Using Edge Histogram Models to Solve Permutation Problems with Probabilistic Model-Building Genetic Algorithms

Shigeyoshi Tsutsui
Martin Pelikan
David E. Goldberg

IlliGAL Report No. 2003022
August 2003

Illinois Genetic Algorithms Laboratory
University of Illinois at Urbana-Champaign
117 Transportation Building
104 S. Mathews Avenue
Urbana, Illinois 61801 USA
Phone: 217-333-0897
Fax: 217-244-5705
Web: http//www-illigal.ge.uiuc.edu

Using Edge Histogram Models to Solve Permutation Problems with Probabilistic Model-Building Genetic Algorithms

Shigeyoshi Tsutsui
Dept. of Management and Information Science, Hannan University
5-4-33 Amamihigashi, Matsubara, Osaka 580-5802, Japan
tsutsui@hannan-u.ac.jp

Martin Pelikan
Dept. of Math and Computer Science, University of Missouri at St. Louis
8001 Natural Bridge Rd., St. Louis, MO 63121
pelikan@illigal.ge.uiuc.edu

David E. Goldberg
Illinois Genetic Algorithms Laboratory and Dept. of General Engineering,
University of Illinois at Urbana-Champaign
117 Transportation Building, 104 S. Mathews Ave., Urbana, IL 61801
deg@illigal.ge.uiuc.edu

Abstract

Recently, there has been a growing interest in probabilistic model-building genetic algorithms (PMBGAs), which replace traditional variation operators of genetic and evolutionary algorithms by building and sampling a probabilistic model of promising solutions. In this paper we propose a PMBGA that uses *edge histogram based sampling algorithms (EHBSAs)* to solve problems with candidate solutions represented by permutations. Two sampling algorithms—the sampling without template (EHBSA/WO) and the sampling with template (EHBSA/WT)—are presented. The proposed algorithms are tested on several instances of the traveling salesman problem (TSP). The results show that EHBSA/WT works fairly well even with a small population size on all tested problem instances and that it outperforms popular two-parent recombination operators for permutations and other PMBGAs for permutation problems. Combining EHBSA with a simple local heuristic for solving TSP called 2-OPT improves the performance of the algorithm, enabling efficient solution to problems of hundreds of cities. Nonetheless, unlike most other TSP solvers, EHBSA is not limited to solving TSP instances, but it can be applied to any problem defined on permutations.

Key words: Probabilistic model-building genetic algorithms (PMBGAs), estimation of distribution algorithms (EDAs), permutation problems, edge histogram matrix, traveling salesman problem.

1 Introduction

Genetic Algorithms (GAs) (Holland 1975; Goldberg 1989) are widely used as robust black-box optimization techniques applicable across a broad range of real-world problems, including parameter optimization, scheduling, design, and combinatorial optimization. GAs evolve a population of candidate solutions (individuals) to the given problem starting with a randomly generated population. Each iteration starts by creating a population of promising individuals using a selection operator, which favors high-quality solutions by making more copies of better solutions at the expense of the worse ones. New candidate solutions are generated by applying recombination and mutation operators to the selected population of solutions. Recombination creates new candidate solutions by combining bits and pieces of promising solutions, whereas mutation creates new solutions by perturbing selected solutions slightly. Since both recombination and mutation introduce variation into the population of selected solutions, these operators are often called variation operators.

GAs should work well for problems that can be decomposed into sub-problems of bounded difficulty (Goldberg, 2002) and for problems where similar solutions are of similar quality. However, fixed, problem-independent variation operators are often incapable of effective exploitation of the selected population of high-quality solutions and the search for the optimum often becomes intractable (Thierens, 1995; Goldberg, 2002; Pelikan, 2002). One of the most promising research directions that focus on eliminating this drawback of fixed, problem-independent variation operators, is to look at the generation of new candidate solutions as a learning problem, and use a probabilistic model of selected solutions to generate the new ones (Pelikan, Goldberg, & Lobo, 2002; Pelikan, 2002; Larrañaga & Lozano, 2002). The probabilistic model is expected to reflect the problem structure and, as a result, this approach might provide more effective exploitation of promising solutions than recombination and mutation operators in traditional GAs. The algorithms based on learning and sampling a probabilistic model of promising solutions to generate new candidate solutions are called *probabilistic model-building genetic algorithms (PMBGAs)* (Pelikan, Goldberg, & Lobo, 2002), *estimation of distribution algorithms (EDAs)* (Muehlenbein & Paass, 1996), or *iterated density estimation algorithms (IDEAs)* (Bosman & Thierens, 2000).

Most work on PMBGAs focuses on optimization problems where candidate solutions are represented by fixed-length vectors of discrete or continuous variables. However, for many combinatorial problems permutations provide a much more natural representation for candidate solutions. Despite the great success of PMBGAs in the domain of fixed-length discrete and continuous vectors, only few studies can be found on PMBGAs for permutation and scheduling problems permutations (Bosman & Thierens, 2001, 2002; Robles et

al., 2002), and even these studies take an indirect approach of mapping permutation problems to fixed-length vectors of discrete or continuous variables, what in some cases necessitates the use of repair operators to correct invalid permutations.

This paper introduces a promising approach to learning and sampling probabilistic models for permutation problems using *edge histogram models*. Edge histogram models can be used to *directly* encode and sample probability distributions over permutations without requiring a transformation of permutations to another domain or a repair operator. The results presented here indicate that using edge histogram models within the PMBGA framework provides competitive results on several benchmark instances of the traveling salesman problem (TSP). Nonetheless, the methods proposed here are not limited to TSP like most other TSP solvers and specialized variation operators are. As a result, this work provides a promising direction for solution of any problem that can be formulated within the domain of fixed-length permutations; flow shop scheduling is an example of such a problem as described in Tsutsui & Miki, 2002.

The paper starts with a brief description of PMBGAs with the focus on PMBGAs for permutation problems. Section 3 describes the two proposed EHBSAs. Section 4 provides the empirical analysis of EHBSAs on several benchmarks of TSP. Section 5 shows how EHBSA can be combined with local search to improve its performance on TSP, and presents empirical results of the hybrid method that combines EHBSA with a simple local TSP heuristic called 2-OPT. Finally, Section 5 summarizes and concludes the paper.

2 Probabilistic Model-Building Genetic Algorithms

Probabilistic model-building genetic algorithms (PMBGAs) evolve a population of candidate solutions to the given problem by building and sampling a probabilistic model of promising solutions. PMBGAs start with a random population of candidate solutions (individuals). Each iteration of PMBGAs starts by selecting better individuals from the current population. Next, the probability distribution of the selected population of individuals is estimated. New individuals are then generated according to this estimate, forming the population of candidate solutions for the next generation. The process is repeated until the termination conditions are satisfied.

