



SOLVING FLOW SHOP SCHEDULING PROBLEMS WITH PROBABILISTIC MODEL-BUILDING GENETIC ALGORITHMS USING EDGE HISTOGRAMS

Shigeyoshi Tsutsui^{*1} and *Mitsunori Miki*^{*2}

^{*1} Department of Management and Information Science, Hannan University
5-4-33 Amamihigashi, Matsubara, Osaka 580-5802 Japan
tsutsui@hannan-u.ac.jp

^{*2} Department of Knowledge Engineering and Computer Sciences, Doshisha University
Kyo-tanabe, Tatara Miyakodani 1-3, Kyoto, 610-0321, Japan
mmiki@mail.doshisha.ac.jp

ABSTRACT

In evolutionary algorithms based on probabilistic modeling, the offspring population is generated according to the estimated probability density model of the parent instead of using recombination and mutation operators. In this paper, we have proposed a probabilistic model-building genetic algorithms (PMBGAs) for solving flow shop problems using edge histogram based sampling algorithms (EHBSAs). Two types of sampling algorithms EHBSA/WO and EHBSA/WT were presented. The results showed EHBSA/WT worked fairly well with a small population size in the test problems used.

1. INTRODUCTION

Genetic Algorithms (GAs) [7] are widely used as robust searching schemes in various real world applications, including function optimization, optimum scheduling, and many combinatorial optimization problems. Traditional GAs start with a randomly generated population of candidate solutions (individuals). From the current population, better individuals are selected by the selection operators. The selected solutions produce new candidate solutions by applying recombination and mutation operators.

Recently, there has been a growing interest in developing evolutionary algorithms based on probabilistic models [13], [19]. In this scheme, the offspring population is generated according to the estimated probabilistic model of the parent population instead of using traditional recombination and mutation operators. The model is expected to reflect the problem structure, and as a result it is expected that this approach provides more effective mixing capability than recombination operators

in the traditional GAs. These algorithms are called probabilistic model-building genetic algorithms (PMBGAs) or estimation of distribution algorithms (EDAs). In a PMBGA, better individuals are selected from an initially randomly generated population like in standard GAs. Then, the probability distribution of the selected set of individuals is estimated and new individuals are generated according to this estimate, forming candidate solutions for the next generation. The process is repeated until the termination conditions are satisfied.

Many studies on PMBGAs have been performed in discrete (mainly binary) domain and there are several attempts to apply PMBGAs in continuous domain. However, a few studies on PMBGAs in permutation representation domain are found. According to [Pelikan 99b], these PMBGAs in binary string representation can be classified into three classes depending on the complexity of models they use; (1) no interactions, (2) pairwise interactions, and (3) multivariate interactions. In models with no interactions, interactions among variables are treated independently. Algorithms in this class work well on problems which have no interactions among variables. These algorithms include the PBIL [1], cGA [8], and UMDA [11] algorithms. In pairwise interactions, some pairwise interactions among variables are considered. These algorithms include the MIMIC algorithm [6], the algorithm using dependency trees [2]. In models with multivariate interactions, algorithms use models that can cover multivariate interactions. Although the algorithms require increased computational time, they work well on problems which have complex interactions among variables. These algorithms include ECGA [9] and BOA [12, 14].

Studies to apply PMBGAs in continuous domains have also been made. These include continuous PBIL with Gaussian distribution [15] and a real-coded variant of PBIL with iterative interval updating [15]. The

UMDA and MIMIC were introduced in continuous domain. All above algorithms do not cover any interactions among the variables. In EGNA [10], a Gaussian network learns to estimate a multivariate Gaussian distribution of the parent population. In [3], two density estimation models, i.e., the normal distribution, and the histogram distribution are discussed. These models are intended to cover multivariate interaction among variables. In [4], it is reported that the normal distribution models have shown good performance. In [5], a normal mixture model combined with a clustering technique is introduced to deal with non-linear interactions. In [20], an evolutionary algorithm using marginal histogram models in continuous domain was proposed.

