# An Estimation of Distribution Algorithm for the Flexible Job-Shop Scheduling Problem

Shengyao Wang, Ling Wang, Gang Zhou, and Ye Xu

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,
{g-zhou09,xuye05}@mails.tsinghua.edu.cn

**Abstract.** In this paper, an effective estimation of distribution algorithm (EDA) is proposed to solve the flexible job-shop scheduling problem with the criterion to minimize the maximum completion time (makespan). With the framework of the EDA, the probability model is built with the superior population and the new individuals are generated based on probability model. In addition, an updating mechanism of the probability model is proposed and a local search strategy based on critical path is designed to enhance the exploitation ability. Finally, numerical simulation is carried out based on the benchmark instances, and the comparisons with some existing algorithms demonstrate the effectiveness of the proposed algorithm.

**Keywords:** flexible job-shop scheduling problem, makespan, estimation of distribution algorithm, probability model, critical path.

## 1 Introduction

The flexible job-shop scheduling problem (FJSP) is a generalization of the classical job shop scheduling problem (JSP) for flexible manufacturing systems. Each machine may have the ability of performing more than one type of operations. The FJSP consists of two sub-problems: the routing sub-problem that assigns each operation to a machine among a set of capable machines, and the scheduling sub-problem that sequences the assigned operations on all machines to obtain a feasible schedule to minimize the objective function. Therefore, the FJSP is more difficult than classical JSP because it should determine the assignment of operations to machines as well as the sequence of all operators. It has been proved that the FJSP is NP-hard. Hence, the study of the FJSP in theory, methodology and applications has significant importance in both academic and application fields.

Due to the complexity of the FJSP, meta-heuristics and evolutionary algorithms have been widely used. In [1], a tabu search (TS) algorithm based on the integrated approach was proposed and then improved in terms of computation time and solution quality with two neighborhood functions developed [2]. In [3], a genetic algorithm (GA) hybridized with variable neighborhood search was proposed [3], and a GA integrating different strategies was proposed in [4]. Recently, a knowledge-based ant

D.-S. Huang et al. (Eds.): ICIC 2011, LNAI 6839, pp. 9–16, 2012.

colony optimization algorithm (KBACO) was proposed in [5] and a hybrid TS with an efficient neighborhood structure was presented in [6].

As a new population-based algorithm, estimation of distribution algorithm (EDA) [7] has gained an increasing study and applications during recent years. Population-based incremental learning (PBIL) is the earliest model of the EDA. 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 [7] 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 [8]. However, to the best of our knowledge, there is no research work about the EDA for solving the FJSP. In this paper, we will propose an effective EDA for solving the FJSP with a criterion of makespan. A probability model is built and an updating mechanism is proposed. In addition, a local search strategy based on critical path is developed to improve the convergence speed. Finally, we use a set of benchmark instances to test the performances of the EDA and to compare it with some existing methods to further demonstrate the effectiveness of the EDA.

The remainder of the paper is organized as follows. In Section 2, the FJSP is described in brief. In Section 3 the basic EDA is introduced, and the framework of EDA for the FJSP 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

The flexible job-shop scheduling problem (FJSP) is commonly defined as follows:

There are a set of $n$ jobs $J=\{J_1,J_2,\ldots,J_n\}$ to be processed on $m$ machines $M=\{M_1,M_2,\ldots,M_m\}$. A job $J_i$ is formed by a sequence of $n_i$ operations $\{O_{i,1},O_{i,2},\ldots,O_{i,n_i}\}$ to be performed one after another according to the given sequence. The execution of $O_{i,j}$ requires one machine out of a set of $m_{ij}$ given machines $M_{i,j}\subseteq M$. Preemption is not allowed, i.e., each operation must be completed without interruption once it starts. All jobs and machines are available at time 0. Setup time of machines and move time between operations are negligible. The processing time of $O_{i,j}$ performed on machine $M_k$ is $\mathrm{r}_{i,j,k}>0$. The FJSP is to determine both the assignment of machines and the sequence of operations on all the machines to minimize a certain objective function, e.g. makespan ($C_{\max}$).

