Int. J. Production Economics 141 (2013) 24–33

Contents lists available at SciVerse ScienceDirect

Int. J. Production Economics

Addressing the advantages of using ensemble probabilistic models in Estimation of Distribution Algorithms for scheduling problems

Shih-Hsin Chen [a,*], Min-Chih Chen [b]

[a] Department of Electronic Commerce Management, Nanhua University, 62248, Taiwan, ROC
[b] Department of Information Management, WuFeng University, 62153, Taiwan, ROC

ARTICLE INFO

Article history:
Received 4 November 2011
Accepted 10 May 2012
Available online 22 May 2012

Keywords:
Estimation of Distribution Algorithms
Single machine scheduling problem
Permutation flowshop scheduling problem
Self-Guided Genetic Algorithm

ABSTRACT

Estimation of Distribution Algorithms (EDAs) have recently been recognized as a prominent alternative to traditional evolutionary algorithms due to their increasing popularity. The core of EDAs is a probabilistic model which directly impacts performance of the algorithm. Previous EDAs have used a univariate, bi-variate, or multi-variable probabilistic model each time. However, application of only one probabilistic model may not represent the parental distribution well. This paper advocates the importance of using ensemble probabilistic models in EDAs. We combine the univariate probabilistic model with the bi-variate probabilistic model which learns different population characteristics. To explain how to employ the two probabilistic models, we proposed the Ensemble Self-Guided Genetic Algorithm (eSGGA). The extensive computation results on two NP-hard scheduling problems indicate the advantages of adopting two probabilistic models. Most important of all, eSGGA can avoid the computation effort overhead when compared with other EDAs employing two models. As a result, this paper might point out a next generation approach for EDAs.

1. Introduction

In recent years, Estimation of Distribution Algorithms (EDAs) have attracted a lot of attention and have emerged as a prominent alternative to traditional evolutionary algorithms (Aickelin et al., 2007; Chang et al., 2008b; Chen et al., 2009; Zhang and Li, 2011; Hauschild and Pelikan, 2011; Pan and Ruiz, 2012). Compared with genetic algorithms (GAs) that employ the crossover and mutation operators to generate solutions, EDAs do not use the crossover or mutation. Instead, they explicitly extract global statistical information from the previous search and build a posterior probability model of promising solutions from which new solutions are sampled. It is the most important characteristic to distinguish EDAs from GAs (Zhang et al., 2005).

A number of the latest papers on EDAs in solving some NP-hard scheduling problems (Jarboui et al., 2009; Chen et al., 2009, 2011; Ceberio et al., 2012; Zhang and Li, 2011; Pan and Ruiz, 2012) have shown that EDAs are able to perform effectively. Ceberio et al. (2012), in particular, extensively tested 13 famous permutation-based approaches in EDAs on permutation flowshop scheduling problems (PFSPs) and other three combinatorial optimization problems. Their paper has provided a good basis for comparison for researchers.

However, most EDAs obtained statistical information from only one model, being it a univariate, bi-variate, or multi-variate probabilistic model. It is much limited to select an appropriate model a priori. Some researches pointed out that the ensemble of different models can improve robustness of optimization at minimum cost (Lim et al., 2010; Goel et al., 2007; Samad et al., 2008). To the best of our knowledge, only Jarboui et al. (2009), and Pan and Ruiz (2012) simultaneously used two probabilistic models, i.e., the univariate and bi-variate probabilistic models, to generate statistical data, which have been the key distinguishing characteristic from the approaches employed in the past. The ensemble models could learn the positional and interaction information between the variables and sample new solutions from the models. This approach could generate more accurate parental distributions for EDAs. So, this research attempts to propose a new algorithm combining both the univariate and bi-variate probabilistic models and then to validate the performance of the two-model EDAs.

In this paper, we adopted a recently proposed EDA in Chen et al. (2012a) which is termed as the Self-Guided Genetic Algorithm (SGGA) to explain how to use two probabilistic models in the algorithm. In addition, this novel algorithm used the statistical information to guide the crossover and mutation operators without sampling new solutions after the probabilistic model was built. The probabilistic model actually estimated the quality of generated solutions by crossover and mutation operators first. Those with estimated higher quality will be allowed to enter the

* Corresponding author.
E-mail address: shihhsin@mail.nhu.edu.tw (S.-H. Chen).

0925-5273/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ijpe.2012.05.010

next generation for further evolution. By using this approach, the probabilistic model guided the evolutionary process and SGGA dealt with the PFSP in terms of the solution quality and computational efficiency successfully. In order to enhance the performance of SGGA, both univariate and bi-variate probabilistic models are employed in this ensemble model. The resultant algorithm is named as Ensemble Self-Guided Genetic Algorithm (eSGGA).

To evaluate the performance of the proposed algorithm and the effectiveness of using two probabilistic models together, eSGGA was compared with some EAs and famous permutation-oriented EDAs on two intractable scheduling problems in the literature. The first scheduling problem included single machine scheduling problem with an objective to minimize the total sum of earliness and tardiness penalties. As a generalization of weighted tardiness scheduling, the problem is strongly NP-hard (Lenstra et al., 1977). The studied single machine scheduling problem is regarded as NP-complete (Li, 1997). Secondly, permutation flowshop scheduling problem with minimization of the makespan is tested in this paper, which is one of the most extensively studied problems in the area of scheduling (Framinan et al., 2004; Hejazi and Saghafian, 2005; Ruiz and Maroto, 2005). Because most variants of the PFSP are NP-complete, research on the PFSP has been focused on developing effective heuristics (Reeves, 1995; Minella et al., 2008; Chang et al., 2010; Jarboui et al., 2009).

Contributions: Because the probabilistic model directly impacts the performance of EDAs (Lozano et al., 2006), this research advocates the importance of combining both univariate and bivariate statistical information. The proposed algorithm eSGGA along with two other researches based on the similar idea (Jarboui et al., 2009; Pan and Ruiz, 2012) are used to compare its performance with other EDAs considering statistical information from only one model. The experimental results show the superiority of this approach. In addition, it is the same for eSGGA which keeps the simplicity and efficiency than other EDAs even two probabilistic models are employed. Finally, since EDAs have not been extensively developed in the permutation-based problems (Ceberio et al., 2012), this work is of importance in the area of EDAs.

The rest of the paper is organized as follows: Section 2 introduces the scheduling problems that have been widely studied. Over the years, there has been an increasing interest for EDAs. In order to highlight the differences between our work and previous EDAs, we review many latest EDAs focusing on the scheduling problems in Section 3. Section 4 explains how to employ two probabilistic models in EDAs. It should also be noted that the probabilistic models are not applied to sample new solutions, but used to estimate the solution quality instead. Section 5 provides a detailed explanation of the eSGGA. Section 6 presents the experimental results on the performance of the proposed algorithm in treating the two scheduling problems. Section 7 draws the conclusions of this paper.

2. Problem formulations

The definitions of the single machine scheduling problems and the permutation flowshop problem are presented in Sections 2.1 and 2.2, respectively.

2.1. Single machine scheduling problems with total earliness and tardiness cost

In this paper, we tackle a deterministic single machine scheduling problem without release dates in an attempt to minimize the total sum of earliness and tardiness penalties. A detailed formulation of the problem is described as follows: A set of n independent jobs $\{J_1 J_2, \ldots J_n\}$ has to be scheduled without preemptions on a single machine that can handle at most one job at a time. It is assumed that the machine will be continuously available from time zero onwards and no unforced machine idle time is allowed. Job J_j, $j=1,2,\ldots,n$, becomes available for the processing at the beginning, requires a processing time p_j and should be completed on its due date d_j. To represent the processing sequence of each job, we introduce a binary variable x_{ij} in this model. This variable indicates whether the job J_i is assigned to the position j. For any given schedule determined by the x_{ij}, the earliness and tardiness of J_j can be defined as $E_j=\max(0,d_j-C_j)$ and $T_j=\max(0,C_j-d_j)$, respectively, where C_j is the completion time of J_j. The mathematical model of this problem is shown as follows:

$$\sum_{j=1}^{n}(\alpha_j E_j+\beta_j T_j) \tag{1}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{ij}=1 \quad \forall\, j=1 \text{ to } n \tag{2}$$

$$\sum_{j=1}^{n} x_{ij}=1 \quad \forall\, i=1 \text{ to } n \tag{3}$$

$$C_j-d_j+E_j-T_j=0 \tag{4}$$

$$x_{ij}\in\{0,1\} \tag{5}$$

The objective in Eq. (1) is then to find a schedule that minimizes the sum of the earliness and tardiness penalties of all jobs $\sum_{j=1}^{n}(\alpha_j E_j+\beta_j T_j)$, where α_j and β_j are the earliness and tardiness penalties of job J_j. Eq. (2) assigns a job to a position, we ensure that each position has a job to be processed. Eq. (3) guarantees each job to be assigned. Eq. (4) represents the relationship of the completion time and due day to the earliness and tardiness cost.