A study on PMBGAs in permutation domain is found in [18]. In it, PMBGAs are applied to solving TSP using two approaches. PMBGAs are also applied to solve job shop scheduling problems and graph matching problems [19]. Previous study [21] proposed an approach of PMBGAs in permutation representation domain, focusing solving the Traveling Salesman Problem (TSP) and compared its performance with traditional recombination operators. In the approach, we developed a symmetrical edge histogram matrix from the current population, where an edge is a link between two nodes in a string. We then sample nodes of a new string according to the edge histogram matrix. We called this method the edge histogram based sampling algorithm (EHBSA). We proposed two types of EHBSAs, edge histogram based sampling algorithm without template (EHBSA/WO) and edge histogram based sampling algorithm with template (EHBSA/WT). The results showed EHBSA/WT worked fairly well on the test suit taken from TSPLIB.

In this paper, we extend the EHBSAs to solving flow shop scheduling problem, a typical well known problem in the area of scheduling. In a flow shop problem, each string represents a sequence of jobs to be processed. For example, string $s = \{1, 2, 3, 0\}$ means that job 1 is first processed, then jobs 2, 3, and 0 follows in this sequence. In this case, there are four edges, i.e., 1->2, 2->3, 3->0, and 0->1. Thus in the flow shop problem, each edge is directional and the edge histogram model becomes asymmetrical. This is a big difference from previous study in [21]. In Section 2 of this paper, the EHBSA is described. The empirical analysis is given in Section 3. Section 4 concludes the paper.

2. EDGE HISTOGRAM BASED SAMPLING ALGORITHM FOR FLOW SHOP SCHEDULING

This section describes how the edge histogram based sampling algorithm (EHBSA) can be used to (1) model promising solutions and (2) generate new solutions by simulating the learned model.

2.1 The basic description of the algorithm

An edge is a link or connection between two nodes and has important information about the permutation string. Some crossover operators, such as Edge Recombination (ER) [25] and enhanced ER (eER) [24] which are used in traditional two-parent recombination, use the edge distribution only in the two parents string. The basic idea of the edge histogram based sampling algorithm (EHBSA) is to use the edge histogram of the whole population in generating new strings.

The algorithm starts by generating a random permutation string for each individual population of candidate solutions. Promising solutions are then selected using any popular selection scheme. An edge histogram matrix (EHM) for the selected solutions is constructed and new solutions are generated by sampling based on the edge histogram model. New solutions replace some of the old ones and the process is repeated until the termination criteria are met.

2.2 Developing edge histogram matrix for flow shop scheduling

Previous study [21] proposed a symmetrical edge histogram matrix. In this paper, we represent it as $EHM_{(s)}$ to distinguish an asymmetrical edge histogram matrix. In scheduling problem, an edge in a string is directional. Thus, we must consider an asymmetrical edge histogram matrix $EHM_{(A)}$.

Let string of k th individual in population $P(t)$ at generation t represent as $s'_k = (\pi'_k(0), \pi'_k(1), \dots, \pi'_k(L-1))$. ($\pi'_k(0), \pi'_k(1), \dots,$ and $\pi'_k(L-1)$) are the permutation of $(0, 1, \dots, L-1)$ representing a possible job sequence, where L is the length of the permutation.

An asymmetric edge histogram matrix $EHM_{(A)}^t$ (e'_{ij}) ($i, j = 0, 1, \dots, L-1$) of population $P(t)$ consists of L^2 elements as follows:

$$e'_{i,j} = \begin{cases} \frac{N}{k=1} \delta_{i,j}(s'_k) + \varepsilon & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases} \quad (1)$$

where N is the population size, $\delta_{ij}(s'_k)$ is a delta function defined as

$$\delta_{i,j}(s'_k) = \begin{cases} 1 & \text{if } \exists h [h \in \{0, 1, \dots, L-1\} \\ & \wedge \pi'_k(h) = i \wedge \pi'_k((h+1) \bmod L) = j] \\ 0 & \text{othersise} \end{cases} \quad (2)$$

and ε ($\varepsilon > 0$) is a bias to control *pressure* in sampling nodes just like those used for adjusting the selection pressure in the proportional selection in GAs. The average number of edges of element e'_{ij} (ij) in $EHM_{(A)}^t$ is

