Proceedings of the 32nd Chinese Control Conference
July 26-28, 2013, Xi'an, China

An Estimation of Distribution Algorithm for Solving Hybrid Flow-shop Scheduling Problem with Stochastic Processing Time

WANG Shengyao[1], WANG Ling[1], XU Ye[1]

1. Tsinghua National Laboratory for Information Science and Technology (TNList), Department of Automation, Tsinghua University, Beijing 100084, China
E-mail: wangshengyao@tsinghua.org.cn

Abstract: In this paper, an effective estimation of distribution algorithm (EDA) is proposed to solve the hybrid flow-shop scheduling problem with stochastic processing time. Considering the effectiveness and robustness of a schedule, the schedule objective is to minimize the makespan of the initial scenario as well as the deviation of the makespan between all stochastic scenarios and the initial one. In the proposed EDA, a bi-objective evaluation function is employed to evaluate the individuals of the population. A probability model is presented to describe the probability distribution of the solution space. A mechanism is provided to update the probability model with the superior individuals. By sampling the probability model, new individuals can be generated among the search region with the promising solutions. Numerical testing results based on some well known benchmark instances are provided. The comparisons with the existing genetic algorithm demonstrate the effectiveness and robustness of the proposed EDA.

Key Words: Hybrid flow-shop scheduling problem, stochastic processing time, estimation of distribution algorithm, probability model, robust scheduling

1 Introduction

Flexible manufacturing system (FMS) is an automatic machinery manufacturing system which can adapt to the changes of processing objects. Production scheduling plays an important role in the FMS. Developing effective and efficient algorithms for scheduling problems is significant to improve the processing efficiency and the economic benefits. The hybrid flow-shop scheduling problem (HFSP) [1] is an extension of the classical flow-shop scheduling problem for the FMS. Many real industrial manufacturing problems can be modeled as the HFSP, like those in the fields of textile, paper, electronics and steelmaking industries [2-3]. Meanwhile, the HFSP with multiple stages and jobs is strongly NP-hard [4]. Therefore, it is of strong academic significance and engineering application value to study the HFSP in terms of theory and solution methodology.

Since the HFSP was proposed in 1954 [5], many approaches have been proposed. Roughly, they can be divided into three types: the exact methods, the heuristics and the meta-heuristics. The exact algorithms, such as branch and bound [6], can obtain the optimal solutions of a given problem, but they can only be used to solve the small-scaled problems due to the complexity of the problems. The heuristics, such as dispatching rules [7] and divide-and-conquer strategy [8], are relatively efficient. Nonetheless, the quality of the obtained solutions is often not satisfactory, especially for the large-scaled problems. With the development of computational intelligence, some meta-heuristics have been presented for solving the scheduling problem. These algorithms make it possible to achieve the satisfactory schedules for the large-scaled problems with reasonable computational effort. Among meta-heuristics, genetic algorithm (GA) [9-10], particle swarm optimization [11], ant colony optimization [12], tabu search [13], differential evolution [14], and shuffled frog leaping algorithm [15] have already been applied to solve the HFSP.

In the literature, it is often assumed that the processing times are deterministic values. However, in practice, several sources of uncertainty have effect on the production, thus the processing times of the operations are not deterministic [16]. Consequently, it is more significant to study the HFSP with stochastic processing time, namely the stochastic HFSP (SHFSP). The SHFSP is much closer to the real applications. However, to the best of our knowledge, the only presented algorithm for solving the SHFSP is a GA developed recently [17]. In [17], it also defined a robust bi-objective evaluation function to obtain a robust and effective solution that is only slightly sensitive to data uncertainty. So, the study on the SHFSP is still in its infancy, and it is important to develop novel and effective algorithms for the problem.