Considering the inclusion of both earliness and tardiness costs in the objective function, it is compatible with the philosophy of just-in-time production, emphasizing to produce goods only when they are needed. The early costs may represent the cost of completing a job early, the deterioration cost of perishable goods or a holding (stock) cost for finished goods. The tardy costs can represent the cost of rush shipping, lost sales and loss of goodwill. We assume that no unforced machine idle time is allowed and let the machine idle only when no job is available for processing. This assumption reflects that the cost of machine idleness is higher than the early cost stemming from completing any job before its due date in a production setting or the capacity of the machine is limited as compared with its demand so that the machine must remain in operation all the times. In Ow and Morton (1989) and Wu et al. (1993), they provided some specific examples of production arrangements with these characteristics. A characteristic of the deterministic problem is an assumption that the set of jobs is ready to process jobs in the beginning.

2.2. Permutation flowshop scheduling problems

Flowshops provide a convenient means to model serial manufacturing processes. The flowshop is a processing facility that consists of several machines on which jobs are processed in a sequential manner. In the PFSP, all the jobs follow the same processing order on each of the machines and every job is independent to each other. The objective is to find a permutation π^* that minimizes $C_{\max}(\pi)$. The PFSP to minimize the makespan can be defined as follows.

Suppose that there are n jobs and m machines. Let $p(i,j)$, $1 \le i \le n$, $1 \le j \le m$, be the processing time of job i on machine j and $\pi = (\pi_1, \ldots, \pi_n)$ be a job permutation (i.e., processing order of the jobs). Then, the completion times $C(\pi_i,j)$ are calculated as follows:

$$C(\pi_1,1) = p(\pi_1,1) \tag{6}$$

$$C(\pi_i,1) = C(\pi_{i-1},1) + p(\pi_i,1) \quad \text{for } i = 2,\ldots,n \tag{7}$$

$$C(\pi_1,j) = C(\pi_1,j-1) + p(\pi_1,j) \quad \text{for } j = 2,\ldots,m \tag{8}$$

$$C(\pi_i,j) = \max\{C(\pi_{i-1},j), C(\pi_i,j-1)\} + p(\pi_i,j) \quad \text{for } i = 2,\ldots,n,\ j = 2,\ldots,m \tag{9}$$

The makespan is

$$C_{\max}(\pi) = C(\pi_n,m) \tag{10}$$

Framinan et al. (2004) provided a review and a classification of the heuristics for the PFSP. Hejazi and Saghafian (2005) presented a comprehensive survey of the major results on this problem over the period from 1954 to 2004. Exact methods and heuristic methods (including evolutionary algorithms) were employed in these studies, which could also be used as a good reference for the PFSP with the objective of minimizing the makespan, denoted as $n/m/p/C_{\max}$. Ruiz and Maroto (2005) provided a comprehensive review and evaluation of the heuristics for the PFSP. Obviously, the heuristics have been a major methodology to deal with the PFSP. The reader may refer to Ruiz and Maroto (2005) for a detailed review of metaheuristics, including tabu search, simulated annealing, genetic algorithms, iterated local search, and hybrid techniques, for tackling various flowshop scheduling problems.

3. Importance of EDAs and a brief survey on recently developed EDAs

In contrast with implicit processing of building blocks in GA, EDAs explicitly depend on the used probability model. Sometimes the blocks are built based on simple selection and crossover are not effective enough to get optimum solution as they might not effectively preserve important patterns (Pelikan et al., 2002). The advantage of the probability model is its decisive factor affecting the performance of EDAs. The more accurate the probability model is, the more effective the algorithm will be in preventing the disruption of important building blocks (Lozano et al., 2006). Excellent surveys for previous EDAs already exist in Lozano et al. (2006), Hauschild and Pelikan (2011), and Ceberio et al. (2012).

Many attempts have been recently made in the area of EDAs to solve the scheduling problems or the combinatorial optimization problems. In Chang et al. (2008b), they proposed a hybrid framework to alternate between EDAs and genetic operators for solving the single machine scheduling problem. The benefit of a hybrid framework is that although EDAs improve the solution quality efficiently in first few runs, the loss of diversity grows very fast as more iterations are run (Shapiro, 2006; Branke et al., 2007; Chen et al., 2010). An univariate probability model was used in their algorithm. Apart from that, Liu et al. (2011) and Wang et al. (2011) combined the particle swarm optimization algorithms with EDAs to solve the PFSP and terminal assignment problems, respectively.

Jarboui et al. (2009) offered a hybrid approach, named EDA-VNS, that combined the EDAs with the variable neighborhood search (VNS) (Hansen and Mladenović, 2001), to solve the PFSPs by the minimization of the total flowtime. Their probabilistic model considered the order of the job queue and the building blocks of the jobs. This was the first attempt which took into account both the first order and the second order statistical information. In addition, VNS is an improvement procedure as the EDA is run. Jarboui et al. (2009) found EDA-VNS was effective in small benchmarks; however, when it came to larger size problems, VNS was better than EDA-VNS in terms of the objective values and the computational time. It might be worthwhile to explore why EDA-VNS did not outperform the VNS in large size benchmarks. After that, a new EDA in Pan and Ruiz (2012) also employed the job permutation and similar blocks of jobs to solve lot-streaming flowshop problems. Their definitions of the job permutation and similar blocks were different from those of Jarboui et al. (2009). Moreover, they also introduced a diversity measure to restart the evolutionary progress when the population diversity decreased to a certain level.

Instead of sampling new solutions from the probabilistic model, Chen et al. (2009) incorporated a new technique into the probabilistic model. Their algorithm was named Guided Memetic Algorithm (GMA) which reduced the possible neighborhood combinations using guided operations to remove inferior moves. The probabilistic models was regarded as a first approximation of a fitness surrogate which controlled the balance between genetic search and local search.

Later on, this major concept of GMA was further realized in the SGGA (Chen et al., 2012a) to guide the evolutionary direction. Their univariate probabilistic model acted as a fitness surrogate which predicted the fitness of the new solutions generated by a crossover or mutation operator. Based on the evaluation of candidate solutions, SGGA was able to select an appropriate solution by the genetic operators. Due to the time-complexity of SGGA is less than $O(n^2)$ when the probabilistic model was used to sample new solutions, SGGA was shown to be more efficient than the previous EDAs. Although the probabilistic model had been employed to be a fitness predictor before (Sastry et al., 2006; Brownlee et al., 2008; Chen et al., 2009; Lima et al., 2009), no previous approaches used the probabilistic model in the same way as that used in SGGA. Consequently, SGGA pointed to a new research direction in EDAs.

In Zhang and Li (2011), they applied the longest common subsequence together with EDAs to mine good building blocks. Their own results could be explained by a similar research on EA/G (Evolutionary Algorithm with Guided Mutation) (Zhang et al., 2005). EA/G also let the offspring inherit good genes from the best solution with a certain probability. This approach could speed up the convergence of EDAs. Both algorithms employed a parameter to control the population diversity.

Ceberio et al. (2012) examined 13 famous permutation-based EDAs on some combinatorial optimization problems, including the PFSP, the traveling salesman problem, the quadratic assignment problem, and the linear ordering problem. Among these EDAs, EHBSA$_{WT}$ (Edge Histogram Based Sampling Algorithm with Template) (Tsutsui, 2009) and NHBSA$_{WT}$ (Node Histogram Based Sampling Algorithm with Template) (Tsutsui, 2006) provided consistent results on their four studied problems. Given their excellent efforts, we adopted their empirical results to evaluate the proposed algorithm and recent EDAs.