# 3 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 [4]. 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 the updating mechanism should be well developed to estimate the underlying probability distribution. Nevertheless, the EDA pays more attention to global exploration while its exploitation capability is limited. So, an effective EDA should balance the exploration and the exploitation abilities.

# 4 EDA for FJSP

## 4.1 Solution Representation

Every individual of the population is a solution of the FJSP. The solution of the FJSP is a combination of operation scheduling decisions and machine assignment. So, a solution can be expressed by the processing sequence of operations and the assignment of operations on the machines. It consists of two vectors corresponding to the two sub-problems of the FJSP, i.e., operation sequence vector and machine assignment vector.

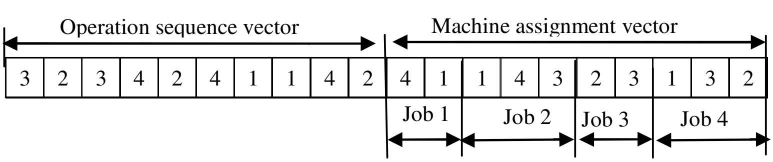


**Fig. 1.** Illustration of the representation of a feasible solution

For the operation sequence vector, the number of genes equals to the total number of operations $T_o$. Let job number denote the operations of each job. The $k^{th}$ occurrence of a job number refers to the $k^{th}$ operation in the sequence of this job. For the machine assignment vector, each number represents the corresponding selected machine for each operation. So the number of genes is also $T_o$. For example, a feasible solution for a problem with 4 jobs and 4 machines is shown in Fig. 1. The operation sequence and machine assignment can be interpreted as follows: ($O_{3,1}$, $M_2$), ($O_{2,1}$, $M_1$), ($O_{3,2}$, $M_3$), ($O_{4,1}$, $M_1$), ($O_{2,2}$, $M_4$), ($O_{4,2}$, $M_3$), ($O_{1,1}$, $M_4$), ($O_{1,2}$, $M_1$), ($O_{4,3}$, $M_2$), ($O_{2,3}$, $M_3$). The Gantt chart of this solution is shown in Fig. 2.

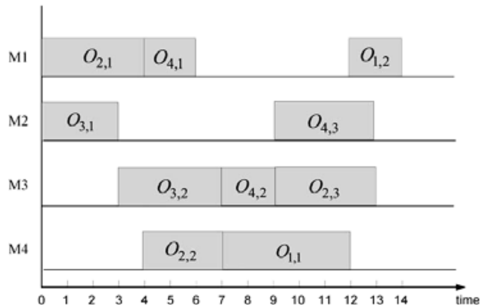


**Fig. 2.** Gantt chart of the solution shown in Fig. 1

## 4.2 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 two probability matrixes.

The element $p_{ij}(t)$ of operation probability matrix $A_1$ represents the probability that job $J_j$ appears before or in position $i$ of the sequence vector at generation $t$. The value of $p_{ij}$ refers to the importance of a job when scheduling the operations on machines. For all $i$ ($i = 1,2,\cdots,T_o$) and $j$ ($j = 1,2,\cdots,n$), $p_{ij}$ is initialized to $p_{ij}(0) = 1/n$, which ensures that the whole solution space can be sampled uniformly.

The element $q_{ijk}(t)$ of machine probability matrix $A_2$ represents the probability that operation $O_{i,j}$ is processed on machine $M_k$ at generation $t$. The value of $q_{ijk}$ indicates the appropriateness of an operation processed on a certain machine. The probability matrix $A_2$ is initialized as follows:

$$q_{ijk} = \begin{cases} 1/m_{ij}, \text{if } O_{i,j} \text{ can be processed on machine } M_k \\ 0, \text{else} \end{cases} \tag{1}$$

In each generation of the EDA, the new individuals are generated via sampling according to the probability matrix $A_1$ and $A_2$. To generate a new solution, the operation sequence vector should be generated first. For every position $i$, job $J_j$ is selected with the probability $p_{ij}$. If job $J_j$ has already appeared $n_j$ times in the operation sequence vector, it means the processing procedure of job $J_j$ is completed. So, the whole column $p_{1j}, p_{2j}, \cdots, p_{T_o j}$ of probability matrix $A_1$ will be set as zero.