As a kind of particular evolutionary algorithm based on statistical learning, estimation of distribution algorithm [18] has gained increasing research and applications in several fields during recent years, such as feature selection, cancer classification, multidimensional knapsack problem, quadratic assignment problem, machinery structure design, flexible job-shop scheduling, nurse rostering, resource-constrained project scheduling, and so on [19-24]. As for the HFSP, in our previous work, an enhanced EDA with a hybrid decoding method was presented [25] and a compact EDA which employed only two individuals was introduced [26]. For the good performance of the EDA-based algorithms to the problems above, it is expected to solve the SHFSP efficiently by a well-designed EDA and to provide a new approach for solving the problem. Therefore, in this paper, we will propose an effective EDA to solve the SHFSP. To be specific, a probability model is presented to describe the probability distribution of the solution space and a

[*]This work is supported by the National Key Basic Research and Development Program of China (2013CB329503), the National Science Foundation of China (61174189 and 61025018), the Doctoral Program Foundation of Institutions of Higher Education of China (20100002110014), and the National Science and Technology Major Project of China (No.2011ZX02504-008).

mechanism is provided to update the probability model with the superior individuals. By sampling the probability model, new individuals can be generated among the search region with promising solutions. Numerical tests based on some well known benchmark instances are carried out. The comparisons with the existing GA demonstrate the effectiveness and robustness of the proposed algorithm.

The remainder of the paper is organized as follows. The SHFSP is described in Section 2. In Section 3 the basic EDA is introduced, and the framework of the EDA for solving the SHFSP is proposed in Section 4. Simulation results and comparisons are provided in Section 5. Finally, we end the paper with some conclusions in Section 6.

2 Problem description

2.1 Nomenclature

n: the number of jobs to be processed;

s: the number of processing stages;

m_k: the number of the machines at stage k;

X_{ijk}: a binary variable which is equal to 1 if job j is assigned to machine i at stage k and is equal to 0 otherwise;

Y_{ijk}: a binary variable which is equal to 1 if job i precedes job j at stage k and is equal to 0 otherwise;

R_i: the releasing time of job i;

S_{ik}: the starting time of job i at stage k;

P_{ik}: the processing time of job i at stage k;

C_{ik}: the completing time of job i;

L: a very large constant;

N: the number of stochastic scenarios;

$\xi_i(I)$: the i^{th} set of the sampled parameters from the initial scenario I;

P_I: the processing time of the initial scenario;

$\alpha \in [0,1]$: the uncertainty degree of the processing times.

2.2 The HFSP

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

Typically, the HFSP is supposed that [25-26]: all the n jobs are independent and available to be processed at the initial time; the releasing time of all machines is not considered or set as 0; one machine can process only one operation and one job can be processed at only one machine at a time; once an operation is started, it cannot be interrupted; for all the n jobs, the processing times on each machine are known in advance; buffers between stages are infinite; the time between different machines for transportation is negligible. In Fig. 1, it illustrates an example of HFSP.

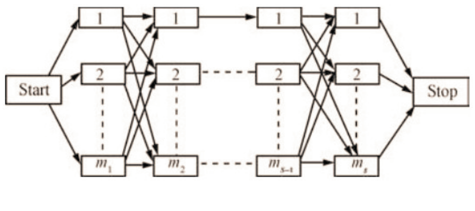


Fig. 1 An example of HFSP

The HFSP can be formulated as follows [27]:

$$\text{Minimize: } C_{\max} \quad (1)$$

Subject to:

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

$$C_{ik} = S_{ik} + P_{ik} \quad (3)$$

$$\sum_{i=1}^{m_k} X_{jik} = 1 \quad (4)$$

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

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

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

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

As for the above formulation: formula (1) implies that the objective function is to minimize the makespan of the HFSP, i.e., the maximum completion time of all the jobs as shown in formula (2); formula (3) describes the computation of C_{ik}; formula (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.

2.3 The Stochastic HFSP

In the SHFSP, the processing times are described as random variables. To evaluate a solution of the stochastic problem, a scenario modeling approach is often adopted to represent the characteristic of the uncertainty. It constructs a set of stochastic scenarios in which the processing times are sampled according to a certain probability distribution. Same as the literature [17], the uniform distribution is used in this paper.

The initial scenario I is a deterministic scenario, which represents the characteristics of the problem. The set of the stochastic scenarios is represented by sampling the initial scenario I. To be specific, $\xi_i(I)$ is supposed to be uniformly distributed between $[P_I - \alpha P_I, P_I + \alpha P_I]$. For a given solution, the average of the makespan calculated from all the scenarios is regarded as its makespan.

3 Estimation of Distribution Algorithm

EDA is a new paradigm in the field of evolutionary computation, which employs explicit probability distributions in optimization [18]. Compared with GA, EDA reproduces new population implicitly instead of the crossover and mutation. In 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 procedure of the basic EDA can be described as follows.