Except the single machine scheduling problem and flowshop scheduling problems, Wang and Fang (2012a,b) proposed the EDAs to solve the multi-mode resource-constrained project scheduling problem (MRCPSP). Since the job sequence was not a permutation, they adapted the proposed EDAs by two problem-specific techniques. Each chromosome was encoded by the modes of activity and decoded by the multi-mode serial schedule generation scheme. Their encoding and decoding scheme could lead the EDAs to solve different categories of problems.

To conclude these recent EDAs, only a few researches took into account the order of the job queue and the building blocks of the jobs together (Chen et al., 2012b; Jarboui et al., 2009; Pan and Ruiz, 2012). Some researchers pointed out the ensemble of

different models generally outperform most of the individual models (Lim et al., 2010; Goel et al., 2007; Samad et al., 2008). This ensemble model approach might provide more accurate parental distributions in EDAs. EDAs might capture the problem structure well when variable interactions exist, particularly when there are variable interactions in scheduling problems. This research has conducted extensive experiments to valid the EDAs using only one model to generate probabilistic information as well as the algorithms using two statistical models. In addition, this paper has extended a latest introduced SGGA to eSGGA that employs univariate and bi-variate probabilistic models at the same time. The detailed information of the probabilistic models and eSGGA will be depicted in the next section.

4. Establishing probabilistic models and the fitness estimation

The probabilistic model is the core of EDAs (Lozano et al., 2006). Before the proposed algorithm is presented, we define both univariate and bi-variate probabilistic models used by eSGGA in Section 4.1. After that, the ensemble models are used to estimate the solution quality. We introduce this fundamental approach of eSGGA in Section 4.2.

4.1. Univariate and Bi-variate probabilistic models

To let probabilistic models extract information from the parental distribution, a set of M better chromosomes X^1, X^2, …, X^M are selected at the current generation t. Any selection method, such as the proportional selection and the tournament selection, can be used for this purpose. For simplicity, we adopt the 2-tournament selection in our method. The univariate model is introduced first. In Eq. (11), let a binary variable $X^i_{k[k]}$ represent the n jobs in chromosome i

$$X^i_{k[k]} = \begin{cases} 1 & \text{if job } k \text{ is at position } [k] \\ 0 & \text{otherwise} \end{cases}, \quad k=1,\ldots,n,\ i=1,\ldots,M \tag{11}$$

where $[k]$ is the position of job k in X. We also investigate the univariate model applied by Jarboui et al. (2009) and Pan and Ruiz (2012) in which the binary variable $X^i_{k[k]}$ before or at position $[k]$ in the population is taken into account, a different approach from ours. We find the definition of Eq. (11) is suitable for eSGGA in the prior experiments.

After we sum up the statistical information from all M chromosomes to the $X^i_{k[k]}$, the univariate model $\phi_{k[k]}(t)$ in Eq. (12) is obtained. $\phi_{k[k]}(t)$ represents the number of times that job k is at position $[k]$

$$\phi_{k[k]}(t) = \sum_{i=1}^{M} X^i_{k[k]}, \quad k=1,\ldots,n \tag{12}$$

When it comes to the bi-variate probabilistic model, we define a new binary variable $v^i_{k'k}(t)$ in Eq. (13). $v^i_{k'k}(t)$ indicates whether job k is immediately after the job k' in chromosome i.

$$v^i_{k'k}(t) = \begin{cases} 1 & \text{if job } k \text{ is after the job } k' \\ 0 & \text{otherwise} \end{cases}, \quad k=1,\ldots,n,\ i=1,\ldots,M \tag{13}$$

where $k \neq k'$. After we summarize the statistical information of $v^i_{k'k}(t)$ from the M chromosomes, the bi-variate statistical information $\psi_{k'k}(t)$ could be obtained in Eq. (14). $\psi_{k'k}(t)$ indicates the number of times that job k immediately after the job k'

$$\psi_{k'k}(t) = \sum_{i=1}^{M} v^i_{k'k}, \quad k=1,\ldots,n,\ k \neq k' \tag{14}$$

At each generation, we update the ensemble models by PBIL (Baluja, 1994) in Eqs. (14) and (15). The two statistics with learning continually modify the search space and then improve the performance. We may point out that both Jarboui et al. (2009) and Pan and Ruiz (2012) did not employ the learning algorithm. In this research, two learning rates, λ_ϕ and λ_ψ, are decided by Design-of-Experiment (DOE)

$$\phi_{k[k]}(t) = \phi_{k[k]}(t) \times (1.0-\lambda_\phi) + \phi_{k[k]}(t-1) \times \lambda_\phi, \quad \lambda_\phi \in (0,1) \tag{15}$$

$$\psi_{k'k}(t) = \psi_{k'k}(t) \times (1.0-\lambda_\psi) + \psi_{k'k}(t-1) \times \lambda_\psi, \quad \lambda_\psi \in (0,1) \tag{16}$$

After $\phi_{k[k]}(t)$ and $\psi_{k'k}(t)$ learn from the previous search, we consider how to form the probabilistic models which use both the statistics. Let $P_t(X_{k[k]})$ be the probability value of the job k in position $[k]$. This research likes to select a job k which has higher probability value than other jobs when the univariate and bi-variate statistical information are used. To do this, $\phi_{k[k]}(t)$ is multiplied by $\psi_{k'k}(t)$ which is proportioned to the summarized probability values of all unscheduled jobs that could be assigned at positions $[k]$. In addition, it is noticeable that when we select a job at the first position, $\psi_{k'k}(t)$ could be zero in the most cases while only few $\psi_{k'k}(t) > 0$. It causes a problem that only few jobs could be selected at the first position for producing offspring. The population diversity is decreased easily in this case (Pan and Ruiz, 2012). As a result, Pan and Ruiz (2012) use the univariate model to select a job at the first position.

It is reasonable to use the univariate model at the first position because there is no prior job; however, this research suggests that a random value with the uniform distribution is used according to our pilot experiments. As a result, the combined model $P_{k[k]}(t)$ is formulated as follows:

$$P_t(X_{k[k]}) = \begin{cases} U(0,1), & [k]=1 \\ \phi_{k[k]}(t) \times \psi_{k'k}(t) / \sum_{l \in \Omega} (\phi_{l[k]}(t) \times \psi_{k'l}(t)), & [k]=2,3,\ldots,n \end{cases} \tag{17}$$

where $k=1,\ldots,n$ and Ω is the set of the unscheduled jobs. In previous EDAs, P_t is used to sample new solutions. Instead of sampling new solutions, Section 4.2 uses P_t in a different way.

4.2. Predicting the solution quality

After the probabilistic model P_t is built in Eq. (17), the estimated quality of a solution X is defined as follows:

$$Q_t(X) = \prod_{k=1}^{n} P_t(X_{k[k]}) \tag{18}$$

where $[k]$ is the position of job k in X. $Q_t(X)$ is an estimation value of how 'promising' X is (Chen et al., 2012a). This estimation is much easier to compute. For example, when we modify a parent solution X to be two new solutions Z^1 and Z^2 by crossover or mutation operator, we calculate the difference of $Q_t(Z^1)$ and $Q_t(Z^2)$. If $Q_t(Z^1)-Q_t(Z^2) > 0$, it means that the new solution Z^1 might be better than the other solution Z^2 when they are both the variants of the parent solution X. Thus, Z^1 is used in the population and this characteristic let SGGA or eSGGA be distinguished from previous EDAs.

After the two probabilistic models and fitness estimation are defined in this section, how eSGGA guides the evolutionary direction is discussed in the next section.

5. Proposed methodology

Because eSGGA takes the advantages from SGGA, the major characteristic of eSGGA remains that the probabilistic model guides the evolutionary direction without sampling new solutions. The probabilistic model is used to evaluate the quality of candidate solutions generated by the crossover and mutation operators. Then, the algorithm decides a better candidate to be applied in the population beforehand. The major difference between eSGGA and SGGA is that the former employs the ensemble models, while the latter uses the univariate probabilistic model only. eSGGA is able to deal with the order information and the variable interactions of the jobs.

Although the main procedure of the eSGGA is similar to that of SGGA, the probabilistic model P_t of eSGGA is different from SGGA. P_t also alters the behavior and the calculations of self-guided crossover and self-guided mutation. We will explain the detailed differences later. The main procedure of the eSGGA is described as follows:

Notation:

- *Population*: a set of solutions.
- *PopSize*: the size of *Population*.
- G: the maximum number of generations.
- t: generation index.
- P_t: probabilistic model at generation t. P_t is composed of the univariate and bi-variate probability models.
- *Parentset*: the set of parent solutions selected from the current population.
- *Newset*: the set of new solutions generated at each generation.
- N: the size of *Newset*.