As mentioned above, most work on PMBGAs focuses on parameter optimization for problems defined over fixed-length vectors of discrete or real-valued variables (see Pelikan (2002), Pelikan et al. (2002), and Larrañaga & Lozano (2002) for an overview of these methods), but only few studies can be found that focus on optimization of problems with solutions represented by permutations. Bosman & Thierens (2001, 2002) use random keys to represent permutations together with a marginal product factorization to estimate the continuous distribution of random keys. Robles, Miguel, and Larrañaga (2002) also use random keys to enable the use

of PMBGAs from continuous parameter optimization in solving permutation problems. Additionally, they use discrete PMBGAs in combination with the all-time-modification technique for repairing disrupted permutations.

The next section describes edge histogram models that can be used to model and sample permutations within the PMBGA framework. Subsequently, the proposed methods are tested on a number of instances of the traveling salesman problem (TSP) with and without an additional heuristic 2-OPT.

3 Edge Histogram Based Sampling Algorithm (EHBSA)

This section describes the edge histogram based sampling algorithm (EHBSA) and its use within the PMBGA framework for (1) modeling promising solutions and (2) generating new solutions by simulating the learned model.

3.1 Basic Description of the Algorithm

An *edge* is a link or connection between two nodes. A set of edges that creates a path in the graph that includes all nodes can be used as an alternative representation of a permutation; the first node of the path will be the first element of the permutation, while the remaining elements in the permutation can be obtained by following the remainder of the path until the last node of the path is reached. Some crossover operators, such as Edge Recombination (ER) (Whitley, Starkweather, & Fuquay, 1989) and enhanced ER (eER) (Starkweather et al., 1991)—which can both be used in traditional two-parent recombination of conventional GAs—use the edge-based representation of permutations for only the two parents strings. The basic idea of our approach, *the edge histogram based sampling algorithm (EHBSA),* is to collect and exploit information about the edges globally across the entire population of high-quality permutations from the selected set of solutions.

EHBSA starts by generating a random permutation for each individual population of candidate solutions. Promising solutions (permutations) are then selected using any popular selection scheme based on their quality computed using the problem-specific evaluation function that defines the problem. Next, an *edge histogram matrix (EHM)* for the selected solutions is constructed (see Section 3.2). New solutions are then generated by sampling the edge histogram matrix (see Section 3.3). The new solutions replace some of the old ones and the process is repeated until the termination criteria are met. EHBSA can thus be seen as a permutation version of the PMBGA based marginal histogram models proposed by Tsutsui, Pelikan, & Goldberg (2001).

3.2 Learning Edge Histogram Matrix

Let the permutation represented by the kth individual in population $P(t)$ at generation t be denoted by

$s^t_k = (\pi^t_k(0), \quad \pi^t_k(1), \ldots, \pi^t_k(L\text{-}1))$, where $(\pi^t_k(0), \pi^t_k(1), \ldots,$ and $\pi^t_k(L\text{-}1))$ define a permutation of (0, 1, ..., L-1), and L is the length of the permutation. The edge histogram matrix EHM^t $(e^t_{i,j})$ $(i, j = 0,1, .., L\text{-}1)$ of the population $P(t)$ is symmetrical and consists of L^2 elements:

$$e^t_{i,j} = \begin{cases} \sum_{k=1}^{N} (\delta_{i,j}(s^t_k) + \delta_{j,i}(s^t_k)) + \varepsilon & \text{if } i \neq j \\ 0 & \text{if } i = j \end{cases}, \tag{1}$$

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

$$\delta_{i,j}(s^t_k) = \begin{cases} 1 & \text{if } \exists h\, [h \in \{0,1,\cdots L-1\} \wedge \pi^t_k(h) = i \wedge \pi^t_k((h+1) \bmod L) = j] \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

and $\varepsilon > 0$ biases the sampling toward random permutations and it can thus be seen as a form of mutation. To use a comparable pressure toward random permutations for all problems and parameter settings, ε should be proportional to the expected value of $e^t_{i,j}$. Since the average of elements $e^t_{i,j}$ for $i \neq j$ in EHM^t is $2LN/(L^2\text{-}L) = 2N/(L\text{–}1)$, we get

$$\varepsilon = \frac{2N}{L-1} B_{\text{ratio}} \tag{3}$$

where B_{ratio} ($B_{\text{ratio}} > 0$) or the bias ratio is a constant related to the pressure toward random permutations. A smaller value of B_{ratio} reflects the real distribution of edges in the parent population, whereas a bigger value of B_{ratio} will allow infrequent addition of new edges (see Section 3.3). An example of EHM^t is shown in Figure 1.

$$\begin{array}{l} s^t_1 = (\,0,\ 1,\ 2,\ 3,\ 4\,) \\ s^t_2 = (\,1,\ 3,\ 4,\ 2,\ 0\,) \\ s^t_3 = (\,3,\ 4,\ 2,\ 1,\ 0\,) \\ s^t_4 = (\,4,\ 0,\ 3,\ 1,\ 2\,) \\ s^t_5 = (\,2,\ 1,\ 3,\ 4,\ 0\,) \end{array} \qquad \begin{pmatrix} 0 & 3.1 & 2.1 & 2.1 & 3.1 \\ 3.1 & 0 & 4.1 & 3.1 & 0.1 \\ 2.1 & 4.1 & 0 & 1.1 & 3.1 \\ 2.1 & 3.1 & 1.1 & 0 & 4.1 \\ 3.1 & 0.1 & 3.1 & 4.1 & 0 \end{pmatrix}$$

(a) $P(t)$ (b) EHM^t

Figure 1. An example of a symmetric edge histogram matrix for $N = 5$, $L = 5$, $B_{\text{ratio}} = 0.04$

Although the edge histogram matrix defined above is *symmetric* (i.e., $e_{i,j} = e_{j,i}$) and it is thus applicable only to problems where the orientation of edges does not matter (e.g., to symmetric TSP instances), an *asymmetric* edge histogram matrix can be defined for problems where the orientation of edges does matter (e.g., for asymmetric TSP and flow shop scheduling) An asymmetric EHM^t can be defined analogically to the symmetric matrix EHM^t defined Eq. 1, but only edges from i to j will be counted in computing e^t_{ij}.