Step 1: Initialize the population;

Step 2: Select the superior sub-population;

Step 3: Estimate the probability distribution of the superior sub-population;

Step 4: Sample the probability model to generate the new population;

Step 5: If the stopping condition is met, the algorithm ends and outputs the best solution that is obtained in the searching procedure; else, go to step 2.

The critical step of EDA is to estimate the probability distribution. EDA uses 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.

4 EDA for the SHFSP

4.1 Evaluation Function

Considering the effectiveness and robustness of the schedule, to evaluate a solution x, an evaluation function [17] aggregating the two objectives is employed, which is expressed as follows:

$$f(x) = \lambda \frac{C_{\max}^{I}(x) - \text{LB}}{\text{LB}} + (1-\lambda) \times \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}[C_{\max}^{\xi_i(I)}(x) - C_{\max}^{I}(x)]^2}}{DEV\_MAX(x^*)} \quad (9)$$

where $\lambda \in (0,1)$ represents the weight coefficient; LB is the lower bound of the initial scenario, $DEV\_MAX(x^*) = \max\{(C_{\max}^{I}(x^*) - C_{\max}^{I_{\min}}(x^*)),(C_{\max}^{I_{\max}}(x^*) - C_{\max}^{I}(x^*))\}$; x^* is the best solution; $C_{\max}^{I_{\min}}(x^*)$ is the makespan of the minimal processing time calculated from the initial scenario, i.e., $I_{\min}=I-\alpha I$; $C_{\max}^{I_{\max}}(x^*)$ is the makespan of the maximal processing time calculated from the initial scenario, i.e., $I_{\max}=I+\alpha I$. The two values $C_{\max}^{I_{\min}}(x^*)$ and $C_{\max}^{I_{\max}}(x^*)$ are obtained by integrating the solution x^* obtained for the initial scenario I in the minimal and maximal scenarios.

It can be seen that, the schedule objective is to minimize the makespan of the initial scenario as well as the deviation of the makespan between all stochastic scenarios and the initial one. Note that LB and $DEV\_MAX(x^*)$ are used to normalize the two objectives since they do not have the same measurement scale.

4.2 Encoding and Decoding

Each individual of the population is a solution of the SHFSP, which is expressed by an integer number sequence with the length of n and determines the processing order of the first stage. For example, a solution x={5,2,3,1,4} represents that job 5 is processed first at the first stage, and next are job 2, job 3, and job 1 in sequence. Job 4 is the last job to be processed.

To decode a sequence is to assign the jobs to the machines at each stage so as to form a feasible schedule and calculate the makespan. For the SHFSP, the decoding procedure is 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. As for the machines assignment, the first available machine rule [28] is employed. That is, the job is assigned to the machine with the earliest release time.

For a given solution x, the makespan values of the initial scenario and stochastic scenarios are obtained with the decoding method and the target value $f(x)$ is calculated as introduced in section 4.1.

4.3 Probability Model and Updating Mechanism

The EDA produces new population 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 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 and all the elements of P will be normalized to maintain that each row sums up to 1. An individual is constructed until all the jobs appear in the sequence. In such a way, *Psize* individuals are generated.

Next, it determines the superior sub-population with the best *SPsize* solution. And then, the probability matrix P is updated according to the following equation:

$$p_{ij}(l+1) = (1-\beta)p_{ij}(l) + \frac{\beta}{i \times SPsize}\sum_{k=1}^{SPsize} I_{ij}^{k}, \forall i,j \quad (10)$$

where $\beta \in (0,1)$ 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.

$$I_{ij}^{k} = \begin{cases} 1, \text{if job } j \text{ appears before or in position } i \\ 0, \text{else} \end{cases} \quad (11)$$

The above updating process can be regarded as a kind of increased learning, where the second term on the right hand side of formula (10) represents learning information from the superior sub-population.

4.4 Procedure of the EDA for SHFSP

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

In the above EDA, it stops when the maximum number of generations *Gen* is satisfied.