Algorithm:

1: Initialize *Population*
2: Evaluate(*Population*)
3: $t \leftarrow 0$
4: Initialize P_t
5: **while** $t < G$
6: *Parentset* = Select(*Population*)
7: $P_{t+1} \leftarrow$ modelupdate(*Parentset*, P_t)
8: Generate *Newset* by using self-guided crossover
9: Mutate every new solution in *Newset* by using self-guided mutation
10: Evaluate(*Newset*)
11: Use the solutions in *Newset* to replace the N worst solutions in *Population*.
12: $t \leftarrow t+1$
13: **end while**

In Line 1 of the Algorithm, a set of solutions are generated randomly or by using a heuristic to form the initial *Population*, such as NEH (Nawaz et al., 1983), a well-known heuristic approach for flowshop scheduling problems. The objective function values of all the solutions in *Population* are computed in Line 2. Line 3 initializes the generation index t. The probability model P_t is initialized to be $1/n$ in Line 4.

Line 6 selects a number of solutions from *Population* to form *Parentset* by a number of independent 2-tournament selections. Line 7 computes P_t introduced in the previous section. In Line 8, we employ the self-guided crossover operator (the details of this crossover are given in Section 5.1) to produce N solutions. In Line 9, we mutate each new solution generated in Line 8 by using the self-guided mutation operator (the details are given later). Line 10 computes the objective function values of all the new solutions. In Line 11, new solutions replace the N worst solutions in *Population* and the *Popsize*$-N$ best solutions in the old population will enter the new population. This elitism scheme is widely used in evolutionary algorithms controlled by an elitism ratio.

We use $Q_t(X)$ to guide crossover and mutation in Sections 5.1 and 5.2. In the following, we drop t in P and Q for simplicity.

5.1. Self-guided crossover operator

A crossover procedure is executed after better chromosomes have been selected. In the proposed self-guided crossover operator, the crossover operator attempts to mate a parent solution with the other appropriate chromosome that yields better offspring. To determine the other appropriate mated solution, a parameter TC is the number of tournament selection of different crossover candidates. The evaluation of the TC choices is to quality function Q defined in Eq. (18) to guide the evolutionary direction.

Because this research employs the two-point center crossover (Murata and Ishibuchi, 1994), the following description is based on this crossover method, where the sequence between the two cut points are altered. Suppose that the two random cut-points are named K and L, where $K < L$. When we set K is 3 and L is 5 in the 6-job permutation, Fig. 1 illustrates a parent solution {53|126|4} could be mated with other two candidates, including {21|653|4} and {65|431|2}. Z^1 and Z^2 are the offsprings if the parent solution X is mated with the two candidates Y^1 and Y^2, respectively.

We could realize that the order of the genes located inside the two random cut-points is changed. For the genes staying outside this range of K and L, they remain the same. In the two-point central crossover, we use the Q to estimate the solution quality of the offspring, the difference between any offspring Z^i and the parent solution X is

$$\Phi(Z^i,X) = Q(Z^i) - Q(X) = \left\{ \prod_{K \le k \le L+1} P(Z^i_{k[k]}) - \prod_{K \le k \le L+1} P(X_{k[k]}) \right\} \times \prod_{1 \le k < K} P(X_{k[k]}) \prod_{L+1 < k \le n} P(X_{k[k]}). \quad (19)$$

The larger the $\Phi(Z^i,X)$ is, the more likely that Z^i better than the X is. In addition, the differences between SGGA and eSGGA are the computation of the $P(X^i_{k[k]})$ (or $P(Z^i_{k[k]})$) which considers the univariate probability together with the bi-variate probability in Eq. (17), and the probability difference at position $L+1$ should be calculated because its prior job is changed. The self-guided two-point crossover works as follows:

Notation:

- X: the parent solution.
- TC: the number of candidates for mating.
- Y^i: $i = 1, \ldots, TC$: candidates for the second parent.
- Z^i: offspring, i.e., the new solution generated by crossover.

Self-guided two-point crossover:

1: Randomly select two crossover points $K < L$.
2: **for** $i = 1$ to TC **do**
3: Let X be the first parent and Y^i be mated with X, and the crossover points be K and L. Perform two-point crossover on X and Y^i and generate $Z^i = (z^i_1, \ldots, z^i_n)$.
4: **end for**

X: {53|126|4} Y^1: {21|653|4} Z^1: {53|216|4}
X: {53|126|4} Y^2: {65|431|2} Z^2: {53|612|4}

Fig. 1. Parent solution X could be mated with two candidates (Y^1 and Y^2).

5: Compare $\Phi(Z^i,X)$, $(i=1,\ldots,TC.)$. Output the Z^i with the largest Φ value as Z.

In this self-guided two-point crossover, we need to compare $\Phi(Z^i,X)$. Given

$$\Phi(Z^l,X)-\Phi(Z^m,X)=Q(Z^l)-Q(Z^m)$$
$$=\left\{\prod_{K\le k\le L+1}P(Z^l_{k[k]})-\prod_{K\le k\le L+1}P(Z^m_{k[k]})\right\}\times\prod_{1\le k<K}P(X_{k[k]})\prod_{L<k\le n}P(X_{k[k]}),\Phi(Z^l,X)>\Phi(Z^m,X) \quad (20)$$

if and only if

$$\prod_{K\le k\le L+1}P(Z^l_{k[k]})>\prod_{K\le k\le L+1}P(Z^m_{k[k]}) \quad (21)$$

We use (21) in the self-guided two-point crossover to compare $\Phi(Z^i,X)$ which is to select a better candidate to be mated with X. By using this approach, we preserve the original concept of SGGA. However, both univariate and bi-variate probability models are used in the self-guided crossover operator. In addition, our approach is not limited to two-point central crossover operator. It could be revised to other permutation-based crossover operators to measure the difference between the parent solution and the offspring.

When we do the self-guided mutation, Q is also used to estimate the difference in the mutated solution to the original solution, which is explained in the next section.

5.2. Self-guided mutation operator

Let X be a solution (i.e., a permutation from 1 to n) to be mutated. Suppose we swap the positions of job i and j in X and obtain Y. We can compute the quality difference in Q caused by this swap as follows:

$$\Delta_{ij}=Q(Y)-Q(X)=\{P(X_{i[j]})P'(X_{i+1[i+1]})P(X_{j[i]})P'(X_{j+1[j+1]})-P(X_{i[i]})P(X_{i+1[i+1]})P(X_{j[j]})P(X_{j+1[j+1]})\}\prod_{k\notin\{i,i+1,j,j+1\}}P(X_{k[k]}) \quad (22)$$

The larger the Δ_{ij} is, the more likely that Y is better than the X is. Both self-guided mutation and the same with the self-guided crossover consider the univariate and bi-variate probability information to compute the $P(X_{k[k]})$. In addition, the probability changes of the jobs after the swapped genes (i.e., the position at $[i+1]$ and $[j+1]$) should be considered as well.

In our proposed self-guided mutation operator, we randomly pick some pairs of jobs. This parameter is named *TM* which sets the number of swap operations. For each pair, we compute the differences in Q caused by swapping the positions of its two jobs in the parent solution X. Then, we conduct the swapping with the largest difference in X to produce a new solution. Formally, the self-guided mutation operator works as follows:

Notation:

- X: parent solution, i.e., the solution to be mutated.
- *TM*: the number of swappings to be considered.
- Y: offspring, i.e., the new solution.

Self-guided mutation:

1: Randomly select *TM* pairs of jobs: $\{i_1,j_1\},\ldots,\{i_{TM},j_{TM}\}$.
2: Compare $\Delta_{i_1j_1},\ldots,\Delta_{i_{TM}j_{TM}}$ and find the pair $\{i_k,j_k\}$ with the largest Δ value.
3: Swap the positions of jobs i_k,j_k in X to generate Y.