Time complexity of learning the edge histogram matrix is lower-bounded by a product of the number N of individuals used to learn the matrix and the permutation length L. Time complexity of the learning is also

lower-bounded by the number of elements in EHM^t. In the steady-state evolutionary model used in this paper, the initial computation of the edge histogram matrix (EHM^0) takes $O(N L + L^2)$ time steps, whereas the complexity of updating the edge-histogram model in subsequent iterations can be bounded by $O(L)$. In the generational evolutionary model, learning the edge histogram matrix in each generation would take $O(N L)$ steps.

The following subsection describes how EHM^t can be used to sample new solutions using two sampling procedures.

3.3 Sampling Algorithms

In this subsection, we describe two sampling algorithms for sampling the edge histogram matrix EHM^t: (1) the edge histogram based sampling algorithm without template (EHBSA/WO), and (2) the edge histogram based sampling algorithm with template (EHBSA/WT).

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

Let us denote the elements of the permutation to be sampled by $c[i]$ for $i \in \{0, 1, \ldots, L\text{-}1\}$. EHBSA/WO starts by randomly selecting the initial element of the new permutation (denoted by $c[0]$). The sampling continues recursively using a variant of the roulette-wheel selection (Goldberg, 1989). Let us assume that the last element of the new permutation that we have generated is $c[i]$ (that means that we have generated i+1 elements so far). The new element c[i+1] is set to j (we restrict potential values of j so that $j \neq c[k]$ for all $k \in \{0, 1, \ldots, i\}$; otherwise, the new edge would create a cycle) with a probability proportional to the element $e^t_{c[0],j}$ of the given edge histogram matrix EHM^t. The sampling continues until the entire permutation has been generated. Figure 2 shows the schematic description of the edge histogram based sampling algorithm without template (EHBSA/WO).

1. Set the position counter $p \leftarrow 0$.
2. Obtain first node $c[0]$ randomly from $[0, L\text{–}1]$.
3. Construct a roulette wheel vector $rw[]$ from EHM^t as $rw[j] \leftarrow e^t_{c[p],j}$ (j=0, 1, .., L–1).
4. Set to 0 previously sampled nodes in $rw[]$ ($rw[\ c[i]\ ] \leftarrow 0$ for i =0, .., p).
5. Sample next node $c[p+1]$ with probability $rw[x] / \sum_{j=0}^{L-1} rw[j]$ using roulette wheel $rw[]$.
6. Update the position counter $p \leftarrow p+1$.
7. If $p<L\text{–}1$, go to Step 3.
8. Obtain a new individual string $c[]$.

Figure 2. Edge histogram based sampling algorithm without template (EHBSA/WO)

The EHBSA/WO sampling described above is only applicable to problems where the absolute position of each node in a permutation string does not matter (for example, TSP). Slight modifications are required to apply EHBSA/WO to problems where the absolute position of each node in a permutation string does matter (for example, scheduling). Note that EHBSA/WO is similar to the sampling in Ant Colony Optimization (ACO) (Dorigo, Maniezzo, & Colorni, 1996). The most apparent difference between EHBSA and ACO is in the way the algorithms bias the search toward high-quality solutions. Like most other evolutionary algorithms, EHBSA uses a selection operator to select promising solutions, which are in turn used to create a model for generating new candidate solutions. On the other hand, in ACO the generation of new solutions is guided by the intensity of pheromone trails in the search space, and ACO thus favors high-quality solutions by specifying an update rule for pheromone trails that ensures that the areas of the search space that are likely to provide better solutions contain pheromone trails of higher intensity (this is ensured by developing a specific update rule for each class of problems). An important feature of EHBSA is that EHBSA does not need a new update rule for a new problem, but it can approach any problem defined on permutations directly without any modifications. For a more detailed discussion on the relationship between EHBSA and ACO, please see Tsutsui (2003), who also describes an approach to use some of the techniques used in EHBSA to improve ACO.

Time complexity of sampling one permutation from the edge histogram matrix can be bounded by $O(L^2)$.

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

EHM^t described in Section 3.2 does not consider interactions between edges, which can make the sampling rather ineffective in practice. EHBSA/WT attempts to improve the sampling by using a template. EHBSA/WT starts by selecting a *template individual* from the selected population $P(t)$. We use a simple strategy for selecting the template individual and set the template to a random individual from $P(t)$ (note that all individuals in $P(t)$ have gone through the process of selection, so their quality can be assumed to be relatively high in comparison with that of a fully random individual). A new individual is created by copying a part of the template directly and generating the remaining positions according to the edge histogram matrix learned from the population of promising solutions. Other strategies could be used for selecting templates; for example, the best individual in the current population could be used as a template. It is advantageous, however, to use multiple templates in generating the new population of permutations, because using a single template for all new individuals would make the effects of recombination more local.

A crucial question when using a template to sample new permutations is how to determine the positions of permutation elements or nodes that are to be copied from the template and the positions of nodes that are going to be generated anew. To ensure robustness across a wide spectrum of problems, it should be advantageous

to introduce variation both in the positions of permutation elements that are to be generated anew as well as in the number of these elements.

To choose positions to generate anew, we use a simple method inspired by n-point crossover of simple GAs. In EHBSA/WT/n, the template permutation is first mapped onto a circle so that the last position is followed by the first one. To generate each new individual, n>1 positions in the template are selected at random, dividing the template into n segments of variable length. The segment the elements of which are to be generated anew is chosen randomly out of these n segments; all remaining elements are copied directly from the template. Randomness in generating cut positions and choosing one of these segments introduces variation in segment positions and lengths of elements that are going to be generated anew. The probability distribution of segment lengths can be controlled by n; for n=2, segment lengths are distributed uniformly, but as n grows, shorter segment lengths become dominant (see Section 3.4 for a more detailed discussion of this issue). Again, other methods can be used to select positions that are to be generated anew.

Figure 3 shows an example of EHBSA/WT/3 where 3 cut points are used. In this example, three segments, segment1, segment2 and segment3 are generated by cut points cut[0], cut[1] and cut[2], and segment1 is chosen for sampling nodes. Nodes of the new string in segment0 and segment2 (before cut[1] and after cut[2]) are the same as the nodes of the template. New nodes are sampled for only segment1 (from cut[1] up to, but not including, cut[2]) based on the EHM^t using a similar algorithm to that in EHBSA/WO.