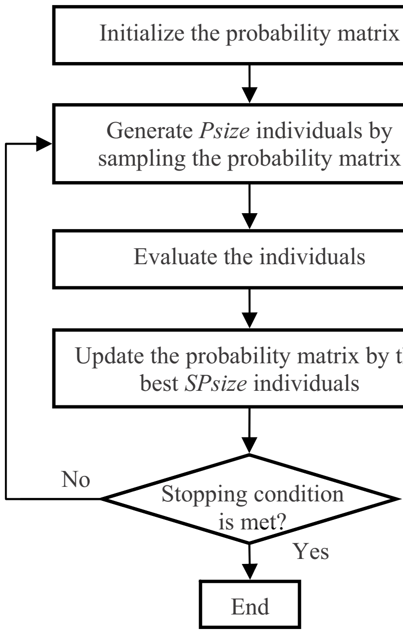
4.5 Computation Complexity Analysis

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

For the sampling process, every position is generated with the roulette strategy by sampling the probability matrix P. A sequence is constructed with the complexity $O(n^2)$ and *Psize* individuals can be generated with the complexity $O(n^2 Psize)$.

For the updating process, first it is with the computational complexity $O(Psize\log Psize)$ by using the quick sorting method to select the best *SPsize* individuals from population; then, it is with the complexity $O[n(SPsize+n)]$ to update all the $n \times n$ elements of P. Thus, the computational complexity for updating process is $O[n(SPsize+n)+ Psize\log Psize]$.

It can be seen that the complexity of the proposed EDA is not large and the algorithm could be efficient for solving the considered problem.



Initialize the probability matrix
Generate *Psize* individuals by sampling the probability matrix
Evaluate the individuals
Update the probability matrix by the best *SPsize* individuals
No
Stopping condition is met?
Yes
End

Fig. 2 The procedure of EDA for SHFSP

5 Simulation and Comparison

To test the performance of the EDA, numerical tests are carried out based on some benchmarks used in [17]. Table 1 presents the configurations and the LB values of these benchmarks. The algorithm is coded in C and run on Thinkpad T420 with a 2.3GHz processor and 2GB RAM.

Table 1: Configurations and LB values

No.	Benchmark	*n*	*s*	LB value
1	j10c5a2	10	5	88
2	j10c5b1	10	5	130
3	j10c5c1	10	5	68
4	j10c5d1	10	5	66
5	j15c5a1	15	5	178
6	j15c5b1	15	5	170
7	j15c5c1	15	5	85
8	j15c5d1	15	5	167
9	2center20job	20	2	161
10	2center50job	50	2	256
11	2center100job	100	2	402

To compare the proposed EDA to the GA [17], the best solution obtained by the algorithm is evaluated using extra 100 times to confirm the accuracy as the literature [17]. The parameters of the EDA are set as follows: *Psize*=50, *SPsize*=10, *N*=20, *β*=0.1, and *Gen*=100. Furthermore, the evaluation time of the EDA is 10^5 and it is no less than 2×10^6 of the GA [17].

Tables 2-4 provide the detailed results for the benchmarks with the uncertainty degree equals to 10%, 25% and 50%, respectively. The column "$C^I_{\max}$" presents the best makespan value of the initial scenario. The column "*STD*" presents the deviation of the makespan between all stochastic scenarios and the initial one, which is calculated as follows:

$$STD=\sqrt{\frac{1}{100}\sum_{i=1}^{100}(C_{\max}^{\xi_i(I)}-C_{\max}^{I})^2} \qquad (12)$$

The column "*AVG*" presents the average makespan value of 100 stochastic scenarios. The column "*DEV*" (in %) presents the deviation between *AVG* and $C^I_{\max}$, which is calculated as follows:

$$DEV=\frac{AVG-C_{\max}^{I}}{C_{\max}^{I}}\times100 \qquad (13)$$

Furthermore, Tables 5-6 provide the summarized results of $C^I_{\max}$ and *STD* for all the benchmarks with different degrees of uncertainty.