In the self-guided mutation operator, we need to compare the differences in *TM* quality. A simple way to compare two differences is given as follows:

$$\Delta_{ij}-\Delta_{lm}=\{P(X_{i[j]})P'(X_{i+1[i+1]})P(X_{j[i]})P'(X_{j+1[j+1]})P(X_{l[l]})\times P(X_{l+1[l+1]})P(X_{m[m]})P(X_{m+1[m+1]})-P(X_{i[i]})P(X_{i+1[i+1]})P(X_{j[j]})P(X_{j+1[j+1]})\times P(X_{l[m]})P'(X_{l+1[l+1]})P(X_{m[l]})P'(X_{m+1[m+1]})\}\times\prod_{k\notin\{i,i+1,j,j+1,l,l+1,m,m+1\}}P(X_{k[k]}),$$

$\Delta_{ij}>\Delta_{lm}$

if and only if

$$P(X_{i[j]})P'(X_{i+1[i+1]})P(X_{j[i]})P'(X_{j+1[j+1]})\times P(X_{l[l]})P(X_{l+1[l+1]})P(X_{m[m]})P(X_{m+1[m+1]})-P(X_{i[i]})P(X_{i+1[i+1]})P(X_{j[j]})P(X_{j+1[j+1]})\times P(X_{l[m]})P'(X_{l+1[l+1]})P(X_{m[l]})P'(X_{m+1[m+1]})>0 \quad (23)$$

Based on the above results, we can use (23) to compare Q differences in the self-guided mutation operator according to the largest Δ value.

In general, eSGGA is different from previous EDAs because eSGGA embeds the probabilistic model in the crossover and mutation operators to explore and exploit the solution space. In addition, eSGGA takes the ensemble model approach which may provide a robust result in different problems. To validate the performance, we test eSGGA on two different NP-Hard scheduling problems in the following section.

6. Experimental studies

This paper advocates the benefit of applying ensemble models to EDAs. eSGGA and other two EDAs, JEDA (Jarboui et al., 2009) and PREDA (Pan and Ruiz, 2012), fall into the same group. Three algorithms are used to compare the performance against other EDAs without using two probabilistic models. To evaluate the performance of these algorithms, we have conducted extensive computational experiments on single machine scheduling problems and permutation flowshop scheduling problems. The stopping criterion is the number of total examined evaluations and the replication of each algorithm on each instance is 30 times on a Windows 2003 server (Intel Xeon 3.2 GHz).

6.1. Single machine scheduling problems

There are numerous data sets published in the literature (Sourd and Kedad-Sidhoum, 2003) for the single machine scheduling problems, including 20, 30, 40, 50, 60, and 90 jobs. Each data set of 20 jobs up to 50 jobs contains 49 instances (problems) whereas there are 9 instances in the data set of 60 jobs and 90 jobs. We carry out our experiments on these total 214 instances.

In order to test the robustness of the algorithms, the stopping criteria are based on the number of examined solutions, which are 75,000, 100,000, and 125,000 solutions. This setting is identical to our previous research (Chen et al., 2010). The three solution evaluations stand for the different implementation environments allowing low, medium, and high level of CPU time with the same population size 100. Moreover, when we use 75,000 solutions, it means that the algorithms stop at generation 750.

We have compared five EDAs-based algorithms in our experiments. The brief concept of the compared algorithms and their own parameter settings are shown as follows:

- ACGA (Artificial Chromosome with Genetic Algorithms) (Chang et al., 2008a): This algorithm alternates the EDA operator with genetic operators. The probabilistic model belongs to a univariate

probabilistic model. By using the hybrid framework, both global and local information is used. We have employed the same code as in Chang et al. (2010).

- EA/G (Zhang et al., 2005): EA/G is also named as guided mutation. This algorithm inherits a proportion of salient genes from an elite. Then, the rest part of a new solution is copied from its parent and the rest is sampled from a univariate probabilistic model. We have employed the same data as in Chen et al. (2011).
- SGGA (Chen et al., 2012a): The probabilistic model is used to guide the evolutionary process of crossover and mutation. The probabilistic model also belongs to a univariate probabilistic model. We have employed the same code as in Chen et al. (2012a).
- JEDA (Jarboui et al., 2009): JEDA is the first algorithm which focuses on both the order of the jobs in the sequences and the similar blocks of jobs. Even though JEDA was considered to apply variable neighborhood search (VNS), we do not use VNS so that we could evaluate the performance of the probabilistic models in JEDA.
- PREDA (Pan and Ruiz, 2012): This algorithm was developed upon JEDA, but they used the different formulas to calculate sum of the probabilistic model and adopted restart to refresh model according to the diversity. The method of sampling chromosome is also different from JEDA. We have employed the same algorithm in Pan and Ruiz (2012) to design for single machine scheduling problems.
- eSGGA: We code this proposed algorithm in Java. The parameters of the eSGGA are fixed experimentally as follows: $Pc=0.9$, $Pm=0.5$, $Tc=4$, $Tm=2$, $\lambda_{\phi}=0.1$, $\lambda_{\psi}=0.9$, and a parameter $Interval=7$. *Interval* specifies the timing to refresh the information in the probabilistic models which may reduce the computation overhead.

Table 1 shows the partial statistical results of the average objective values on the six sets where the stopping criterion is 125,000. ACGA, EA/G, SGGA, JEDA and PREDA algorithms are compared with eSGGA. We adopted the experimental results from ACGA, EA/G and SGGA proposed on our earlier researches. The JEDA and PREDA algorithms are coded by the authors because the original paper did not conduct the experiments on the single machine scheduling problems. We already followed their description to code the EDAs into the comparisons. Both algorithms provide fundamental comparison results on the single machine scheduling problem and the flowshop scheduling problems.

We mark the best result in bold in Table 1. The partial results of Table 1 indicate that eSGGA is better than SGGA. In addition, the total average objective value of eSGGA is better than others. Even though PREDA gets 10 chances to have the lowest average objective values in the instances of job 20 to job 60, PREDA does not perform well in the large-size instances. The convergence behavior of these EDAs is discussed in Fig. 2.

In order to obtain a meaningful convergence plot at the generation 750, 1000, and 12,500, we have replicated the six algorithms 30 times. Once the 30 objective values of different generations are obtained, the data mean is obtained to depict Fig. 2. In Fig. 2, we point out that JEDA converges slowly in the early stage. Its performance is improved as the number of generations increases. Compared with JEDA, PREDA performs

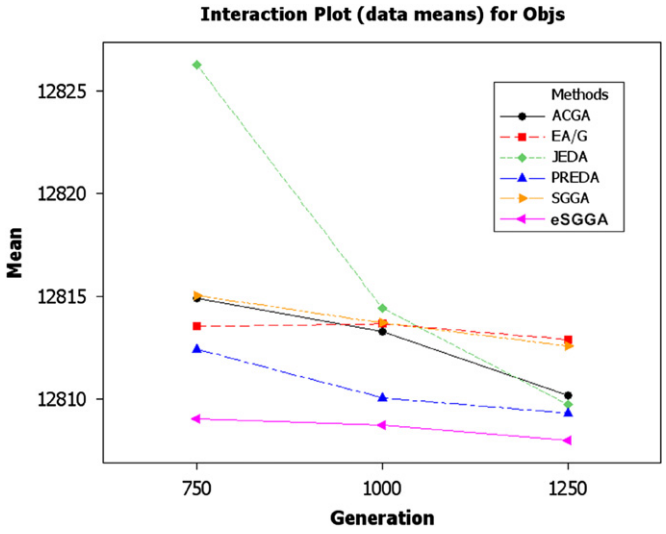

Fig. 2. Different solutions were examined and evaluated by ACGA, EA/G-GA, SGGA, JEDA, PREDA and eSGGA.

Table 1
Average objective values of single machine scheduling problems with the minimization of total earliness and tardiness (examined 125,000 solutions).