$LN/(L^2-L) = N/(L-1)$. So, ε is determined by a bias ratio B_{ratio} ($B_{\text{ratio}} > 0$) of this average number of edges as

$$\varepsilon = \frac{N}{L-1} B_{\text{ratio}} \quad (3)$$

A smaller value of B_{ratio} reflects the real distribution of edges in sampling of nodes and a bigger value of B_{ratio} will give a kind of perturbation in the sampling. Example of $EHM_{(S)}$ and $EHM_{(A)}$ are shown in Fig. 1.

2.3 Sampling methods

In this subsection, we describe how to sample a new string from the edge histogram matrix $EHM_{(A)}$. As for $EHM_{(A)}$ in [21], we use two types of sampling methods, edge histogram based sampling algorithm without template (EHBSA/WO), and edge histogram based sampling algorithm with template (EHBSA/WT).

2.3.1 Edge histogram based sampling algorithm without template (EHBSA/WO)

In a symmetrical edge histogram matrix such as for the symmetrical TSP, the absolute positions (loci) of a string have no meaning. For example, string $s_1 = (0, 1, 2, 3, 4)$ and string $s_2 = (4, 0, 1, 2, 3)$ represent the same tour. However, in scheduling problems, these two string represent completely different two solutions. Thus, we must consider how to determine the initial position and what node we assign to the position. In this paper, we propose two types of EHBSA/WO, EHBSA/WO1 and EHBSA/WO2.

a) *EHBSA/WO1*: Let us represent a new individual permutation by c []. In EHBSA/WO1, the initial position is always the first position, i.e. $c[0]$. The value for $c[0]$ is taken from a *pseud template individual* PT [] which is taken from current population $P(t)$ randomly, and a new individual permutation c [] is generated straightforwardly as follows:

1. Set the position counter $p \leftarrow 0$.
2. Choose a pseud template PT [] from $P(t)$.
3. Obtain first node as $c[0] \leftarrow PT[0]$.

$$\begin{array}{ccc}
 s_1^t = (0, 1, 2, 3, 4) & \begin{pmatrix} 0 & 3.1 & 2.1 & 2.1 & 3.1 \\ 3.1 & 0 & 4.1 & 3.1 & 0.1 \\ 2.1 & 4.1 & 0 & 1.1 & 3.1 \\ 2.1 & 3.1 & 1.1 & 0 & 4.1 \\ 3.1 & 0.1 & 3.1 & 4.1 & 0 \end{pmatrix} & \begin{pmatrix} 0 & 2.05 & 1.05 & 2.05 & 0.05 \\ 1.05 & 0 & 2.05 & 2.05 & 0.05 \\ 1.05 & 2.05 & 0 & 1.05 & 1.05 \\ 0.05 & 1.05 & 0.05 & 0 & 4.05 \\ 3.05 & 0.05 & 2.05 & 0.05 & 0 \end{pmatrix} \\
 \text{(a) } P(t) & \text{(b) } EHM_{(S)}^t & \text{(c) } EHM_{(A)}^t
 \end{array}$$

4. Construct a roulette wheel vector rw [] from $EHM_{(A)}$ as $rw[j] = e'_{c[p]j}$ ($j=0, 1, \dots, L-1$).
5. Set to 0 previously sampled nodes in rw ($rw[c[i]] = 0$ for $i=0, 1, \dots, p$).
6. Sample next node $c[p+1]$ with probability $rw[x] / \sum_{j=0}^{L-1} rw[j]$ using roulette wheel rw [].
7. Update the position counter $p \leftarrow p+1$.
8. If $p < L-1$, go to Step 4.
9. Obtain a new individual string c [].

b) *EHBSA/WO2*: In EHBSA/WO1, the initial sampling position is fixed to 0. On the other hand in EHBSA/WO2, the initial sampling position is chosen from $[0, L-1]$ randomly as follows:

1. Obtain random initial sampling position p_{initial} from $[0, L-1]$.
2. Choose a pseud template PT [] from $P(t)$.
3. Obtain first node as $c[p_{\text{initial}}] \leftarrow PT[p_{\text{initial}}]$.
4. Set the position counter $p \leftarrow p_{\text{initial}}$.
5. Construct a roulette wheel vector rw [] from $EHM_{(A)}$ as $rw[j] = e'_{c[p]j}$ ($j=0, 1, \dots, L-1$).
6. Set to 0 previously sampled nodes in rw ($rw[c[i]] = 0$ for $i = p_{\text{initial}}, (p_{\text{initial}}+1) \bmod L, \dots, p$).
7. Sample next node $c[(p+1) \bmod L]$ with probability $rw[x] / \sum_{j=0}^{L-1} rw[j]$ using roulette wheel rw [].
8. Update the position counter $p \leftarrow (p+1) \bmod L$.
9. If $(p+1) \bmod L \neq p_{\text{initial}}$, go to Step 5.
10. Obtain a new individual string c [].

2.3.2 Edge histogram based sampling algorithm with template (EHBSA/WT)

$EHM_{(A)}$ described in Section 2.2 is in a marginal edge histogram. It has no graphical structure. EHBSA/WT is intended to make up for this disadvantage by using a template in sampling a new string, and is the same with EHBSA/WT proposed for $EHM_{(S)}$ in [21]. In generating each new individual, a *template individual* is chosen from $P(t)$ (normally, randomly). The n ($n > 1$) cut points are applied to the template randomly. When n cut points

Fig. 1. Examples of symmetrical and asymmetric edge histogram matrices for $N = 5$, $L = 5$, $B_{\text{ratio}} = 0.04$

are obtained for the template, the template should be divided into n segments. Then, we choose one segment randomly and sample nodes for the segment. Nodes in other $n-1$ segments remain unchanged. We denote this sampling method by EHBSA/WT/ n . Since average length of one segment is L/n , EHBSA/WT/ n generates new strings which are different L/n nodes on average from their templates. Fig. 2 shows an example of EHBSA/WT/3. In this example, nodes of new string from after cut[2] and before cut[1] are the same as the nodes of the template. New nodes are sampled from cut[1] up to, but not including, cut[2] based on the $EHM_{(A)'}'$.

The sampling method for EHBSA/WT/ n is as follows:

1. Choose a template $T[]$ from $P(t)$.
2. Obtain sorted cut point array $\text{cut}[0], \text{cut}[1], \dots, \text{cut}[n-1]$ randomly.
3. Choose a cut point $\text{cut}[l]$ by generating random number $l \in [0, n-1]$.
4. Copy nodes in $T[]$ to $c[]$ from after $\text{cut}[(l+1) \bmod n]$ and before $\text{cut}[l]$.
5. Set the position counter $p \leftarrow \text{cut}[l]-1$.
6. Construct a roulette wheel vector $rw[]$ from $EHM_{(A)}'$ as $rw[j] = e^{c_{[p],j}}$ ($j=0, 1, \dots, L-1$).
7. Set to 0 copied and previously sampled nodes in $rw[]$ ($rw[c[i)] = 0$ for $i = \text{cut}[(l+1) \bmod n], \dots, p$).
8. Sample next node $c[(p+1) \bmod L]$ with probability $rw[x] / \sum_{j=0}^{L-1} rw[j]$ using roulette wheel $rw[]$.
9. Update the position counter $p \leftarrow (p+1) \bmod L$.
10. If $(p+1) \bmod L = \text{cut}[(l+1) \bmod n]$, go to Step 6.
11. Obtain a new individual string $c[]$.