Table 2: Experimental results for *α*=10%

No.	*λ*	GA				EDA			
		$C^I_{\max}$	*STD*	*AVG*	*DEV*	$C^I_{\max}$	*STD*	*AVG*	*DEV*
1	1	**88**	8.83	96.04	9.13	**88**	**4.94**	**89.08**	1.23
	0.5	93	4.24	96.29	3.53	**89**	**2.28**	**87.96**	-1.17
	0	124	3.73	124.83	0.66	**105**	**2.25**	**104.14**	-0.82
2	1	**130**	7.06	136.66	5.12	**130**	**3.05**	**128.00**	-1.53
	0.5	134	4.34	137.42	2.55	**131**	**2.75**	**129.50**	-1.34
	0	146	4.01	148.10	1.43	**135**	**2.55**	**134.07**	-0.69
3	1	**68**	7.80	75.37	10.83	**68**	**2.14**	**67.03**	-1.43
	0.5	76	3.60	78.64	3.47	**72**	**2.09**	**71.40**	-0.83
	0	89	2.80	89.39	0.43	**82**	**1.89**	**81.74**	-0.31
4	1	**66**	7.92	73.53	11.40	**66**	**2.07**	**65.17**	-1.26
	0.5	72	2.70	73.01	1.40	**69**	**1.95**	**69.59**	0.85
	0	79	1.91	79.74	0.93	**74**	**1.72**	**73.50**	-0.68
5	1	**178**	9.30	186.48	4.76	**178**	**5.08**	**180.94**	1.65
	0.5	185	5.13	188.60	1.94	**183**	**3.33**	**181.36**	-0.90
	0	205	5.07	205.20	0.09	**189**	**3.16**	**187.68**	-0.70
6	1	**170**	9.86	179.31	5.47	**170**	**3.23**	**168.00**	-1.18
	0.5	175	6.82	181.05	3.45	**171**	**3.15**	**169.21**	-1.05
	0	188	4.79	190.47	1.31	**176**	**2.58**	**174.84**	-0.66
7	1	**85**	10.82	95.47	12.31	**85**	**3.19**	**85.95**	1.12
	0.5	94	3.85	95.92	2.04	**92**	**2.24**	**90.81**	-1.29
	0	114	2.83	113.61	-0.34	**100**	**2.14**	**100.47**	0.47
8	1	**167**	8.63	175.08	4.83	**167**	**3.00**	**165.51**	-0.89
	0.5	174	6.04	179.10	2.93	**170**	**2.72**	**168.87**	-0.66
	0	183	4.71	184.22	0.66	**175**	**2.68**	**173.88**	-0.64
9	1	**161**	8.97	169.23	5.11	**161**	**3.02**	**159.87**	-0.70
	0.5	165	4.04	168.30	2.00	**163**	**3.44**	**161.34**	-1.02
	0	177	3.93	175.23	-1.00	**168**	**3.02**	**166.77**	-0.73
10	1	**256**	10.98	266.75	4.19	**256**	**2.55**	**257.25**	0.49
	0.5	269	4.66	272.97	1.47	**263**	**2.50**	**262.58**	-0.16
	0	298	4.27	298.29	0.09	**273**	**2.40**	**273.22**	0.08
11	1	**402**	15.15	416.92	3.71	**402**	**2.84**	**403.35**	0.34
	0.5	417	5.86	421.82	1.15	**412**	**2.74**	**411.72**	-0.07
	0	433	4.96	435.74	0.63	**417**	**2.70**	**416.32**	-0.16

From Tables 2-6, it can be seen that the EDA performs better than the GA in solving the SHFSP. The $C^I_{\max}$ values and the *AVG* values by the EDA are smaller than or equal to that by the GA for all the instances, which means that the EDA is more effective when solving the initial scenario and the stochastic scenarios. In addition, the *STD* values by the EDA are also smaller than that by the GA for all the instances, which means that the EDA is more robust to the data uncertainty. Therefore, it can be concluded that the EDA is more effective and more robust than the GA for solving the SHFSP.

According to Tables 5-6, when the value of λ decreases, the values of $C^I_{\max}$ increase and the values of *STD* decrease. It is because the algorithm only minimizes the $C^I_{\max}$ when λ=1 while it only minimizes the *STD* when λ=0. For λ=0.5, it makes a trade off between the two objectives.