Instance	ACGA		EA/G		SGGA		PREDA		JEDA		eSGGA	
	OBJ	CPU	OBJ	CPU	OBJ	CPU	OBJ	CPU	OBJ	CPU	OBJ	CPU
sks222a	5288.07	0.83	5289.31	3.38	5288.48	0.59	**5287.66**	1.40	5288.90	0.86	5288.07	1.22
sks255a	2381.07	0.81	2380.90	3.35	2380.53	0.59	**2378.40**	1.36	2378.93	0.83	2380.53	1.10
sks288a	**3421.0**	0.81	**3421.0**	3.38	**3421.0**	0.59	**3421.0**	1.40	**3421.0**	0.91	**3421.0**	1.16
sks322a	11,572.97	1.19	11,575.53	5.84	11,571.27	0.83	**11,568.0**	2.46	11,569.37	1.27	11,571.17	1.64
sks355a	6057.13	1.15	6068.20	5.83	6057.07	0.83	**6056.4**	2.37	6057.20	1.17	6056.87	1.48
sks388a	**11,317.0**	1.14	11,319.30	5.84	**11,317.0**	0.82	**11,317.0**	2.36	**11,317.0**	1.14	**11,317.0**	1.45
sks422a	25,658.72	1.60	25,667.53	8.76	25,664.77	1.10	**25,658.07**	3.91	25,660.50	1.65	25,659.20	2.08
sks455a	6442.53	1.55	6423.90	8.76	6427.73	1.12	6434.20	4.34	**6414.33**	1.55	6417.07	1.93
sks488a	**16,862.0**	1.52	16,863.73	8.77	**16,862.0**	1.11	16,864.60	4.27	16,862.87	1.53	16,864.60	1.89
sks522a	**29,315.47**	2.08	29,323.40	12.28	29,319.53	1.45	29,315.63	6.36	29,321.73	2.11	29,319.37	2.51
sks555a	10,213.27	2.03	10,210.27	12.28	10,220.83	1.46	10,201.20	6.20	10,200.67	2.05	**10,199.33**	2.42
sks588a	**24,844.0**	1.98	24,848.37	12.31	24,845.17	1.45	**24,844.0**	6.27	**24,844.0**	2.03	**24,844.0**	2.36
sks622a	43,090.60	2.62	43,066.07	16.63	43,085.40	1.84	**43,059.8**	8.75	43,081.97	2.78	43,100.53	3.10
sks655a	16,186.60	2.57	**16,162.73**	16.73	16,191.13	1.85	16,267.07	8.34	16,198.03	2.69	16,171.73	2.98
sks688a	33,624.20	2.50	33,617.50	16.79	33,613.60	1.83	**33,601.3**	8.20	33,605.30	2.68	33,616.00	2.97
sks922a	88,887.77	4.80	**88,866.47**	34.93	88,868.70	3.38	88,905.10	17.12	88,909.27	5.37	88,872.30	5.54
sks955a	30,722.73	4.72	30,637.73	35.05	30,672.00	3.39	30,680.83	15.71	30,738.70	5.31	**30,628.73**	5.26
sks988a	81,988.20	4.64	81,989.77	35.11	81,989.17	3.39	81,985.63	15.09	82,030.90	4.66	**81,984.17**	4.96
Average	24,881.85	2.14	24,873.98	13.67	24,877.52	1.53	24,880.33	6.44	24,883.37	2.26	**24,872.87**	2.56

consistently by different stopping criteria. PREDA is more likely to achieve better performances than JEDA since PREDA uses better probabilistic models with a consideration of the population diversity. When eSGGA is compared with SGGA, it appears that eSGGA is relatively advantageous, as displayed on the convergence plot, because eSGGA yields promising results by different stopping criteria.

We also take a close look of SGGA and eSGGA about the convergence behavior. In Fig. 3, we select the best results from SGGA and eSGGA for sks655a instance and observe the convergence in different generations. eSGGA converges faster than SGGA before 625 generations. A close look at this comparison reveals that probabilistic model of eSGGA can present more effective than SGGA for single machine scheduling problems. That is, combination of univariate probabilistic model and bivariate probabilistic model can perform well than only using univariate probabilistic model. It follows from what has been said that the order of jobs and the similar block of jobs can provide important information for evolution.

Finally, to test the significance of these algorithms, ANOVA (Analysis of Variance) has been used to provide significant analysis results, as shown in Table 2. In the ANOVA table, the source indicates factors and combinations of factors which may influence the performance (or response). *DF* represents the degree of freedom and *SS* is the sum of squares which represent the deviation of each solution to the data mean. The mean square is equal to *SS* divided by *DF*. Then, we calculate the *F value* according to the mean square of the source factors to the mean square error and obtain the *P value*. If the *P value* of a factor source is < 0.05, it means that there is a significant difference in this factor (Montgomery, 2008).

Due to the *P values* of some factors are < 0.0001 in Table 2, a significant difference exists between these methods. Then, the Duncan grouping test is further conducted to differentiate the performance of these algorithms (see Table 3).

In the Duncan grouping test, *Mean* is the average value and *N* is the number of the observations. If the levels share the same alphabet (i.e., they are in the same group), there is no significant difference between them. Otherwise, they are significantly different (Montgomery, 2008). Based on the results shown in Table 3, it is evident from the Duncan grouping test that the levels are significantly different in their performance. To sum up, eSGGA is the best among the compared algorithms. There is no difference among JEDA, PREDA, and ACGA because those three algorithms are in the same group. However, they perform better than EA/G which adopts a univariate model without employing a hybrid framework like ACGA.

Table 3 shows that eSGGA is very attractive because the algorithm performs well and it is not time-consuming compared with other EDAs. We further examine the performance of the proposed EDAs on PFSPs in the next section.

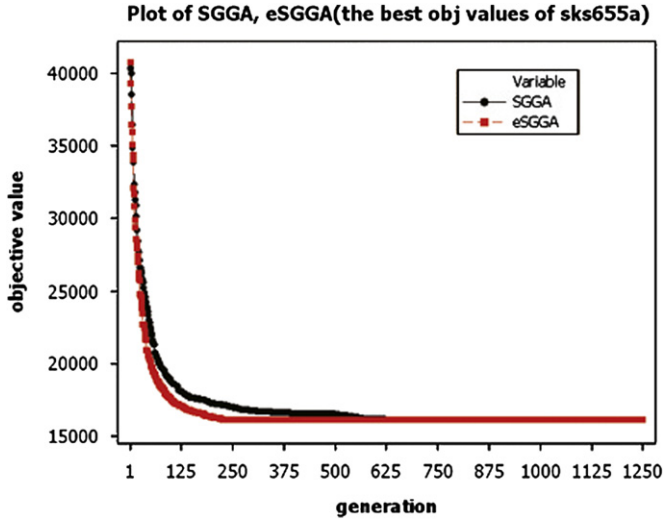

Fig. 3. A close look of SGGA and eSGGA.

6.2. Permutation flowshop scheduling problems

Ceberio et al. (2012) examined 13 well-known EDAs on the flowshop scheduling problems, 9 of which were applied for comparison, including UMDA (Univariate Marginal Distribution Algorithm) (Larrañaga et al., 2000), MIMIC (Mutual Information Maximization for Input Clustering) and $EBNA_{BIC}$ (Estimation of Bayesian Network Algorithm) in Bengoetxea et al. (2002), Tree (Pelikan and Tsutsui, 2007), IDEA-ICE (Iterated Density Estimation Evolutionary Algorithm-Induced Chromosome Elements Exchanger) (Bosman and Thierens, 2001), $EHBSA_{WT}$ (Tsutsui, 2009), $EHBSA_{WO}$ (Edge Histogram Based Sampling Algorithm without Template) (Tsutsui and Miki, 2002), $NHBSA_{WT}$, and $NHBSA_{WO}$ (Node Histogram Based Sampling Algorithm without Template) (Tsutsui, 2006). These EDAs only used one probabilistic model in the algorithm.

The four instance sets of the flowshop benchmarks were selected, such as the tai20 × 5, tai20 × 10, tai50 × 10, and tai100 × 20. In addition, the first six instances from each set were used. So, the amount of the 24 instances were employed to test these EDAs. To make a fair comparison, all the EDAs used the same population size ($10 \times n$) and stopping criterion ($100 \times n$). These algorithms ran 10 times on these instances and then collected the average error

Table 3
Duncan grouping Objective value at the stopping criterion of 125,000 examined solutions.

Duncan grouping	Mean	*N*	Methods
A	12,812.898	6420	EA/G
A	12,812.576	6420	SGGA
B	12,810.149	6420	ACGA
B	12,809.719	6420	JEDA
B	12,809.309	6420	PREDA
C	12,807.977	6420	eSGGA

Table 2
ANOVA Objective value at the stopping criterion of 125,000 examined solutions.

Source	DF	SS	Mean square	*F* Value	*P* Value
Instances	213	7.97E+12	374,103.87818	2.80E+07	< 0.0001
Method	5	121,384.4338	24,276.88677	18.18	< 0.0001
Instances × method	1065	4,616,675.241	4334.90633	3.25	< 0.0001
Error	37,236	49,724,692.66	1335.392971		
Corrected total	38,519	7.97E+12			

Table 4
Average error ratio of EDAs on Taillard instances with makespan criterion.

