is a new paradigm in the field of evolutionary computation, which employs explicit probability distributions in optimization [13]. Compared with the genetic algorithm, the EDA reproduces new population implicitly instead of the crossover and mutation operators. In the EDA, a probability model of the most promising area is built by statistical information based on the searching experience, and then the probability model is used for sampling to generate the new individuals. Meanwhile, the probability model is updated in each generation with the potential individuals of the new population. In such an iterative way, the population evolves and finally satisfactory solutions can be obtained.

The critical step of the above procedure is to estimate the probability distribution. The EDA makes use of the probability model to describe the distribution of the solution space and the updating process reflects the evolutionary trend of the population. Due to the difference of problem types, a proper probability model and an updating mechanism should be well developed to estimate the underlying probability distribution.

IV. The Compact EDA for HFSP

In this section, we will propose a compact estimation of distribution algorithm (cEDA) to solve the HFSP. First, we will introduce the encoding, decoding, probability model and updating mechanism. Then, we will present the framework of the cEDA.

A. Encoding and Decoding

In the algorithm, a new combinatorial encoding and decoding scheme is devised for the HFSP. A solution can be expressed by an integer number sequence with the length of n, which determines the processing order of the first stage. For example, a solution {6, 5, 2, 3, 1, 4} represents that job 6 is processed first in the first stage, and next are job 5, job 2, job 3, job 1 in sequence. Job 4 is the last job to be processed.

To decode a sequence is to divide the jobs to the machines at each stage so as to form a feasible schedule and calculate the makespan of the schedule. For the HFSP, the decoding can be divided into two parts: to decide the jobs order and the machines assignment.

For the jobs order, the algorithm decides the processing order of the first stage referring to the sequence order given. From stage 2 on, the algorithm decides the processing order of the current stage according to the completion time of each job from the previous stage in a non-decreasing order. If some jobs have the same completion time, then the algorithm will randomly determine the processing orders of these jobs.

As for the machines assignment, based on the first available machine rule [18], we proposed the machine assigning rule as follows:

Step 1: $j = 1$;

Step 2: For each job i, decide the earliest start time of each job i according to the completion time of each job at the previous stage $C_{i,j-1}$ and the earliest release time of each machine r_k, and compute $\max(r_k, C_{i,j-1})$. If $j = 1$, $C_{i,j-1} = 0$;

Step 3: For each job i, choose the machine s if $P_{i,j,s} + \max(r_s, C_{i,j-1})$ has the smallest value. Update the completion time of job i and the release time of machine s at stage j;

Step 4: $j = j + 1$;

Step 5: Go to Step 2 until all jobs at all stages are scheduled.

Consider the following problems with 3 stages and 6 jobs as showed in Table I. For the solution {6, 5, 2, 3, 1, 4}, it can get a schedule with the Gantt chart as shown in Fig. 2.

TABLE I
PROCESSING TIME

stage	1		2		3	
	machine					
job	1	2	3	4	5	6
1	2	2	4	3	1	1
2	2	3	2	3	2	1
3	4	3	2	3	2	2
4	2	3	1	1	2	1
5	4	2	2	1	4	5
6	1	2	3	2	3	6

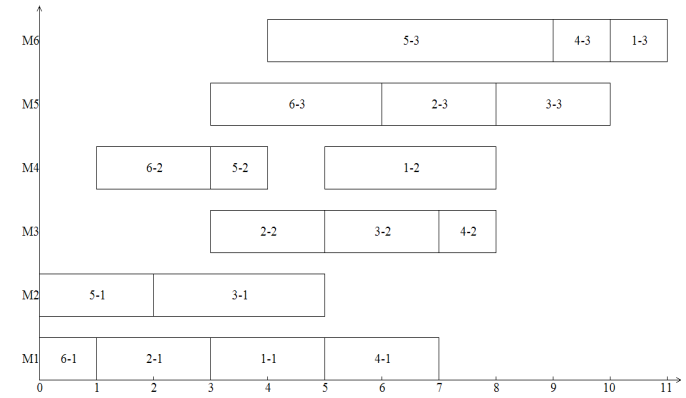

Fig. 2 The Gantt chart of the solution.

B. Probability Model and Updating Mechanism

Different from the GA that produces offspring through crossover and mutation operators, the EDA does it by sampling according to a probability model which has a great effect on the performances of the EDA. 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 job j appears before or in position i of the solution sequence at generation l. The value of p_{ij} refers to the importance of a job when deciding the jobs order. For all i and j, p_{ij} is initialized to $p_{ij}(0) = 1/n$, which ensures that the whole solution space can be sampled uniformly.