3. EMPIRICAL STUDY

3.1 Experimental methodology

3.1.1 Evolutionary models

Evolutionary model is the same as the model used for

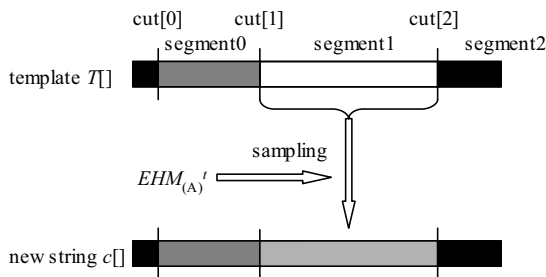


Fig. 2. An example of EHBSA/WT/3

symmetrical $EHM_{(S)}'$ in [21] as follows:

(1) *Evolutionary model for EHBSA/WT*: Let the population size be N , and let it, at time t , be represented by $P(t)$. The population $P(t+1)$ is produced as follows (Fig. 3):

1. Edge distribution matrix $EHM_{(A)}'$ described in Subsection 3.2 is developed from $P(t)$.
2. A template individual $T[]$ is selected from $P(t)$ randomly.
3. EHBSA/WT described in Subsection 3.3.2 is performed using $EHM_{(A)}'$ and $T[]$, and generate a new individual $c[]$.
3. The new $c[]$ individual is evaluated.
4. If $c[]$ is better than $T[]$, then $T[]$ is replaced with $c[]$, otherwise $T[]$ remains, forming $P(t+1)$.

(2) *Evolutionary model for EHBSA/WO1 and EHBSA/WO2*: Evolutionary model for EHBSA/WO is basically the same as the model for EHBSA/WT except EHBSA/WO uses a pseudo template $PT[]$.

(3) *Evolutionary model for two-parent recombination operators*: To compare the performance of proposed methods with the performance of traditional two-parent recombination operators, we designed an evolutionary model for two-parent recombination operators. For fair comparison, we design it as similar as possible to that of the EHBSA. We generate only one child from two parents. Using one child from two parents is already proposed for designing the GENITOR algorithm by Whitley et al. [25]. In our generational model, two parents are selected from $P(t)$ randomly. No bias is used in this selection. Then we apply a recombination operator to produce one child. This child is compared with its parents. If the child is better than the worst parent, then the parent is replaced with the child.

3.1.2 Flow shop problems and performance measures

General assumptions of flow shop scheduling problems can be described as follows: Jobs are to be processed on multiple machines sequentially. There is one machine at

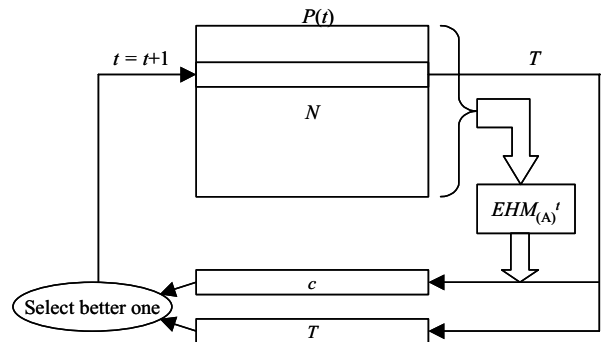


Fig. 3. Evolutionary model for EHBSA/WT

each stage. Machines are available continuously. A job is processed on one machine at a time without preemption, and a machine processes no more than one job at a time. In this paper, we assume that L jobs are processed in the same order on m machines. This means that our flow shop scheduling is the L -job and m -machine sequence problem. The purpose of this problem is to determine the sequence of L jobs. This sequence is denoted by a permutation string of $\{0, 1, \dots, L-1\}$. The problem to find a permutation which minimizes the *makespan* (i.e., the completion time of all jobs).

As test problems, we generated two flow shop scheduling problems, 20-job and 10-machine, and 30-job and 10-machine. In design each problem, we specified the processing time of each job at each machine as a random integer in the interval $[1, 99]$.

We compared EHBSA with popular order based two-parent recombination operators, namely, the original order crossover OX [23], the enhanced edge recombination operator eER [24], and the partially mapped crossover [7]. Ten (10) runs were performed. Each run continued until the population was converged, or evaluations reached E_{\max} . Values of E_{\max} were 200000. The performance is measured by the minimum makespan in ten run (Best) and average of makespans in ten runs (Aver). Population sizes of 60, 120, 240 were used for EHBSA, and 60, 120, 240, 480,960 for other operators, respectively. As to the bias ratio B_{ratio} in Eq. 3, a B_{ratio} values of 0.05 was used.