Instance	UMDA	MIMIC	$EBNA_{BIC}$	TREE	IDEA-ICE	$EHBSA_{WT}$	$EHBSA_{WO}$	$NHBSA_{WT}$	$NHBSA_{WO}$	PREDA	JEDA	SGGA	eSGGA
ta001	1.16	1.55	1.12	1.64	1.96	**0.30**	1.41	1.27	1.49	1.20	1.16	1.20	1.16
ta002	0.96	0.67	1.40	1.16	0.90	**0.05**	0.49	0.27	0.31	0.88	0.49	0.46	0.36
ta003	3.95	3.36	3.27	4.18	5.06	1.30	3.04	0.71	1.23	2.04	1.73	1.19	**0.50**
ta004	1.29	2.01	1.28	3.74	4.83	**0.19**	1.81	0.22	1.17	0.78	0.70	0.90	0.48
ta005	1.26	1.22	1.33	1.86	3.18	**0.51**	1.17	0.96	1.17	0.82	1.21	0.92	0.70
ta006	1.28	1.93	1.26	2.81	4.45	**0.00**	1.26	0.53	1.13	1.06	1.54	1.36	1.00
ta011	2.47	2.78	3.06	4.92	6.06	**0.53**	1.52	0.58	1.11	0.95	0.56	0.82	0.71
ta012	3.30	3.41	2.77	5.65	6.12	0.88	3.09	0.81	1.16	1.24	1.10	0.53	**0.28**
ta013	4.55	3.33	4.06	5.06	6.66	1.10	2.08	1.18	1.77	1.53	1.50	1.18	**0.82**
ta014	2.81	2.47	3.17	5.39	6.31	**0.37**	2.27	0.65	1.26	1.31	1.15	0.77	0.87
ta015	3.47	4.16	3.60	5.31	7.07	**0.44**	0.99	0.58	1.23	0.98	0.93	0.84	0.66
ta016	2.96	2.71	2.63	4.52	5.66	**0.55**	1.56	0.73	1.32	0.87	1.50	0.90	0.61
ta041	5.37	4.43	6.41	8.09	8.50	3.50	9.18	3.74	4.51	3.57	3.61	**2.95**	3.00
ta042	5.14	5.02	4.63	8.35	9.19	3.52	8.97	3.88	4.98	**2.34**	3.30	3.16	2.75
ta043	4.94	5.07	4.89	8.59	10.69	4.16	10.68	4.29	4.25	3.38	3.90	3.13	**2.74**
ta044	4.34	2.61	4.09	5.07	7.19	1.81	7.20	1.72	1.66	1.30	1.98	1.40	**1.05**
ta045	6.30	4.00	6.01	7.92	8.43	3.72	9.34	3.58	2.42	3.09	3.84	3.29	**2.15**
ta046	5.13	3.85	6.26	6.61	6.88	2.69	7.47	2.90	4.22	2.36	3.00	2.75	**2.35**
ta081	13.13	6.21	13.28	10.72	12.51	8.18	14.51	7.84	8.80	5.53	5.85	5.96	**5.46**
ta082	11.38	4.18	11.53	8.60	11.55	6.51	13.05	5.84	6.35	4.02	4.05	4.02	**3.37**
ta083	10.83	4.82	10.82	8.45	10.59	6.46	12.26	5.81	6.44	3.77	3.91	3.81	**3.29**
ta084	10.73	4.03	11.08	7.42	9.79	5.82	11.98	5.15	5.32	**2.95**	3.67	3.44	3.19
ta085	11.95	4.82	11.68	8.84	11.07	6.80	12.45	6.52	7.40	**3.97**	4.46	4.40	4.05
ta086	10.31	5.50	11.13	9.24	11.47	6.81	12.43	6.68	7.45	4.27	4.58	4.41	**3.99**
Avg.	5.38	3.51	5.45	6.01	7.34	2.76	6.26	2.77	3.26	2.26	2.49	2.24	**1.90**

ratio (*ER*). Here, the parameters of the eSGGA were fixed experimentally as follows: $Pc=0.9$, $Pm=0.5$, $Interval=7$, $Tc=2$, $Tm=4$, $\lambda_\phi=0.1$ and $\lambda_\psi=0.1$. *Interval* specifies the timing to refresh the information in the probabilistic models.

We collect the result of average error ratio (*ER*) because it is often used to evaluate the performance of algorithms applied to deal with the PFSPs. The error ratio of a solution X_i generated by an algorithm is calculated as Eq. (24). X_i is a solution generated by an algorithm and $C_{\max}(X_i)$ is the makespan of the solution X_i. U_i is the best known solution for Taillard instances which applied the version in year 2005.

$$ER_i = \frac{C_{\max}(X_i) - U_i}{U_i} \tag{24}$$

SGGA, JEDA, PREDA, and eSGGA were compared with the nine selected EDAs in Table 4. Through the 24 instances, the lowest average error ratio in each instance was marked in boldface. The results presented that eSGGA got 11 lowest average error ratio in the 24 instances. $EHBSA_{WT}$ performed well in nine instances. It indicated $EHBSA_{WT}$ was good at small-size problems. In terms of large-size problems (50–100 jobs), eSGGA performed well since eSGGA had 8 lowest average objective values out of the 12 instances. In addition to JEDA, PREDA, and eSGGA, they also outperformed the permutation-oriented EDAs which employed only one probabilistic model. Through the comparisons with EDAs using one or two probabilistic models, EDAs could benefit from integrate two probabilistic models. That is, the ensemble framework of EDAs might be effective when we solve different problems.

7. Conclusions

This paper investigated the concept of ensemble models in EDAs, simultaneously considering two probabilistic models. The ensemble models learn the different population characteristics which could generate more accurate parental distributions for EDAs. Given the ensemble models provide better individual information for EDAs, the ensemble models may improve robustness of optimization at minimum cost. In order to show how these two probabilistic models were combined and validate the performance of ensemble models in EDAs, we proposed eSGGA which applied both univariate and bi-variate probabilistic models together. eSGGA could represent better parental distribution than SGGA. Meanwhile, eSGGA also inherited the characteristics of SGGA, including the solution estimation during the crossover and mutation procedures. The evolutionary process remained guided by the probabilistic models used in eSGGA.

eSGGA was compared with two EDAs which also employed ensemble model approach, and some EDAs applying only one model on two NP-Hard scheduling problems. The experimental results showed that EDAs with ensemble models indeed performed well in contrast to EDAs with only one model. Most important of all, eSGGA outperformed other algorithms in the literature. As a result, this paper has indicated a potential direction to further improve the performance of EDAs through the framework of ensemble model for the future research. Moreover, because there is no theoretical analysis of the ensemble models in EDAs, it might be studied and then to know the performance improvement of using the ensemble model approach.

Acknowledgment

The authors like to thank the two anonymous reviewers, for their constructive comments, who led to a significant improvement of this article.

References

Aickelin, U., Burke, E.K., Li, J., 2007. An estimation of distribution algorithm with intelligent local search for rule-based nurse rostering. Journal of the Operational Research Society 58 (12), 1574–1585.

Baluja, S., 1994. Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning. Technical Report.

Bengoetxea, E., Larrañaga, P., Bloch, I., Perchant, A., Boeres, C., 2002. Inexact graph matching by means of estimation of distribution algorithms. Pattern Recognition 35 (12), 2867–2880.