Table 3: Experimental results for α=25%

No.	λ	GA				EDA			
		$C^I_{\max}$	*STD*	*AVG*	*DEV*	$C^I_{\max}$	*STD*	*AVG*	*DEV*
1	1	**88**	9.63	96.19	9.30	**88**	**7.06**	**89.6**	1.82
	0.5	93	5.36	96.07	3.30	**90**	**5.07**	**89.27**	-0.81
	0	109	4.18	107.85	-1.05	**91**	**4.09**	**90.82**	-0.20
2	1	**130**	7.10	135.40	4.15	**130**	**5.68**	**128.46**	-1.18
	0.5	134	5.51	136.21	1.64	**130**	**4.93**	**129.18**	-0.63
	0	139	5.43	140.89	1.35	**135**	**5.13**	**135.19**	0.14
3	1	**68**	9.88	77.14	13.44	**68**	**3.80**	**70.31**	3.40
	0.5	75	4.25	77.56	3.41	**72**	**3.66**	**71.68**	-0.44
	0	84	3.87	83.60	-0.47	**78**	**3.45**	**78.33**	0.42
4	1	**66**	8.07	73.39	11.19	**66**	**3.80**	**68.56**	3.88
	0.5	72	3.42	72.21	0.29	**69**	**3.07**	**69.73**	1.06
	0	78	2.83	76.81	-1.52	**73**	**2.59**	**73.51**	0.70
5	1	**178**	12.78	188.32	5.79	**178**	**8.79**	**181.87**	2.17
	0.5	183	9.28	188.88	3.21	**181**	**6.49**	**179.87**	-0.62
	0	208	8.42	209.74	0.80	**185**	**6.34**	**183.18**	-0.98
6	1	**170**	9.36	177.41	4.35	**170**	**6.05**	**168.49**	-0.89
	0.5	171	7.68	176.05	2.95	**170**	**5.84**	**168.8**	-0.71
	0	182	6.96	185.06	1.68	**174**	**5.67**	**173.31**	-0.40
7	1	**85**	11.95	96.33	13.32	**85**	**4.33**	**87.93**	3.44
	0.5	93	3.68	95.62	2.81	**89**	**3.12**	**90.24**	1.39
	0	102	3.48	102.26	0.25	**95**	**3.03**	**94.46**	-0.57
8	1	**167**	9.31	174.25	4.34	**167**	**6.79**	**168.37**	0.82
	0.5	172	6.57	175.07	1.78	**167**	**5.76**	**166.55**	-0.27
	0	180	5.98	180.58	0.32	**174**	**5.76**	**173.91**	-0.05
9	1	**161**	11.00	170.02	5.60	**161**	**6.17**	**162.52**	0.94
	0.5	164	6.27	167.72	2.26	**163**	**5.53**	**163.89**	0.55
	0	178	6.18	178.76	0.42	**168**	**5.42**	**167.09**	-0.54
10	1	**256**	13.50	268.60	4.92	**256**	**6.82**	**259.65**	1.43
	0.5	265	6.89	269.14	1.56	**264**	**5.29**	**265.57**	0.59
	0	284	5.90	286.04	0.71	**274**	**5.21**	**276.00**	0.73
11	1	**402**	16.50	417.64	3.89	**402**	**7.70**	**405.89**	0.97
	0.5	415	8.64	421.40	1.54	**413**	**6.33**	**411.56**	-0.35
	0	441	7.68	442.02	0.23	**419**	**5.98**	**418.70**	-0.07

6 Conclusion

In this paper, an estimation of distribution algorithm was proposed for solving the hybrid flow-shop scheduling problem with stochastic processing time. In the proposed EDA, a bi-objective evaluation function was employed to evaluate the individuals of the population. A probability model was presented to describe the probability distribution of the solution space and mechanism was provided to update the probability model with the superior individuals. By sampling the probability model, new individuals were generated among the search region with the promising solutions. The effectiveness and robustness were demonstrated by numerical testing results and the comparison to the existing GA. The future work is to design effective EDA-based algorithms for other kinds of shop scheduling problems with the fuzzy or interval processing times.