3.1.3 Blind search

In solving scheduling problems using GAs, mutation operators play an important role. Several types of mutation operators are proposed. Also, it is well known that combining GAs with local optimization methods or heuristics greatly improve the performance of the algorithms. However, in this experiment, we use no mutation and no heuristic to see the pure effect of applying proposed algorithms. Thus, the algorithm is a *blind search*.

3.2 Empirical analysis of results

Results on 20-job and 10-machine problem are shown in Table 1. EHBSA/WO1 and EHBSA/WO2 showed obviously worse performance compared with EHBSA/WT and other two-parent recombination operators. We can see that there is no meaningful difference between EHBSA/WO1 and EHBSA/WO2. As was found in [21] with TSP, EHBSA/WT shows much better performance than EHBSA/WO. EHBSA/WT/5 with $N = 60$ found the minimum makespan (1519). In the other operators, PMX showed good performance. The minimum makespan of PMX with $N = 960$ was 1520. OX also showed good performance showing Best = 1520 with $N = 480$. eER

which showed good performance in TSP [21] showed poor performance in this problem. Comparing the performance of EHBSA/WT with other operators, EHBSA/WT is almost the same with PMX and OX. One big difference between EHBSA/WT and PMX/OX is that EHBSA/WT requires a smaller population size to work than PMX/OX.

Results on 30-job and 10-machine problem are shown in Table 2. Performances of EHBSA/WT is much better than EHBSA/WO. Comparing the performance of EHBSA/WT with other operators, EHBSA/WT is almost the same with PMX and OX again showing that EHBSA/WT requires a smaller population size to work than PMX/OX.

From the results described above, we can see that EHBSA/WT/ n works fairly well in flow shop problems used developed in this paper. It has almost the same performance with popular traditional two-parent recombination operators, OX and PMX, and has much better performance than eER. One interesting feature of EHBSA/WT/ n is that it requires smaller population size than traditional two parent recombination operators. This may be an important property of EHBSA/WT/ n . In our experiments, we used a blind search. When we combine EHBSA/WT/ n with some heuristics, it would work well with a smaller population size. As to the number of cut points n , n values of 3, 4, and 5 worked well in these problems.

4. CONCLUSIONS

In this paper, we have proposed a probabilistic model-building genetic algorithms (PMBGAs) in permutation representation domain using the flow shop problem, a typical, well-known optimization problem in permutation domain and compare its performance with traditional recombination operators. In this approach, we developed an edge histogram model from the current population.

Three types of sampling algorithms, EHBSA/WO1, EHBSA/WO2, and EHBSA/WT were presented. The results showed EHBSA/WT worked fairly well with a smaller size of population on the test problems used. It also worked better than well-known traditional two parent recombination operators. Thus, we can confirm that the EHBSA works well also on the flow shop problems which need an asymmetrical edge histogram.

There are many opportunities for further research related to the proposed algorithms. The effect of parameter values of B_{ratio} , number of cut point of the template n , and size of population N , on the performance of the algorithm must be further investigated. We experimented with EHBSAs using a blind search to test the pure mixing capability of the proposed algorithms. But we must