Similarly, the machine assignment vector is generated according to probability matrix $A_2$. In such a way, $P$ individuals are constructed and their makespan are calculated.

Then, the superior population that consists of the best $SP$ solutions is determined, and the matrix $A_1$ and $A_2$ are updated according to the following equations:

$$p_{ij}(t+1) = (1-\alpha)p_{ij}(t) + \frac{\alpha}{i \times SP}\sum_{s=1}^{SP} I_{ij}^{s}, (i = 1,2,\cdots,T_o; j = 1,2,\cdots,n) \tag{2}$$

$$q_{ijk}(t+1) = (1-\alpha)q_{ijk}(t) + \frac{\alpha}{SP}\sum_{s=1}^{SP} \tilde{I}_{ijk}^{s}, (i = 1,2,\cdots,n; j = 1,2,\cdots,n_i; k = 1,2,\cdots,m) \tag{3}$$

where $\alpha \in (0,1)$ is the learning speed, $I_{ij}^{s}$ and $\tilde{I}_{ij}^{s}$ are the indicator function of the $s^{th}$ individual in the superior population that are defined as follows:

$$I_{ij} = \begin{cases} 1, \text{if job } J_j \text{ appears before or in position } i \\ 0, \text{else} \end{cases} \tag{4}$$

$$\tilde{I}_{ijk} = \begin{cases} 1, \text{if operation } O_{i,j} \text{ is processed on machine } M_k \\ 0, \text{else} \end{cases} \tag{5}$$

### 4.3 Local Search Based on Critical Path

It is widely accepted that a local search procedure is efficient in improving the solutions generated by the EDA. In this paper, a local search based on critical path is designed to enhance the local exploitation around the best solution in each generation.

Denote $S_{i,j}^{E}$ as the earliest starting time of operation $O_{i,j}$ and $S_{i,j}^{L}$ as the latest starting time without delaying the makespan. Thus, the earliest completion time of operation $O_{i,j}$ is $C_{i,j}^{E} = S_{i,j}^{E} + t_{i,j,k}$, and the latest completion time is $C_{i,j}^{L} = S_{i,j}^{L} + t_{i,j,k}$. Let $PM_{i,j}^{k}$ be the operation processed on machine $M_k$ right before the operation $O_{i,j}$ and $SM_{i,j}^{k}$ be the operation processed on machine $M_k$ right after $O_{i,j}$. Let $PJ_{i,j} = O_{i,j-1}$ be the operation of job $J_i$ that precedes $O_{i,j}$ and $SJ_{i,j} = O_{i,j+1}$ be the operation of job $J_i$ that follows $O_{i,j}$.

Since the makespan is no shorter than any possible critical path, the makespan may be improved only by moving the critical operations. Let $O_l$ ($l = 1,2,...,N_c$) be the critical operation to be moved, where $N_c$ is the total number of critical operations of a solution. Moving $O_l$ is to delete it from its current position and then to insert it at another feasible position. Obviously, the makespan of the new solution is no larger than the old one. If $O_l$ is assigned before $O_{i,j}$ on machine $M_k$, it can be started as early as $C^E(PM_{i,j}^k)'$ and can be finished as late as $S_{i,j}^{L'}$ without delaying the required makespan. Besides, $O_l$ cannot violate the precedence relations of the same job. Thus, the assignable idle time interval for $O_l$ can be defined by $\max\{C^E(PM_{i,j}^k)', C^E(PJ_l)'\} + t_{l,k} <= \min\{S_{i,j}^{L'}, S^L(SJ_l)'\}$.

The above moving process is repeated until all critical operations are moved. Let $N_l$ be the number of positions to move $O_l$ feasibly, then the total number of moving neighbors of a solution is $N_{total} = \sum_{l=1}^{N_c} N_l$.

Moving critical operations will obtain new solutions. For the FJSP, there may be more than one critical path for a schedule. To improve a solution, all its critical paths should be changed. So, a solution with fewer critical paths is more likely to be improved. Thus, in our algorithm the new solution will replace the old one if one of the following conditions is satisfied: 1) The new solution has a smaller makespan than the old one; 2) The new solution with the same makespan as the old one has fewer critical paths. Moreover, to take advantage of the move of critical operations, we propose the following local search procedure based on moving operations for the best individual of the population in each generation.