In each generation of the EDA, the new individuals are generated via sampling according to the probability matrix P. For every position i, job j is selected with the probability p_{ij}. If job j has already appeared, the whole j-th column of probability matrix P will be set as zero. In such a way, an individual is constructed until all the jobs appear in the sequence and its makespan is calculated.

Similar as cGA [19], the cEDA generates only two individuals in each generation and uses the individual X that has the smaller makespan to update the probability matrix P according to the following equation:

$$p_{ij}(t+1) = (1-\alpha)\, p_{ij}(t) + \frac{\alpha}{i} I_{ij}, (i, j = 1,2,\cdots,n) . \quad (9)$$

where $\alpha \in (0,1)$ is the learning speed, I_{ij} is the indicator function of the individual X that is defined as follows:

$$I_{ij} = \begin{cases} 1, \text{if job } j \text{ appears before or in position } i \\ 0, \text{else} \end{cases} . \quad (10)$$

C. Procedure of cEDA for HFSP

With the design above, the procedure of cEDA for solving the HFSP is illustrated in Fig. 3.

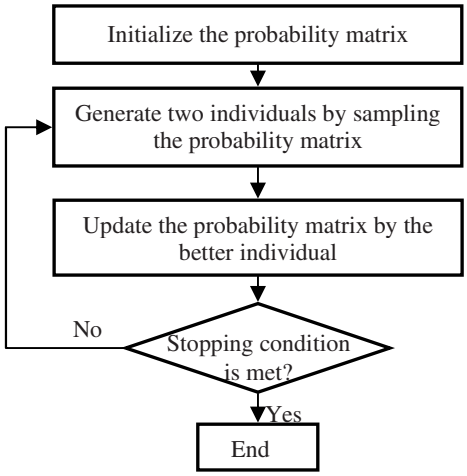

Fig. 3 The procedure of cEDA for HFSP.

From the above procedure, it can be seen that the cEDA samples the probability model to generate two individuals and learns the information of the better one to update the probability model. The algorithm stops when the maximum number of generations *Gen* is satisfied. Compared to the basic EDA, the cEDA is easy to use due to its low complexity and comparatively few parameters.

D. Computation Complexity Analysis

To obtain a new individual by sampling the probability model, every gene is generated with the roulette strategy according to the matrix P. In each generation, the two individuals can be generated with the complexity $O(n^2)$, and the probability matrix P is updated with the complexity $O(n^2)$. So, the complexity of the cEDA is $O(n^2 Gen)$, which is not large. It also can be seen from the numerical simulation in the next section that the cEDA is very efficient in solving the HFSP.

V. SIMULATION AND COMPARISONS

In this section, two problems with different scales taken from literatures [8-9] are used to test the performance of the

proposed algorithm and for comparison. Problem 1 is an example of production scheduling problem for metalworking workshop in a car engine plant [8]. Problem 1 has 12 jobs and 3 stages, and the numbers of machines in each stage are 3, 2 and 4, respectively, and the processing time of each job in every machine is listed in Table II. Problem 2 is an example of scheduling problem for the steel production [9]. Problem 2 has 12 jobs and 4 stages, and the numbers of machines in each stage are 3, 3, 2 and 2, respectively, and the processing time of each job in every machine is listed in Table III.

TABLE II
PROCESSING TIME OF PROBLEM I

stage	1			2		3			
	machine								
job	1	2	3	4	5	6	7	8	9
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 III
PROCESSING TIME OF PROBLEM II

stage	1			2			3		4	
	machine									
job	1	2	3	4	5	6	7	8	9	10
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

A. Parameter Setting

Experiments show that the parameters are related to the performance of the EDA based algorithm [20] and need to be set suitably. As the cEDA has only one key parameter α to set, we investigate the influence of the learning speed α on the performance of the algorithm by using problem 2. In all the experiments, we set *Gen* = 2000, i.e. 4000 evaluation times as the stopping condition.

For each parameter condition, the cEDA is run 20 times, and the average results (AR) are listed in Table IV. According to Table IV, we illustrate the trend of influence in Fig. 4.