Table 1. Results on 20-job and 10-machine problem

Model	Population Size N									
	60		120		240		480		960	
	Best	Aver	Best	Aver	Best	Aver	Best	Aver	Best	Aver
EHBSA/WO1	1612.0	1635.0	1606.0	1633.1	1622.0	1634.3	/	/	/	/
EHBSA/WO2	1611.0	1623.8	1611.0	1623.5	1596.0	1627.6				
EHBSA/WT/2	1520.0	1523.5	1523.0	1529.5	1521.0	1530.6				
EHBSA/WT/3	1521.0	1524.5	1520.0	1526.7	1523.0	1532.5				
EHBSA/WT/4	1521.0	1525.7	1524.0	1528.8	1523.0	1529.9				
EHBSA/WT/5	1519.0	1528.8	1521.0	1530.4	1525.0	1534.1				
OX	1539.0	1559.2	1523.0	1545.5	1527.0	1536.6	1520.0	1526.9	1523.0	1527.8
eER	1575.0	1593.3	1589.0	1597.8	1594.0	1602.3	1589.0	1604.5	1602.0	1617.5
PMX	1526.0	1562.0	1528.0	1540.8	1523.0	1532.4	1523.0	1530.2	1520.0	1522.4

Table 2. Results on 30-job and 10-machine problem

Model	Population Size N									
	60		120		240		480		960	
	Best	Aver	Best	Aver	Best	Aver	Best	Aver	Best	Aver
EHBSA/WO1	2243.0	2272.0	2240.0	2275.4	2251.0	2285.3	/	/	/	/
EHBSA/WO2	2257.0	2278.3	2262.0	2282.4	2259.0	2286.5				
EHBSA/WT/2	2097.0	2113.8	2117.0	2124.4	2117.0	2136.6				
EHBSA/WT/3	2089.0	2110.8	2104.0	2116.8	2117.0	2125.9				
EHBSA/WT/4	2096.0	2110.7	2111.0	2117.4	2115.0	2123.5				
EHBSA/WT/5	2089.0	2109.7	2103.0	2116.8	2108.0	2124.3				
OX	2129.0	2150.1	2111.0	2123.2	2095.0	2117.1	2087.0	2108.7	2113.0	2131.9
eER	2194.0	2225.7	2203.0	2227.1	2208.0	2233.8	2217.0	2245.4	2226.0	2256.0
PMX	2109.0	2139.3	2112.0	2118.6	2100.0	2114.0	2087.0	2098.0	2100.0	2114.2

test the algorithms with appropriate heuristics in problems with large numbers of cities.

The authors gratefully acknowledge Prof. David E. Goldberg and Dr. Martin Pelikan for their valuable comments on PMBGAs during my stay at IlliGAL in 2001. This research is partially supported by the Ministry of Education, Culture, Sports, Science and Technology of Japan under Grant-in-Aid for Scientific Research number 13680469, and a grant to RCAST at Doshisha University from Ministry of Education, Culture, Sports, Science and Technology of Japan.

5. REFERENCES

[1] Baluja, S.: Population-based incremental learning: A method for interacting genetic search based function optimization and coemptive learning, Tech. Rep. No. CMU-CS-94-163, Carnegie Mellon University (1994).

2] Baluja, S. and Davies: Using optimum dependency-trees for combinatorial optimization: learning the structure of the search

space, Tech. Rep. No. CMU-CS-97-107, Carnegie Mellon University (1997)

[3] Bosman, P. and Thierens, D.: An algorithmic framework for density estimation based evolutionary algorithms, Tech. Rep. No. UU-CS-1999-46, Utrecht University (1999).

[4] Bosman, P. and Thierens, D.: Continuous iterated density estimation evolutionary algorithms within the IDEA framework, Proc. of the Optimization by Building and Using Probabilistic Models OBUPM Workshop at the Genetic and Evolutionary Computation Conference GECCO-2000, pp.197-200 (2000).

[5] Bosman, P. and Thierens, D.: Mixed IDEAs, Tech. Rep. No. UU-CS-2000-45, Utrecht University (2000).

[6] De Bonet, J. S., Isbell, C. L. and Viola, P.: MIMIC: Finding optima by estimating probability densities, In Mozer, M. C., Jordan, M. I., and Petsche, T. (Eds): Advances in neural information processing systems, Vol. 9, pp. 424-431. (1997).