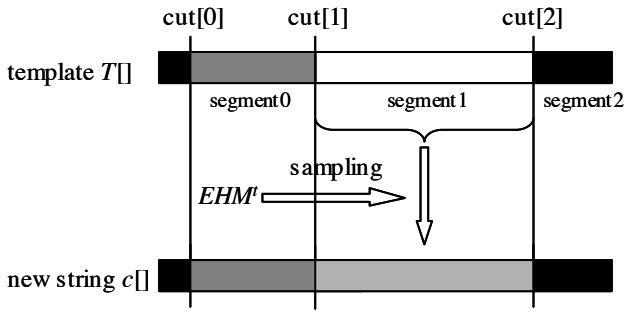

Figure 3. An example of EHBSA/WT/3

Figure 4 shows a schematic description of the edge histogram based sampling algorithm with template and n cut points (EHBSA/WT/n).

1. Choose a template $T[]$ from $P(t)$.
2. Obtain sorted cut point array cut[0], cut[1], .., cut[n–1] randomly.
3. Choose a cut point cut[l] by generating random number $l \in [0, n-1]$.
4. Copy nodes in $T[]$ to $c[]$ from after cut[(l+1) mod n] and before cut[l].
5. Set the position counter $p \leftarrow (\text{cut}[l] - 1 + L) \bmod L$.
6. Construct a roulette wheel vector $rw[]$ from EHM^t as $rw[j] \leftarrow e^t_{c[p],j}$ (j=0, 1, .., L–1).
7. Set to 0 copied and previously sampled nodes in $rw[]$ ($rw[c[i]] \leftarrow 0$ for i = cut[(l+1) mod n], .., 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 $(p+1) \bmod L \neq \text{cut}[(l+1) \bmod n]$, go to Step 6.
11. Obtain a new individual string $c[]$.

Figure 4. Edge histogram based sampling algorithm with template (EHBSA/WT)

Time complexity of EHBSA/WT with respect to the number of elements in a permutation must be at least O(L), but at most O(L^2). Where the actual complexity lies depends on the chosen distribution of segments to generate anew; more specifically, the actual complexity depends on the length of the segments that are to be generated anew. There are two effects of modifying the length of the segments to generate. On one hand, shortening the segments increases the speed of sampling. On the other hand, shortening the segments also decreases the rate at which the sampling explores new regions of the search space. Since in EHBSA/WT/n, the average length of segments that are resampled decreases with the number n of cut-points (see Subsection 3.4), the time complexity of sampling a new solution also decreases with n. More specifically, the time complexity is proportional to L^2/n.

The following section describes experimental methodology and results on several benchmark instances of TSP. The section is followed by a discussion on hybridization of EHBSA, and experimental results of applying a hybrid consisting of EHBSA and 2-OPT to several larger TSP instances.

3.4 Distribution of Segment Lengths using EHBSA/WT/*n*

Above we have mentioned that the distribution of segment lengths to be generated anew depends on the number of cut points. This section looks at the dependency between n and the distribution of segment lengths in somewhat more detail.

The string length L and each cut point are integer values. But for simplicity, here we assume L and each cut point cut[l] (l = 0, 1, …, n−1) to take continuous values. There is an edge between the last node and the first node in a string. Thus, we represent a string as a continuous circle of length L as shown in Figure 5. Then, the length of a segment can be calculated as a continuous value. Without loss of generality, we fix the cut point cut[0] to be located at the origin, and consider the segment between cut point cut[0] and cut point cut[1] as a

typical random segment, where cut point cut[1] is defined as the nearest cut point in clockwise rotation to cut point cut[0] (see Figure 4). Let $F(x)$ and $f(x)$ be the probability distribution function and the probability density function of the length of the segment described above, respectively. Let $F'(x)$ be the probability that all cut points cut[l] ($l = 1, 2, \ldots, n-1$), excluding cut[0], are greater than x. The probability that a cut point cut[l] ($l \neq 0$) is greater than x is $1-x/L$. So, $F'(x)$ is obtained as

$$F'(x) = \left(1 - \frac{x}{L}\right)^{n-1}. \tag{4}$$

Then, $F(x)$ can be obtained from $F'(x)$ as

$$\begin{aligned} F(x) &= 1 - F'(x) \\ &= 1 - \left(1 - \frac{x}{L}\right)^{n-1}. \end{aligned} \tag{5}$$

The density function $f(x)$ is obtained from $F(x)$ as

$$\begin{aligned} f(x) &= \frac{d}{dx} F(x) \\ &= \frac{(n-1)}{L}\left(1 - \frac{x}{L}\right)^{n-2}. \end{aligned} \tag{6}$$

The average of x is obtained as

$$\bar{x} = \int_0^L x \cdot f(x)dx = \frac{L}{n}. \tag{7}$$

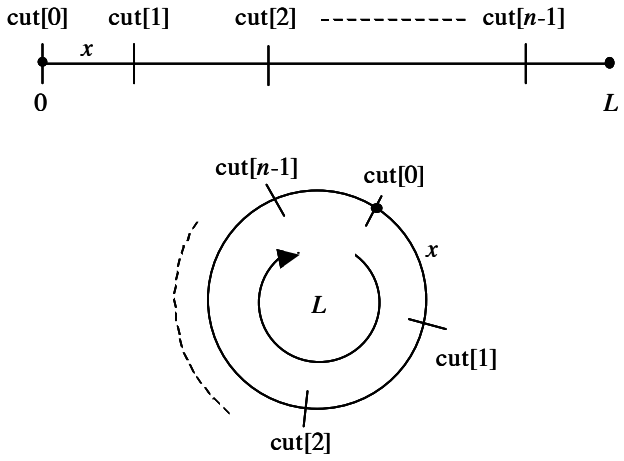

Figure 5. Calculation of length a segment formed by n cut points

Figure 6 shows the probability density function $f(x)$ of segment lengths for EHBSA/WT/n for n=2,3, and 4. For $n = 2$, $f(x)$ is uniformly distributed on $[0, L]$. Thus, the length of a segment to be sampled in EHBSA/WT/2 is uniformly distributed on $[0, L]$. When the length of the segment is small, the EHBSA/WT/2 samples a small number of nodes, performing a kind of local search improvement over the template individual. On the other hand, when the length is large, the EHBSA/WT/2 samples a large number of nodes, performing a larger change to the template. Due to the uniform distribution of segment lengths on $[0, L]$, EHBSA/WT/2 balances global and local improvements.

For $n > 2$, short segments are more likely to occur. As a result, a smaller value of n should work well for problems with a smaller numbers of cities, and a larger number of n work well with problems with a larger numbers of cities; i.e., gr24: $n = 2$, gr48: $n = 3$, and pr76: $n = 5$.