TABLE IV
INFLUENCE OF LEARNING SPEED

α	0.001	0.005	0.01	0.05	0.1	0.2
AR	298.5	298.4	298.2	298.35	298.45	298.6

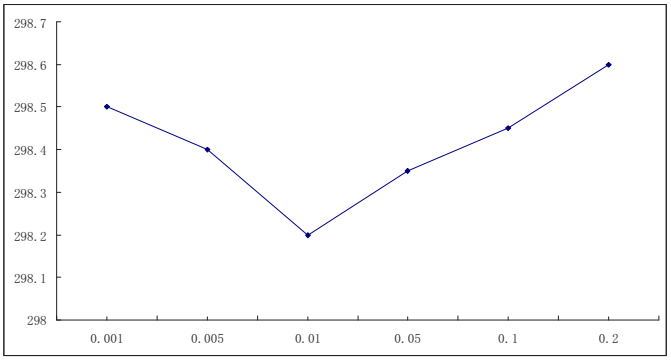

Fig. 4 The trend of influence.

From Fig. 4 it can be seen that the learning speed of the probability model is crucial to the performance of cEDA. Small value of α could lead to slow convergence while large value could lead to premature convergence. Therefore, we set α = 0.01, *Gen* = 2000 in the following simulation.

B. Simulation Results and Comparisons

First, we use problem 1 to test the performance of the cEDA. The algorithm is run with 10 times independently, and the results are listed in Table V, where the results of GA [8], DE [13] and SFLA [14] are directly from literature. Furthermore, the evaluation times of GA, DE and SFLA are all 10000 while cEDA 4000.

From Table V, it can be seen that cEDA can find the solution with makespan 23, which has never been found in the literature [8], [13] and [14]. The Gantt chart is illustrated in Fig. 5. Besides, the average result of our cEDA outperforms GA, DE and SFLA with less evaluation times. So, it is concluded that the cEDA is more effective and efficient in solving the HFSP.

TABLE V
COMPARISON WITH GA, DE AND SFLA FOR PROBLEM I

Test No.	1	2	3	4	5	6	7	8	9	10
GA	30	27	26	27	29	27	26	27	26	28
DE	24	24	24	24	24	25	24	24	24	25
SFLA	24	24	24	24	24	24	24	24	24	24
cEDA	24	24	24	24	24	24	**23**	**23**	24	24

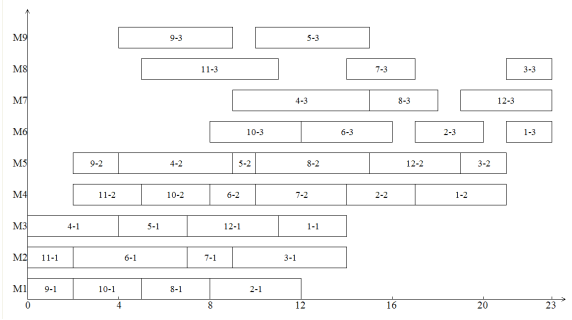

Fig. 5 Gantt chart of the best solution for problem 1.

Next, we use problem 2 to test the performance of the cEDA. The algorithm is run with 10 times independently, and the results are listed in Table VI, where the results of GA [9], DE [13] and SFLA [14] are directly from literature.

Furthermore, the evaluation times of GA, DE and SFLA are all 18000 while cEDA 4000.

TABLE VI
COMPARISON WITH GA, DE AND SFLA FOR PROBLEM II

Test No.	1	2	3	4	5	6	7	8	9	10
GA	347	-	-	-	-	-	-	-	-	-
DE	313	319	313	302	302	315	315	319	302	299
SFLA	297	313	297	297	313	310	313	311	316	306
cEDA	298	298	298	299	297	299	297	298	297	299

From Table VI, it can be seen that the cEDA outperforms GA, DE and SFLA with the less evaluation times. Besides, the run time of the cEDA is less than one second, which proves that the proposed algorithm is efficient.

All in all, the proposed cEDA is effective, efficient and robust in solving the HFSP, and it has superior performance to the existing algorithms.

VI. CONCLUSION

In this paper, an EDA based algorithm was proposed for solving the HFSP. We designed a permutation based encoding and decoding schemes and built a probability model for describing the distribution of the solution space. The cEDA employed only two individuals by sampling based on the probability model and updated the parameters of the probability model with the selected individual. The proposed algorithm was easy to implement in the paper due to its low complexity and comparatively few parameters. The cEDA was effective and efficient in solving the HFSP, which was demonstrated by simulation results and comparisons. The future work is to design effective EDA for the multi-objective HFSP and the problem with lot streaming.