Step 1: Generate $s'$ by moving all critical operations of solution $s$, then let $s' = s$.
Step 2: If the makespan of $s'$ is short than $s$, then go back to Step 1; else, stop.

### 4.4 Procedure of EDA for FJSP

With the design above, the procedure of the HEDA is illustrated in Fig. 3.

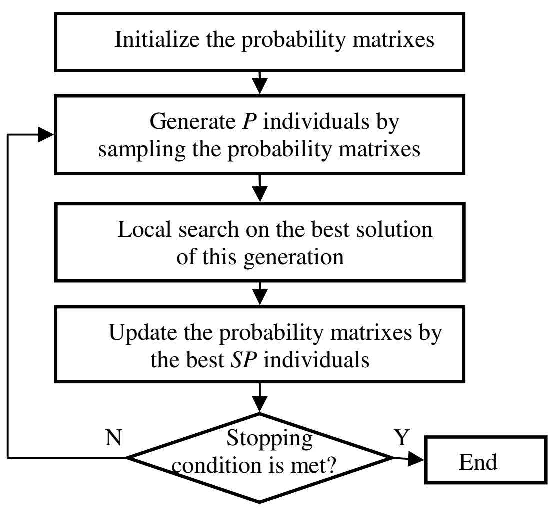


**Fig. 3.** The framework of the EDA

In summary, in the initial stage of evolution, the promising area of the solution space may be found by using the EDA based estimating and sampling. Then, local search strategy is performed in the "good" region to obtain better solutions. The benefits of the EDA and the local search are combined to balance global exploration and local exploitation. The algorithm stops when the maximum number of generations $Gen$ is satisfied.

## 5 Simulation and Comparison

To test the performance of the proposed EDA, numerical simulations are carried out with two well-studied benchmark sets including five Kacem instances (case 1~case 5) [9] and ten BRdata instances (MK1~MK10) [10]. The algorithm is coded in C++ and run on a 3.2GHz Intel Core i5 processor.

For each instance, the algorithm is run 50 times independently. We set $P = n \times m$, $SP = 10\% \times P$, $\alpha = 0.1$, $Gen = 2000$ and test the performance of the EDA and compare it with KBACO [5] and TSPCB [6]. The results are listed in Table 1, including the best solution (Best), average solution (AVG) and standard derivation (SD) as well as the average running time of the EDA.

**Table 1.** Comparisons of EDA with KBACO and TSPCB

| Instance | $n \times m$ | KBACO | | | TSPCB | | | EDA | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Best | AVG | SD | Best | AVG | SD | Best | AVG | SD | $T_{AVG}$(s) |
| Case 1 | 4×5 | **11** | **11.00** | 0 | **11** | **11.00** | N/A | **11** | **11.00** | 0 | <0.01 |
| Case 2 | 8×8 | **14** | 14.30 | 0.46 | **14** | 14.20 | N/A | **14** | **14.00** | 0 | <0.01 |
| Case 3 | 10×7 | **11** | **11.00** | 0 | **11** | **11.00** | N/A | **11** | **11.00** | 0 | 0.26 |
| Case 4 | 10×10 | **7** | 7.40 | 0.48 | **7** | 7.10 | N/A | **7** | **7.00** | 0 | 0.53 |
| Case 5 | 15×10 | **11** | 11.30 | 0.46 | **11** | 11.70 | N/A | **11** | **11.02** | 0.14 | 9.08 |
| Mk1 | 10×6 | **39** | **39.80** | 0.43 | 40 | 40.30 | N/A | 40 | 40.44 | 0.50 | 5.34 |
| Mk2 | 10×6 | 29 | 29.10 | 0.28 | **26** | **26.50** | N/A | **26** | 26.98 | 0.32 | 10.76 |
| Mk3 | 15×8 | **204** | **204.00** | 0 | **204** | **204.00** | N/A | **204** | **204.00** | 0 | 0.08 |
| Mk4 | 15×8 | 65 | 66.10 | 1.06 | 62 | 64.88 | N/A | **61** | **63.66** | 0.93 | 41.26 |
| Mk5 | 15×4 | 173 | 173.80 | 1.08 | **172** | **172.90** | N/A | **172** | 173.30 | 0.49 | 27.66 |
| Mk6 | 10×15 | 67 | 69.10 | 1.03 | 65 | 67.38 | N/A | **63** | **64.98** | 0.67 | 125.44 |
| Mk7 | 20×5 | 144 | 145.40 | 1.42 | **140** | 142.21 | N/A | **140** | **141.58** | 0.87 | 85.71 |
| Mk8 | 20×10 | **523** | **523.00** | 0 | **523** | **523.00** | N/A | **523** | **523.00** | 0 | 0.29 |
| Mk9 | 20×10 | 311 | 312.20 | 1.81 | 310 | 311.29 | N/A | **309** | **310.90** | 0.36 | 314.22 |
| Mk10 | 20×15 | 229 | 233.70 | 3.27 | **214** | **219.15** | N/A | 225 | 229.18 | 1.53 | 481.72 |