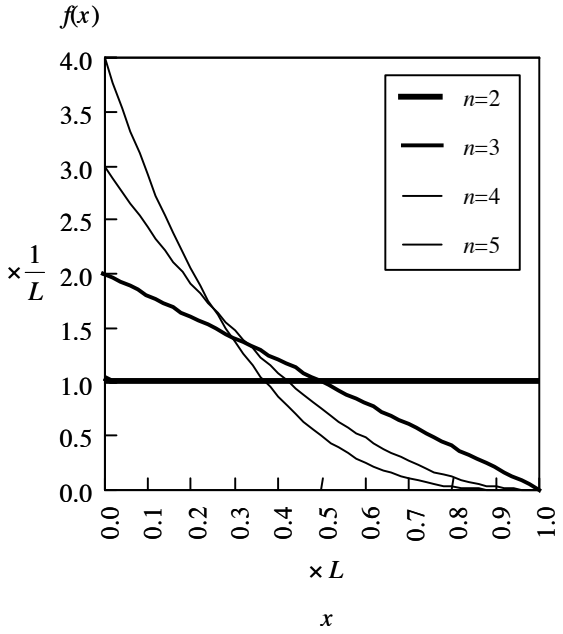

Figure 6. Probability density function $f(x)$ of segment lengths for EHBSA/WT/n

4 Empirical Study

This section applies EHBSA/WO and EHBSA/WT to several benchmark instances of the traveling salesman problem (TSP), which is one of the most popular permutation problems. However, note that the EHBSA is not limited to TSP, because it uses no specific features of TSP; EHBSA is not given the locations of cities in TSP, it does not assume that there is a proper metric defined to compute the distance between the pairs of cities, and it does not assume that the problem can be mapped to an instance of TSP. This important feature distinguishes EHBSA from most other methods that can be used to solve TSP.

4.1 Experimental Methodology

4.1.1 Evolutionary models

This section describes evolutionary models for all methods included in our comparison, i.e. EHBSA/WT, EHBSA/WO, and two-parent recombination operators. All models are based on the steady-state scheme.

(1) ***Evolutionary model for EHBSA/WT***

Let the population size be N, and let it, at time t, be denoted by $P(t)$. The population $P(t+1)$ is produced as follows (see Figure 7):

1. Edge histogram matrix EHM^t (see Subsection 3.2) is computed from $P(t)$
2. A template individual $T[]$ is selected from $P(t)$ randomly.
3. EHBSA/WT (see Subsection 3.3.2) is executed using EHM^t and $T[]$ to generate a new individual $c[]$.
3. The new individual $c[]$ is evaluated.
4. If $c[]$ is better than $T[]$, then $T[]$ is replaced with $c[]$, otherwise the population remains unchanged, forming $P(t+1)$.

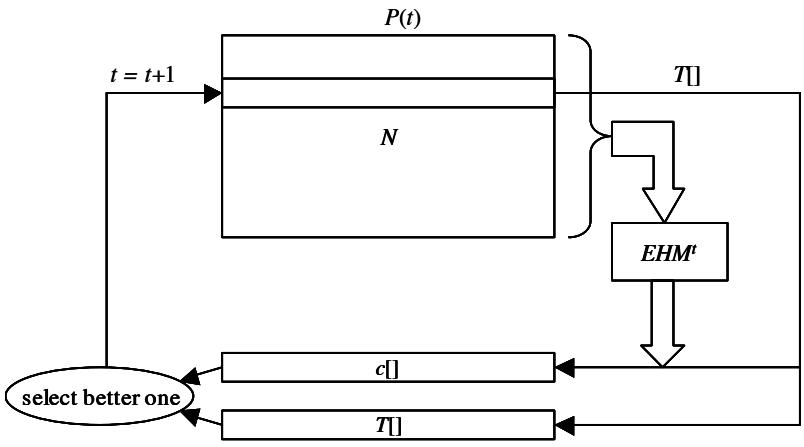

Figure 7. Evolutionary model for EHBSA/WT

(2) ***Evolutionary model for EHBSA/WO***

The evolutionary model for EHBSA/WO is similar to the model for EHBSA/WT, except that EHBSA/WO does not use a template $T[]$. The new string $c[]$ is compared to a randomly selected individual $i[]$ in $P(t)$, and *if* $c[]$ is better than $i[]$, $i[]$ is replaced with $c[]$; otherwise, the population remains unchanged.

(3) ***Evolutionary model for two-parent recombination operators***

To compare the performance of the proposed methods with the performance of traditional two-parent recombination operators, we used the same steady-state scheme for two-parent recombination operators as well and after selecting and recombining two parents, one of the two new offspring is incorporated into the original population. This scheme was previously used in the GENITOR algorithm (Whitley, 1989). 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. The selection pressure thus comes into play via replacement, similarly as in the deterministic crowding (Mahfoud, 1992) and restricted tournament selection (RTS) (Harik, 1995).

4.1.2 Test suite and performance measures

We have tested EHBSA on the Traveling Salesman Problem (TSP), a typical, well-known optimization problem in permutation representation domain. The following benchmark instances of TSP have been used in our empirical study: 24 cities gr24, 48 cities gr48, and 76 cities pr76. The gr24 and gr48 are used in the study of TSP with EDA in Robles et al. (2002). We have compared EHBSA with popular order-based recombination operators, namely, the order crossover OX (Oliver, Smith, & Holland, 1987), the enhanced edge recombination operator eER (Starkweather et al., 1991), and the partially mapped crossover (Goldberg, 1989). We have also compared EHBSA with the results presented in Robles et al. (2002) on gr24 and gr48. Our study focused on the effects of recombination; that is why we did not include mutation or other local operators in our experiments. Incorporating local search is discussed in the next section, where a hybrid method is described that incorporates a popular local heuristic 2-OPT into EHBSA and other compared algorithms to improve solutions locally. Using the hybrid method is shown to improve the efficiency of the search rapidly.

Note that the comparisons presented in this paper are only illustrative, and are presented to indicate that the proposed PMBGA with EHBSA sampling might be a promising approach to solving challenging permutation problems and that the problem instances tested here are not trivial; however, an in-depth comparison with state-of-the-art methods for solving TSP must be done to determine the advantages and disadvantages of using EHBSA compared to other promising approaches for solving this class of problems. Furthermore, the application of EHBSA is not limited to TSP, because EHBSA does not exploit specific properties of TSP, such as the geographical location of the cities and the properties of distance metrics used to evaluate candidate solutions. Considering TSP as a black-box optimization problem puts EHBSA in a slightly more difficult position compared to most other TSP solvers.