REFERENCES

[1] L. Wang, *Shop Scheduling with Genetic Algorithms*, Beijing: Tsinghua University Press, 2003, pp. 138-145.

[2] L. Wang, G. Zhou, Y. Xu, and Y.-H. Jin, "Advances in the Study on Hybrid Flow-shop Scheduling," *Control and Instrument in Chemical Industry*, vol. 38, no. 1, pp. 1-8, 2011.

[3] J.-N.-D. Gupta, "Two-stage hybrid flow shop scheduling problem," *Journal of the Operational Research Society*, vol. 39, no. 4, pp. 359-364, 1988.

[4] S.-M. Johnson, "Optimal two and three-stage production schedules with setup times included," *Naval research logistics quarterly*, vol. 1, no. 1, pp. 61-68, 1954.

[5] S.-A. Brah and J.-L. Hunsucker, "Branch and bound algorithm for the flow-shop with multiple processors," *European Journal of Operational Research*, vol. 51, no. 1, pp. 88-99, 1991.

[6] S.-N. Kadipasaoglu, W. Xiang, and B.-M. Khumawala, "A comparison of sequencing rules in static and dynamic hybrid flow systems," *International Journal of Production Research*, vol. 35, no. 5, pp. 1359-1384, 1997.

[7] G. Vairaktarakis, M. Elhafsi, "The use of flowlines to simplify routing complexity in two-stage flowshops," *IIE Transactions*, vol. 32, no. 8, pp. 687-699, 2000.

[8] H.-R. Zhou, W.-S. Tang, and Y.-H. Wei, "Optimize flexible flow-shop scheduling using genetic algorithm," *Computer Engineering and Applications*, vol. 45, no. 30, pp. 224-226, 2009.

[9] J.-S. Cui, T.-K. Li, and W.-X. Zhang, "Hybrid flowshop scheduling model and its genetic algorithm," *Journal of University of Science and Technology Beijing*, vol. 27, no. 5, pp. 623-626, 2005.

[10] C.-T. Tseng and C.-J. Liao, "A particle swarm optimization algorithm for hybrid flowshop scheduling with multiprocessor tasks," *International Journal of Production Research*, vol. 46, no. 17, pp. 4655-4670, 2008.

[11] K.-C. Ying and S.-W. Lin, "Multiprocessor task scheduling in multistage hybrid flowshops: an ant colony system approach," *International Journal of Production Research*, vol. 44, no. 16, pp. 3161-3177, 2006.

[12] R. Logendran, P. deSzoeke, and F. Barnard, "Sequence-dependent group scheduling problems in flexible flow shops," *International Journal of Production Economics*, vol. 102, no. 1, pp. 66-86, 2006.

[13] Y. Xu, L. Wang, "Differential evolution algorithm for hybrid flow-shop scheduling problems," *Journal of Systems Engineering and Electronics*, vol. 22, no. 5, pp. 794-798, 2011.

[14] Y. Xu, L. Wang, G. Zhou, and S.-Y. Wang, "An effective shuffled frog leaping algorithm for solving hybrid flow-shop scheduling problem," *Lecture Notes in Computer Science*, vol. 6838, pp. 560-567, 2011.

[15] P. Larranaga and J.-A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*, Boston: Kluwer Press, 2002.

[16] S.-D. Zhou and Z.-Q. Sun, "A survey on estimation of distribution algorithms," *Acta Automatica Sinica*, vol. 33, no. 2, pp. 113-124, 2007.

[17] C.-D. Paternina-Arboleda, J.-R. Montoya-Torres, M.-J. Acero-Dominguez, and M.C. Herrera-Hernandez, "Scheduling jobs on a k-stage flexible flow-shop," *Annals of Operations Research*, vol. 164, no. 1, pp. 29-40, 2008.

[18] S.-A. Brah and L.-L. Luan, "Heuristics for scheduling in a flowshop with multiple processors," *European Journal of Operational Research*, vol. 113, no. 1, pp. 113-122, 1999.

[19] G.-R. Harik, F.-G. Lobo, and D.-E. Goldberg, "The compact genetic algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 287-297, 1999.

[20] L. Wang, S.-Y. Wang, and C. Fang, "A hybrid distribution estimation algorithm for solving multidimensional knapsack problem," *Control and Decision*, vol. 26, no. 8, pp. 1121-1125, 2011.