- [7] Goldberg, D. E.: Genetic algorithms in search, optimization and machine learning, Addison-Wesley publishing company (1989).
- [8] Harik, G., Lobo, F. G., and Goldberg, D. E.: The compact genetic algorithm, Proc. of the Int. Conf. Evolutionary Computation 1998 (ICEC 98), pp. 523-528 (1998).
- [9] Harik, G: Linkage learning via probabilistic modeling in the ECGA, Tecnical Report IlliGALReport 99010, University of Illinois at Urbana-Champaign, Urbana, Illinois (1999).
- [10] Larranaga, P., Etxeberria, R., Lozano, J.A., and Pena, J.M.: Optimization by learning and simulation of Bayesian and gaussian networks, University of the Basque Country Technical Report EHU-KZAAIK -4/99 (1999).
- [11] Mhlenbein, H and Paa, G.: From recombination of genes to the estimation of distribution I. Binary parameters, Proc. of the Parallel Problem Solving from Nature - PPSN IV, pp. 178-187 (1996).
- [12] Pelikan, M., Goldberg, D. E., and Cantu-Paz, E.: BOA: The Bayesian optimization algorithm, Proc. of the Genetic and Evolutionary Computation Conference 1999 (GECCO-99), Morgan Kaufmann, San Francisco, CA (1999).
- [13] Pelikan, M., Goldberg, D. E., and Lobo, F. G.: A survey of optimization by building and using probabilistic models, Technical Report IlliGAL Report 99018, University of Illinois at Urbana-Champaign (1999).
- [14] Pelikan, M., Goldberg, D. E., and Cantu-Paz, E.: Linkage problems, distribution estimate, and Bayesian network, Evolutionary Computation, Vol. 8, No. 3, pp. 311-340 (2000).
- [15] Sebag, M. and Ducoulombier, A.: Extending population-based incremental learning to continuous search spaces, Proc. of the Parallel Problem Solving from Nature - PPSN V, pp. 418-427 (1998).
- [16] Servet, I. L., Trave-Massuyes, L., and Stern, D.: Telephone network traffic overloading diagnosis and evolutionary computation techniques, Proc. of the Third European Conference on Artificial Evolution (AE 97), pp. 137-144 (1997).
- [17] Larranaga, P., Etxeberria, R., Lozano, J. A., and Pena, J. M.: Optimization in continuous domains by learning and simulation of Gaussian networks, Proc. of the 2000 Genetic and Evolutionary Computation Conference Workshop Program, pp. 201-204 (2000).
- [18] Robles, V., Miguel, P. D., and Larranaga, P.: Solving the traveling salesman problem with EDAs, Estimation of Distribution Algorithms, Larranaga, P. and Lozano, J. A. (eds), Kluwer Academic Publishers, Chapter 10, pp. 211-229 (2002).
- [19] Larranaga, P. and Lozano, J. A. (eds): Estimation of distribution algorithms, Kluwer Academic Publishers (2002).
- [20] Tsutsui, S., Pelikan, M., and Goldberg, D. E.: Evolutionary Algorithm using Marginal Histogram Models in Continuous Domain, Proc. of the 2001 Genetic and Evolutionary Computation Conference Workshop Program, pp. 230-233 (2001).
- [21] Tsutsui, S.: Probabilistic Model-Building Genetic Algorithms in Permutation Representation Domain Using Edge Histogram, Proc. of the 7th Parallel Problem Solving from Nature - PPSN VII (2002, to appear).
- [22] Johnson, D. S, and McGeoch, L. A.: Experimental analysis of heuristics for the STSP, The Traveling Salesman Problem and its Variations, Gutin and Punnen (eds), Kluwer Academic Publishers, Chapter 1 (to appear).
- [23] Oliver, I, Smith, D., and Holland, J.: A study of permutation crossover operators on the travel salesman problem, Proc. 2nd Int. Conf. on Genetic Algorithms (1987).
- [24] Starkweather, T., McDaniel, S., Mathias, K, Whitley, D, and Whitley, C.: A comparison of genetic sequence operators, Proc of the 4th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, pp. 69-76 (1991).
- [25] Whitley, D., Starkweather, T., and Fuquay, D.: Scheduling problems and traveling salesman problem: The genetic edge recombination operator, Proc. 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann (1989).