Ten runs were performed for each TSP instance and each parameter setting. Each run was terminated either when the optimal tour was found, when the population was converged, or when the number of evaluations reached E_{max} Values of E_{max} were 50000, 500000, and 1000000 for gr24, gr48, and pr76, respectively. Population sizes of 60, 120, 240 were used for EHBSA; for other algorithms, we used additional two population sizes of 480 and 960. The bias ratio B_{ratio} from Eq. 3 is set to 0.005 for all experiments. Determining an optimal value of this parameter is rather difficult and it remains an important topic for future research, but from our experience EHBSA seems to be robust with respect to the choice of this parameter.

We evaluated the performance of the different algorithms by measuring the following quantities:

- *#OPT*, the number of runs in which the algorithm succeeded in finding the optimal tour,
- *ANE*, the average number of evaluations to find the global optimum in those runs where it did find the optimum,
- STD, the standard deviation of the number of evaluations until the global optimum was found, and
- *Aver*, the average length of the best solution over the ten runs.

4.2 Experimental Results

Results on gr24 are shown in Table 1. EHBSA/WO has found the optimal tour in 8, 8, and 10 runs (out of 10 total runs) with N = 60, 120, and 240, respectively. On the other hand, EHBSA/WT/n has found the optimum tour in all 10 runs for all parameter settings used except for EHBSA/WT/4 with N=60. The ANEs of EHBSA/WT/2 and EHBSA/WT/3 was 10713, and 9845, respectively, indicating good performance. The results show that the performance of EHBSA/WT is much better than that of EHBSA/WO.

From other operators, eER showed best performance. The eER with N = 240, 480, and 960 found the optimal tour 10 times and the ANE with N = 240 was 13394, which is a little larger than ANE for EHBSA/WT/n with N = 60. OX showed worse performance than eER. PMX showed worse performance than both eER and OX.

Comparing the performance of EHBSA/WT with that of other operators, EHBSA/WT is slightly better than eER and it is significantly better than OX and PMX. One big difference between EHBSA/WT and eER is that EHBSA/WT requires a smaller population size to work than eER. For gr24, only one PMBGA presented in Robles et al. (2002) was able to find the optimal tour (with population size N=1000) without the use of local search, which indicates much worse performance of those methods compared to practically any variant of EHBSA tested here.

Table 1. Results of gr24

Model		Population Size																			
		60				120				240				480				960			
		#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver
EHBSA	WO	8	17359	10996	1275	8	19608	2436	274	10	38929	4228	1272								
	WT/2	10	10713	7419	1272	10	15472	2289	1272	10	29454	4095	1272								
	WT/3	10	9845	1745	1272	10	16016	2276	1272	10	30835	2737	1272								
	WT/4	9	11320	3349	1272	10	19015	3351	1272	10	33999	3864	1272								
	WT/5	10	13186	1726	1272	10	18327	3380	1272	10	38676	5498	1272								
OX		0	-	-	1345	1	22449	0	1303	4	34140	3793	1295	1	48674	0	1301	0	-	-	1484
eER		1	4738	0	1299	7	6237	654	1276	10	13394	1726	1272	10	23785	1338	1272	10	42767	2404	1272
PMX		0	-	-	1492	0	-	-	1414	2	23191	1798	1341	1	49442	0	1316	0	-	-	1572
(Robles et al.,		#OPT and ANE are not available, best length = 1272 with EMNA, Aver = 1285 with EMNA*																			

Optimum: 1272

E_{max} = 50000

* Best data without heuristic using discrete EDA. Maximum number of evaluations is 50000.

Results on gr48 are shown in Table 2. EHBSA/WO could not find the optimal tour for gr48, which indicates bad scalability of the algorithm. On the other hand, EHBSA/WT/*n* found the optimum tour in all 10 runs for all parameter settings except for EHBSA/WT/2 with N = 60. The ANEs for EHBSA/WT/3 and EHBSA/WT/5 were 64541 and 76552, respectively, indicating good performance. Thus, similarly as for gr24, the results indicate that the performance of EHBSA/WT is better than that of EHBSA/WO.

Other operators showed again worse performance than EHBSA/WO. From other operators, eER ranked the best, OX ranked the second best, whereas the performance of PMX was again the worst. The best #OPT of eER is 5 with N = 960 and the ANE for this case is 166286, which is much larger than EHBSA/WT/*n*. PMX could not find the optimal tour. The results presented in Robles et al. (2002) indicate that all PMBGAs proposed in that study performed much worse than EHBSA/WT/*n*, because they have failed to find the optimal tour for gr48 without the use of local search in all the runs.

Table 2. Results of gr48

Model		Population Size																			
		60				120				240				480				960			
		#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver
EHBSA	WO	0	-	-	5129	0	-	-	5093	0	-	-	5511								
	WT/2	8	109304	54760	5048	10	144032	29115	5046	10	256733	31573	5046								
	WT/3	10	64541	15100	5046	10	101336	10968	5046	10	215003	28109	5046								
	WT/4	10	76552	39570	5046	10	132005	40928	5046	10	227231	38228	5046								
	WT/5	10	95896	39191	5046	10	183356	29095	5046	10	287050	38949	5046								
OX		0	-	-	5527	0	-	-	5268	0	-	-	5200	1	162154	0	5099	2	287852	6706	5082
eER		0	-	-	5653	0	-	-	5233	0	-	-	5098	2	95075	4168	5072	5	166286	4932	5058
PMX		0	-	-	8285	0	-	-	7374	0	-	-	6859	0	-	-	6116	0	-	-	5860
(Robles et al.,		ANE is not available, best length = 5122, Aver = 6717 with MIMIC*																			

Optimum: 5046

E_{max} = 500000

* Best data without heuristic using discrete EDA. Maximum number of evaluations is 50000.

Results on pr76 are shown in Table 3. EHBSA/WO could not find the optimum tour in pr76, which confirms bad scalability of this variant of EHBSA. On the other hand, EHBSA/WT/*n* found the optimal tour in almost all cases. With N = 60, EHBSA/WT/2, 3, 4, and 5 found the optimal tour 4, 4, 9, and 10 times, respectively. With N = 120, EHBSA/WT/2, 3, 4, and 5 found the optimal tour 9, 9, 9, and 10 times, respectively, showing the best performance. Thus, we can see the performance of EHBSA/WT is much better than the performance of EHBSA/WO in this experiment. From other operators, only eER was able to find the optimal tour, what happened only in one run with $N = 480$ and and in 3 runs for $N = 960$, showing worse performance than EHBSA/WT. OX and PMX could not find the optimal tour. There are no experimental results of applying PMBGAs from Robles et al. (2002) to pr76, so the performance of EHBSA cannot be compared to any of those methods.