Table 4: Experimental results for α=50%

No.	λ	GA				EDA			
		$C^I_{\max}$	*STD*	*AVG*	*DEV*	$C^I_{\max}$	*STD*	*AVG*	*DEV*
1	1	**88**	13.12	98.90	12.38	**88**	**9.51**	**89.66**	1.89
	0.5	93	8.34	97.52	4.86	**88**	**8.30**	**88.89**	1.01
	0	111	8.15	109.95	-0.94	**93**	**7.91**	**92.02**	-1.05
2	1	**130**	11.50	136.12	4.70	**130**	**10.56**	**133.14**	2.42
	0.5	**130**	10.03	135.17	3.97	**130**	**9.53**	**131.31**	1.01
	0	137	9.85	138.81	1.32	**135**	**9.36**	**134.18**	-0.60
3	1	**68**	13.12	79.76	17.29	**68**	**8.89**	**74.40**	9.41
	0.5	73	7.28	77.61	6.31	**72**	**6.54**	**74.81**	3.90
	0	79	6.14	80.66	2.10	**76**	**5.77**	**76.75**	0.97
4	1	**66**	10.06	74.37	12.68	**66**	**7.65**	**71.77**	8.74
	0.5	72	5.28	73.84	2.55	**70**	**5.06**	**71.52**	2.17
	0	79	5.02	78.31	-0.87	**73**	**4.94**	**73.80**	1.10
5	1	**178**	16.52	186.86	4.97	**178**	**14.70**	**181.74**	2.10
	0.5	**178**	14.76	184.8	3.82	**178**	**12.11**	**180.83**	1.59
	0	186	13.46	187.97	1.05	**186**	**12.01**	**184.04**	-1.05
6	1	**170**	13.18	177.51	4.41	**170**	**11.68**	**171.94**	1.14
	0.5	**170**	12.24	175.90	3.47	**170**	**11.01**	**170.73**	0.43
	0	182	11.71	183.71	0.93	**174**	**10.92**	**173.29**	-0.41
7	1	**85**	13.78	97.68	14.91	**85**	**9.35**	**92.02**	8.26
	0.5	91	7.29	96.35	5.87	**90**	**6.84**	**94.43**	4.92
	0	100	6.76	99.93	-0.07	**97**	**5.64**	**95.22**	-1.84
8	1	**167**	13.36	172.15	3.08	**167**	**13.16**	**169.26**	1.35
	0.5	**167**	11.81	169.83	1.69	**167**	**10.89**	**169.12**	1.27
	0	173	11.43	174.12	0.64	**170**	**10.36**	**168.57**	-0.84
9	1	**161**	12.59	168.98	4.95	**161**	**10.40**	**165.16**	2.58
	0.5	**162**	10.51	167.61	3.46	**162**	**10.35**	**164.39**	1.48
	0	169	9.37	169.52	0.30	**167**	**9.04**	**167.37**	0.22
10	1	**256**	17.44	270.99	5.85	**256**	**12.27**	**263.79**	3.04
	0.5	264	10.47	270.21	2.35	**263**	**10.43**	**265.64**	1.00
	0	273	10.07	277.49	1.64	**272**	**9.43**	**274.36**	0.87
11	1	**402**	19.92	418.71	4.15	**402**	**14.11**	**409.97**	1.98
	0.5	405	15.98	414.83	2.42	**405**	**13.17**	**412.11**	1.76
	0	432	11.98	434.99	0.69	**425**	**11.44**	**426.95**	0.46

Table 5: The average $C^I_{\max}$ for all the benchmarks

λ	α=10%		α=25%		α=50%	
	GA	EDA	GA	EDA	GA	EDA
1	**161**	**161**	**161**	**161**	**161**	**161**
0.5	168.55	**165**	167	**164.36**	164.09	**163.18**
0	185.09	**172.18**	180.45	**169.64**	174.64	**169.82**

Table 6: The average *STD* for all the benchmarks

λ	α=10%		α=25%		α=50%	
	GA	EDA	GA	EDA	GA	EDA
1	9.57	**3.19**	10.83	**6.09**	14.05	**11.11**
0.5	4.66	**2.65**	6.14	**5.01**	10.36	**9.48**
0	3.91	**2.46**	5.54	**4.79**	9.45	**8.8**