Bosman, P.A.N., Thierens, D., 2001. Crossing the road to efficient ideas for permutation problems. In: A Workshop Within the 2001 Genetic and Evolutionary Computation Conference.
Branke, J., Lode, C., Shapiro, J.L., 2007. Addressing sampling errors and diversity loss in UMDA. In: Proceedings of the Ninth Annual Conference on Genetic and Evolutionary Computation. ACM, pp. 508–515.
Brownlee, A.E.I., McCall, J.A.W., Zhang, Q., Brown, D.F., 2008. Approaches to selection and their effect on fitness modelling in an estimation of distribution algorithm. In: IEEE Congress on Evolutionary Computation, 2008. IEEE, pp. 2621–2628.
Ceberio, J.E., Irurozki, A.M., Lozano, J., 2012. A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems. Progress in Artificial Intelligence 1 (1), 103–117.
Chang, P., Chen, S.H., Fan, C.Y., 2010. Generating artificial chromosomes with probability control in genetic algorithm for machine scheduling problems. Annals of Operations Research 180 (1), 197–211.
Chang, P.C., Chen, S.H., Fan, C.Y., 2008a. Genetic algorithm integrated with artificial chromosomes for multi-objective flowshop scheduling problems. Applied Mathematics and Computation 205 (2), 550–561.
Chang, P.C., Chen, S.H., Fan, C.Y., 2008b. Mining gene structures to inject artificial chromosomes for genetic algorithm in single machine scheduling problems. Applied Soft Computing Journal 8 (1), 767–777.
Chen, S.H., Chang, P.C., Cheng, T.C.E., Zhang, Q., 2012a. A self-guided genetic algorithm for permutation flowshop scheduling problems. Computers and Operations Research 39 (7), 1450–1457.
Chen, S.H., Chang, P.C., Zhang, Q., Wang, C.B., 2009. A guided memetic algorithm with probabilistic model. Information and Control 5 (12), 4753–4764.
Chen, S.H., Chen, M.C., Chang, P.C., Chen, Y.M., 2011. Ea/g-ga for single machine scheduling problems with earliness/tardiness costs. Entropy 13 (6), 1152–1169.
Chen, S.H., Chen, M.C., Chang, P.C., Zhang, Q., Chen, Y.M., 2010. Guidelines for developing effective Estimation of Distribution Algorithms in solving single machine scheduling problems. Expert Systems with Applications 37 (9), 6441–6451.
Chen, Y.M., Chen, M.C., Chang, P.C., Chen, S.H., 2012a. Extended artificial chromosomes genetic algorithm for permutation flowshop scheduling problems. Computers and Industrial Engineering 62 (2), 536–545.
Framinan, J.M., Gupta, J.N.D., Leisten, R., 2004. A review and classification of heuristics for permutation flow-shop scheduling with makespan objective. Journal of the Operational Research Society 55 (12), 1243–1255.
Goel, T., Haftka, R., Shyy, W., Queipo, N., 2007. Ensemble of surrogates. Structural and Multidisciplinary Optimization 33 (3), 199–216.
Hansen, P., Mladenović, N., 2001. Variable neighborhood search: principles and applications. European Journal of Operational Research 130 (3), 449–467.
Hauschild, M., Pelikan, M., 2011. An introduction and survey of estimation of distribution algorithms. Swarm and Evolutionary Computation 1 (3), 111–128.
Hejazi, S.R., Saghafian, S., 2005. Flowshop-scheduling problems with makespan criterion: a review. International Journal of Production Research 43 (14), 2895–2929.
Jarboui, B., Eddaly, M., Siarry, P., 2009. An estimation of distribution algorithm for minimizing the total flowtime in permutation flowshop scheduling problems. Computers and Operations Research 36 (9), 2638–2646.
Larrañaga, P., Etxeberria, R., Lozano, J., Peña, J., 2000. Optimization in continuous domains by learning and simulation of gaussian networks. In: A Workshop within the 2000 Genetic and Evolutionary Computation Conference. Citeseer.
Lenstra, J., Kan, A., Brucker, P., 1977. Complexity of machine scheduling problems. Annals of Discrete Mathematics 1, 343–362.
Li, G., 1997. Single machine earliness and tardiness scheduling. European Journal of Operational Research 96 (3), 546–558.
Lim, D., Jin, Y., Ong, Y., Sendhoff, B., 2010. Generalizing surrogate-assisted evolutionary computation. IEEE Transactions on Evolutionary Computation 14 (3), 329–355.
Lima, C., Pelikan, M., Lobo, F., Goldberg, D., 2009. Loopy substructural local search for the Bayesian optimization algorithm. In: Engineering Stochastic Local Search Algorithms. Designing, Implementing and Analyzing Effective Heuristics, pp. 61–75.
Liu, H., Gao, L., Pan, Q., 2011. A hybrid particle swarm optimization with estimation of distribution algorithm for solving permutation flowshop scheduling problem. Expert Systems with Applications 38 (4), 4348–4360.
Lozano, J.A., Larranaga, P., Inza, I., 2006. Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms. Springer.
Minella, G., Ruiz, R., Ciavotta, M., 2008. A review and evaluation of multiobjective algorithms for the flowshop scheduling problem. INFORMS Journal on Computing 20 (3), 451–471.
Montgomery, D., 2008. Design and Analysis of Experiments. John Wiley & Sons Inc.
Murata, T., Ishibuchi, H., 1994. Performance evaluation of genetic algorithms for flowshop scheduling problems. In: Proceedings of the First IEEE World Congress on Computational Intelligence. IEEE, pp. 812–817.
Nawaz, M., Enscore, E.E., Ham, I., 1983. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega 11 (1), 91–95.
Ow, P.S., Morton, T.E., 1989. The single machine early/tardy problem. Management Science 35 (2), 177–191.
Pan, Q., Ruiz, R., 2012. An estimation of distribution algorithm for lot-streaming flow shop problems with setup times. Omega 40 (2), 166–180.
Pelikan, M., Goldberg, D.E., Lobo, F.G., 2002. A survey of optimization by building and using probabilistic models. Computational Optimization and Applications 21 (1), 5–20.
Pelikan, M., Tsutsui, S., Kalapala, R., 2007. Dependency trees, permutations, and quadratic assignment problem. In: Proceedings of the Ninth Annual Conference on Genetic and Evolutionary Computation, July, pp. 7–11.
Reeves, C.R., 1995. A genetic algorithm for flowshop sequencing. Computers and Operations Research 22 (1), 5–13.
Ruiz, R., Maroto, C., 2005. A comprehensive review and evaluation of permutation flowshop heuristics. European Journal of Operational Research 165 (2), 479–494.
Samad, A., Kim, K., Goel, T., Haftka, R., Shyy, W., 2008. Multiple surrogate modeling for axial compressor blade shape optimization. Journal of Propulsion and Power 24 (2), 302–310.
Sastry, K., Pelikan, M., Goldberg, D.E., 2006. Efficiency enhancement of estimation of distribution algorithms. Computational Intelligence 33, 161–185.
Shapiro, J., 2006. Diversity loss in general estimation of distribution algorithms. In: Parallel Problem Solving from Nature—PPSN IX. Springer, pp. 92–101.
Sourd, F., Kedad-Sidhoum, S., 2003. The one-machine problem with earliness and tardiness penalties. Journal of Scheduling 6 (6), 533–549.
Tsutsui, S., 2006. A comparative study of sampling methods in node histogram models with probabilistic model-building genetic algorithms. In: IEEE International Conference on Systems, Man and Cybernetics, 2006, SMC'06, vol. 4. IEEE, pp. 3132–3137.
Tsutsui, S., 2009. Effect of using partial solutions in edge histogram sampling algorithms with different local searches. In: IEEE International Conference on Systems, Man and Cybernetics, 2009. IEEE, pp. 2137–2142.
Tsutsui, S., Miki, M., 2002. Solving flow shop scheduling problems with probabilistic model-building genetic algorithms using edge histograms. In: Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution And Learning (SEAL02).
Wang, J., Cai, Y., Zhou, Y., Wang, R., Li, C., 2011. Discrete particle swarm optimization based on estimation of distribution for terminal assignment problems. Computers and Industrial Engineering 60 (4), 566–575.
Wang, L., Fang, C., 2012a. An effective estimation of distribution algorithm for the multi-mode resource-constrained project scheduling problem. Computers and Operations Research 39 (2), 449–460.
Wang, L., Fang, C., 2012b. A hybrid estimation of distribution algorithm for solving the resource-constrained project scheduling problem. Expert Systems with Applications 39 (3), 2451–2460.
Wu, S.D., Storer, R.H., Chang, P.C., 1993. One-machine rescheduling heuristics with efficiency and stability as criteria. Computers and Operations Research 20 (1), 1–14.
Zhang, Q., Sun, J., Tsang, E., 2005. An evolutionary algorithm with guided mutation for the maximum clique problem. IEEE Transactions on Evolutionary Computation 9 (2), 192–200.
Zhang, Y., Li, X., 2011. Estimation of distribution algorithm for permutation flow shops with total flowtime minimization. Computers and Industrial Engineering 60 (4), 706–718.