Table 3. Results of pr76

Model		Population Size																			
		60				120				240				480				960			
		#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver
EHBSA	WO	0	-	-	119136	0	-	-	128208	0	-	-	142206								
	WT/2	4	360128	180323	108352	9	457147	65821	108174	7	871319	55042	108201								
	WT/3	4	248091	57073	108385	9	472719	63530	108171	8	853801	77162	108201								
	WT/4	9	341482	139089	108247	9	607544	146197	108247	0	-	-	108496								
	WT/5	10	494674	85334	108159	10	797963	121591	108159	0	-	-	108807								
OX		0	-	-	129603	0	-	-	121642	0	-	-	116591	0	-	-	113412	0	-	-	112259
eER		0	-	-	142003	0	-	-	122217	0	-	-	111839	1	179675	0	109119	3	394887	22321	108507
PMX		0	-	-	236827	0	-	-	213528	0	-	-	187601	0	-	-	164883	0	-	-	158515

Optimum: 108159

$E_{max} = 1000000$

5 Improving Performance of EHBSA With Local Search

Similarly as for other optimization problems and recombination-based optimizers, using local search for improving solutions locally as the search progresses can significantly improve the performance of EHBSA. In this section we show an example of such a hybrid approach, where EHBSA is combined with a simple local heuristic for improving TSP solutions called 2-OPT. The section first describes the basic procedure of 2-OPT and how 2-OPT can be incorporated into EHBSA. Next, the section presents promising results of applying the hybrid method EHBSA+2-OPT to several larger instances of TSP and compares these results with those obtained by hybridizing other techniques included in the comparison.

5.1 Combining EHBSA with 2-OPT

2-OPT proceeds by checking pairs of nonadjacent edges in a given tour, and computing the improvement in the tour length after rearranging these pairs of edges by exchanging the terminal nodes of the two edges in each pair as shown in Figure 8. If no pair of edges can be rearranged to improve the current tour, the algorithm is terminated. Otherwise, the pair of edges that improves the current tour the most is rearranged, and the algorithm is executed again.

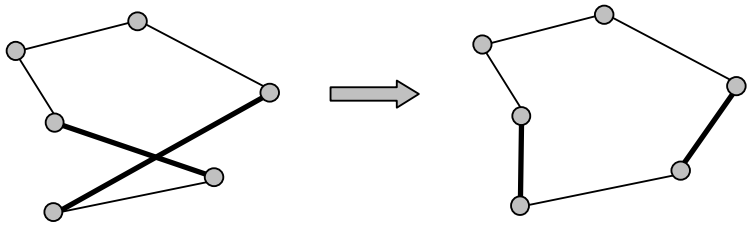

Figure 8. An example of one step of 2-OPT local search improving the total cost of the tour in TSP.

Since there are L edges for a tour of length L, checking all pairs of edges can be done in $O(L^2)$ steps. Determining an upper bound on the number of passes through all edges until no more improvement is possible is more difficult; a conservative bound would assume that all pairs of edges can be rearranged, but in practice no more than L improvements can be expected to take place, yielding a bound $O(L^3)$.

In EHBSA we use 2-OPT to improve every newly generated solution. Applying the 2-OPT heuristic to each new solution generated by EHBSA thus increases the computational cost of generating the new solution from $O(L^2)$ to $O(L^3)$.

5.3 Empirical Results for EHBSA/WT with 2-OPT

In the experiments presented in this section, the maximum number of evaluations is set to 100,000 for all tested TSP instances. The remaining settings are the same as those for the experiments presented earlier in the paper.

The results for a 226-city problem pr226 are shown in Table 4. In all cases, EHBSA/WT shows very good performance. Already with populations of size N=15, EHBSA/WT converges to the optimum in 7 to 9 runs, depending on the number of cut points. For the populations of N=30 and more, EHBSA/WT converges to the optimum in all 10 runs. The average number of evaluations until convergence to the optimum is small in all cases, and it increases with the population size. Note that the number of evaluations for solving the 226-city

problem with a hybrid EHBSA+2-OPT is lower than that for solving the 24 city problem with pure EHBSA; using local search thus clearly improves the performance and reliability of EHBSA as was expected.

On the other hand, other recombination operators perform rather poorly. OX is not able to find the optimal tour in any of the 10 runs with N=15, it succeeds in 6 out of the 10 runs with N=30, and only for populations N=60 and more the convergence with OX becomes more reliable, yielding success in at least 8 or 9 out of 10 runs. Extended ER (eER) performs even worse than OX, yielding only a few successful runs for N=30 and N=60, while achieving more reliable convergence only for population sizes N=240 and more. The performance of PMX is similar to that of OX. Comparing the population sizes for the reliable parameter settings reveals that EHBSA perform best out of all tested algorithms, OX and PMX are second best, and eER performs worst.

Table 4. Results for pr226.

Population Size			15				30				60				120			
Measure			#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver
Model	EHBSA	WT/2	8	514	138	80393	10	877	138	80369	10	1711	92	80369				
		WT/3	7	692	121	80369	10	1076	121	80369	10	1943	283	80369				
		WT/4	9	816	192	80369	10	1362	192	80369	10	2456	374	80369				
		WT/5	9	911	189	80369	10	1534	189	80369	10	2669	567	80369				
	OX		0	-	-	80536	6	921	383	80424	9	2429.1	753	80369	8	3115	753	80370
	eER		0	-	-	80475	0	-	-	80388	2	1254.5	113	80375	4	2430	531	80372
	PMX		1	578	0	80516	2	822	339	80389	4	1688.5	138	80380	9	3345	638	80369

Population Size			240				480				960			
Model	EHBSA	WT/2												
		WT/3												
		WT/4												
		WT/5												
	OX		10	5613	1321	80370	10	9850.3	1345	80369	10	18610	1798	80369
	eER		9	4135.4	826	80369	9	8274	1542	80369	10	11190	2705	80369
	PMX		10	5388	578	80369	10	9182.9	1390	80369	10	17242	2482	80369

Optimum: 80369

Table 5 shows that on the 318-city problem lin318, EHBSA/WT again proves its robust and scalable performance by reaching the optimum reliably even with small populations of N=15 individuals. The number of evaluations increases compared to the 226-city problem, but it is still smaller than that for the 24-city problem without the local search. This reconfirms the advantages of using local search in combination with a recombination-based method EHBSA/WT. The performance of OX and PMX decreases compared to the 226-city TSP, but for some settings these operators yield comparable performance to EHBSA with some settings. However, it seems that this TSP instance is still relatively easy for OX and PMX. Extended ER performs again worst of all compared methods.