References

[1] L. Wang, *Shop Scheduling with Genetic Algorithms*. Beijing: Tsinghua University Press, 2003, 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*, 38(1): 1-8, 2011.
[3] Q.K. Pan, L. Wang, K. Mao, J.H. Zhao, and M. Zhang, An effective artificial bee colony algorithm for a real-world hybrid flowshop problem in steelmaking process, *IEEE Trans. on Automation Science and Engineering*, 10(2): 307-322, 2013.
[4] J.N.D. Gupta, Two-stage hybrid flow shop scheduling problem, *J of the Operational Research Society*, 39(4): 359-364, 1988.
[5] S.M. Johnson, Optimal two and three-stage production schedules with setup times included, *Naval Research Logistics Quarterly*, 1(1): 61-68, 1954.
[6] S.A. Brah and J.L. Hunsucker, Branch and bound algorithm for the flow-shop with multiple processors, *European J of Operational Research*, 51(1): 88-99, 1991.
[7] S.N. Kadipasaoglu, W. Xiang, and B.M. Khumawala, A comparison of sequencing rules in static and dynamic hybrid flow systems, *Int. J of Production Research*, 35(5): 1359-1384, 1997.
[8] G. Vairaktarakis, and M. Elhafsi, The use of flowlines to simplify routing complexity in two-stage flowshops, *IIE Transactions*, 32(8): 687-699, 2000.
[9] H.R. Zhou, W.S. Tang, and Y.H. Wei, Optimize flexible flow-shop scheduling using genetic algorithm, *Computer Engineering and Applications*, 45(30): 224-226, 2009.
[10] J.S. Cui, T.K. Li, and W.X. Zhang, Hybrid flowshop scheduling model and its genetic algorithm, *J of University of Science and Technology Beijing*, 27(5): 623-626, 2005.
[11] C.T. Tseng and C.J. Liao, A particle swarm optimization algorithm for hybrid flowshop scheduling with multiprocessor tasks, *Int. J of Production Research*, 46(17): 4655-4670, 2008.
[12] K.C. Ying and S.W. Lin, Multiprocessor task scheduling in multistage hybrid flowshops: an ant colony system approach, *Int. J of Production Research*, 44(16): 3161-3177, 2006.
[13] R. Logendran, P. deSzoeke, and F. Barnard, Sequence-dependent group scheduling problems in flexible flow shops, *Int. J of Production Economics*, 102(1): 66-86, 2006.
[14] Y. Xu, and L. Wang, Differential evolution algorithm for hybrid flow-shop scheduling problems, *J of Systems Engineering and Electronics*, 22(5): 794-798, 2011.
[15] 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*, 6838: 560-567, 2011.
[16] S.Y. Wang, L. Wang, Y. Xu, and M. Liu, An effective estimation of distribution algorithm for the flexible job-shop scheduling problem with fuzzy processing time, *Int. J of Production Research*, doi: 10.1080/00207543.2013.765077, 2013.
[17] T. Chaari, S. Chaabane, T. Loukil and D. Trentesaux, A genetic algorithm for robust hybrid flow shop scheduling, *Int. J of Computer Integrated Manufacturing*, 24(9): 821-833, 2011.
[18] P. Larranaga and J.A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*, Boston: Kluwer, 2002.
[19] S.Y. Wang, L. Wang, C. Fang, and Y. Xu, Advanced in estimation of distribution algorithms, *Control and Dicision*, 27(7): 961-966, 2012.
[20] L. Wang, S.Y. Wang, and Y. Xu, An effective hybrid EDA-based algorithm for solving multidimensional knapsack problem, *Expert Systems with Applications*, 39(5): 5593-5599, 2012.
[21] L. Wang, S.Y. Wang, and M. Liu, A Pareto-based estimation of distribution algorithm for the multi-objective flexible job-shop scheduling problem, *Int. J of Production Research*, doi: 10.1080/00207543.2012.752588, 2013.
[22] L. Wang, S.Y. Wang, Y. Xu, G. Zhou, and M. Liu, A bi-population based estimation of distribution algorithm for the flexible job-shop scheduling problem, *Computers & Industrial Engineering*, 62(4): 917-926, 2012.
[23] L. Wang, and C. Fang, An effective estimation of distribution algorithm for the multi-mode resource-constrained project scheduling problem, *Computers & Operations Research*, 39(2): 449-460, 2012.
[24] L. Wang, C. Fang, C.D. Mu, and M. Liu, A Pareto-archived estimation-of-distribution algorithm for multiobjective resource-constrained project scheduling problem, *IEEE Trans. on Engineering Management*, doi: 10.1109/TEM.2012.2234463, 2013.
[25] S.Y. Wang, L. Wang, M. Liu, and Y. Xu, An enhanced estimation of distribution algorithm for solving hybrid flow-shop scheduling problem with identical parallel machines, The *Int. J of Advanced Manufacturing Technology*, doi: 10.1007/s00170-013-4819-y, 2013.
[26] S.Y. Wang, L. Wang, and Y. Xu, A compact estimation of distribution algorithm for solving hybrid flow-shop scheduling problem, in: *The 10th World Congress on Intelligent Control and Automation*, Beijing, pp.649-653, 2012.
[27] 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*, 164(1): 29-40, 2008.
[28] S.A. Brah, and L.L. Luan, Heuristics for scheduling in a flowshop with multiple processors, *European J of Operational Research*, 113(1): 113-122, 1999.