From Table 1, it can be seen that the EDA is better than KBACO and no worse than TSPCB in term of solution quality. As for the average performance, the EDA is the best one except only four instances. In addition, the standard derivation of EDA is smaller than KBACO for almost all the instance. Moreover, it can be seen that the average running time of EDA is acceptable, even for relatively larger-scale instances. So, the conclusion is that our EDA is effective and robust in solving the FJSP.

## 6 Conclusion

This was the first report work to apply EDA for solving the FJSP. We designed a probability model with the superior population for the EDA to solve the FJSP by generating new individuals via sampling based on the probability model. With an updating mechanism for the probability model and local search based on critical path, the EDA was effective and efficient in solving the FJSP, which was demonstrated by simulation results and comparisons. The future work is to design effective EDA for the multi-objective FJSP.

**Acknowledgments.** This work was supported by the National NSFC (70871065 and 60834004), Program for New Century Excellent Talents in University (NCET-10-0505) and Doctoral Program Foundation of Institutions of Higher Education of China (20100002110014).

## References

1. Dauzere-Peres, S., Paulli, J.: An Integrated Approach for Modeling and Solving the General Multiprocessor Job-shop Scheduling Problem Using Tabu Search. Annals of Operations Research 70(0), 281–306 (1997)
2. Mastrolilli, M., Gambardella, L.M.: Effective Neighborhood Functions for the Flexible Job Shop Problem. J. of Scheduling 3(1), 3–20 (2000)
3. Gao, J., Sun, L.Y., Gen, M.: A Hybrid Genetic and Variable Neighborhood Descent Algorithm for Flexible Job Shop Scheduling Problems. Computer & Operations Research 35(9), 2892–2907 (2008)
4. Pezzella, F., Morganti, G., Ciaschetti, G.: A Genetic Algorithm for the Flexible Job-shop Scheduling Problem. Computers & Operations Research 35(10), 3202–3212 (2008)
5. Xing, L.N., Chen, Y.W., Wang, P., Zhao, Q.S., Xiong, J.: A Knowledge-based Ant Colony Optimization for Flexible Job Shop Scheduling Problems. Applied Soft Computing 10(3), 888–896 (2010)
6. Li, J.Q., Pan, Q.K., Suganthan, P.N., Chua, T.J.: A Hybrid Tabu Search Algorithm with An Efficient Neighborhood Structure for the Flexible Job Shop Scheduling Problem. International J. of Advanced Manufacturing Technology 52(5-8), 683–697 (2011)
7. Larranaga, P., Lozano, J.A.: Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation. Springer, Netherlands (2002)
8. Zhou, S.D., Sun, Z.Q.: A Survey on Estimation of Distribution Algorithms. Acta Automatica Sinica 33, 113–124 (2007)
9. Kacem, I., Hammadi, S., Borne, P.: Pareto-optimality Approach for Flexible Job-shop Scheduling Problems: Hybridization of Evolutionary Algorithms and Fuzzy Logic. Mathematics and Computers in Simulation 60(3-5), 245–276 (2002)
10. Brandimarte, P.: Routing and Scheduling in A Flexible Job Shop by Tabu Search. Annals of Operations Research 41(3), 157–183 (1993)