Table 5. Results for lin318

Population Size			15				30				60				120			
Measure			#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver
Model	EHBSA	WT/2	8	7133	2821	42046	10	10718	7419	42029	10	10004	4808	42029				
		WT/3	10	11561	5031	42029	10	7648	965	42029	10	20679	4866	42029				
		WT/4	10	13360	7677	42029	10	24558	20815	42029	10	28903	12494	42029				
		WT/5	9	20746	12529	42035	10	19364	8800	42029	10	24982	10576	42029				
	OX		0	-	-	42400	0	-	-	42159	2	5191	315	42127	8	8639	1180	42037
	eER		0	-	-	42240	0	-	-	42124	0	-	-	42068	1	7198	0	42054
	PMX		0	-	-	42417	1	1833	0	42184	3	3993	337	42083	4	8713	1688	42056

Population Size			240				480				960			
Model	EHBSA	WT/2												
		WT/3												
		WT/4												
		WT/5												
	OX		10	14133	2042	42041	10	26820	3736	42029	10	45055	5129	42029
	eER		6	15603	1155	42040	10	33147	2619	42029	10	61717	4160	42029
	PMX		10	15090	2789	42029	10	23934	1123	42029	10	47549	4546	42029

Optimum: 42029

Table 6 shows that increasing the number of cities from 318 to 439 in TSP instance pr439 necessitates a slight increase in the population sizes used in EHBSA/WT, especially for EHBSA/WT with a low number of cut points (n=2, 3, and 4). However, already for *N*=30 the convergence of EHBSA/WT is reliable for all settings except for *n*=2, and for *N*=60, the convergence of EHBSA/WT is reliable also for *n*=2. For many settings, the number of evaluations for the 439-city problem decreases compared to the smaller problem of 318-cities presented above, which indicates good scalability of EHBSA/WT. However, other recombination operators show a significant decrease in performance compared to the 318-city problem lin318. Extended ER is not capable of achieving reliable convergence to the optimum even with a population of size *N*=960, whereas convergence of PMX and OX becomes reliable only with N=960, yielding a significantly inferior performance compared to EHBSA/WT with almost any settings.

Table 6. Results for pr 439

Population Size			15				30				60				120			
Measure			#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver	#OPT	ANE	STD	Aver
Model	EHBSA	WT/2	1	4308	0	107240	5	9535	6919	107228	9	10293	1316	107218				
		WT/3	3	5531	2963	107249	7	11937	4777	107222	10	13447	2019	107217				
		WT/4	5	13922	13981	107228	10	11116	5615	107217	10	17650	3323	107217				
		WT/5	8	11892	4948	107226	10	17155	8182	107217	10	21585	5270	107217				
	OX		0	-	-	107826	0	-	-	107371	0	-	-	107373	1	8875	0	107293
	eER		0	-	-	107473	0	-	-	107285	0	-	-	107222	2	19441	11447	107273
	PMX		0	-	-	107768	0	-	-	107483	0	-	-	107345	1	6689	0	107290

Population Size			240				480				960			
Model	EHBSA	WT/2												
		WT/3												
		WT/4												
		WT/5												
	OX		4	14398	1171	107247	2	25841	4372	107247	8	55550	7219	107247
	eER		4	34603	16951	107223	3	54616	4084	107224	1	74511	0	107222
	PMX		2	14923	333	107247	4	29773	2930	107236	8	62157	7863	107224

Optimum: 107217

To summarize the results for a hybrid EHBSA/WT combined with a 2-OPT local heuristic, we can observe that EHBSA/WT provides robust performance for even large TSP instances, whereas other recombination methods for TSP provide performance that is inferior with respect to reliability and computational efficiency. Comparing the results to those presented in Robles et al. (2002), EHBSA/WT is capable of solving qualitatively larger instances, and it is able to do this more reliably and efficiently than the methods from Robles et al. (2002) can do for significantly smaller problems. EHBSA/WT thus provides a promising solution to TSP, while not limiting the application to this class of problems. We believe that using more advanced local heuristics, such as Lin-Kernighan (LK) (Lin & Kernighan, 1973; Helsgaun, 2000), will provide results competitive with state-of-the-art techniques for solving TSP. However, the use of EHBSA in other problem domains, such as flow shop scheduling, may be a better target area for the proposed algorithm.

6 Summary and Conclusions

In this paper we have proposed several variants of probabilistic model-building genetic algorithms (PMBGAs) based on learning and sampling an edge-histogram matrix for solving permutation problems. Learning an edge-histogram matrix consists of computing the edge-histogram matrix for a population of promising candidate solutions.

Two methods for sampling the edge histogram matrix have been proposed: (1) edge-histogram-based sampling without template (EHBSA/WO), and (2) edge-histogram-based sampling with template (EHBSA/WT). EHBSA/WO uses only the learned edge histogram matrix to generate new candidate solutions, whereas EHBSA/WT uses a template selected from the population of promising candidate solutions as a starting point and modifies the template to generate a new solution. A tunable procedure EHBSA/WT/*n* for selecting the elements of the template permutation to be sampled anew has been proposed. EHBSA/WT/*n* uses a scheme similar to *n*-point crossover to choose a segment of the template that is to be resampled to generate a new solution. One of the important features of an edge histogram matrix is that this model can be learned incrementally, which allows for optimization with the use of only little memory.

EHBSA/WO and EHBSA/WT have been applied to several benchmark instances of the traveling salesman problem (TSP) with and without the use of local search. EHBSA/WT has performed significantly better than EHBSA/WO and other PMBGAs proposed in the past for solving this type of problems. EHBSA has also proven to provide significantly better results than some other popular two-parent recombination techniques for permutations, including extended edge recombination, OX, and PMX. Experimental results indicate that another advantage of EHBSA is the use of significantly smaller population sizes than those that are necessary with most other evolutionary algorithms for permutation problems.

Local search has proven to be advantageous and it has decreased the computational requirements for find-

ing a globally optimal tour in TSP, providing a method capable of solving significantly larger problems of hundreds of cities. However, in future more in-depth empirical analysis should be performed with more advanced local searchers and the results should be compared to state-of-the-art techniques for solving TSP. However, note that EHBSA does not use any specific features of TSP such as the existence of locations for all elements in the permutation or the distance metric for estimating the distances between the cities; all information is acquired automatically by learning and sampling alternative permutations. That is why EHBSA can be used to solve other problems defined on permutations. For example, EHBSA can be easily extended to solution of scheduling problems as described in Tsutsui & Miki (2002).

Despite the promising results, the paper should be understood as one of the first steps toward scalable solution of large permutation problems with PMBGAs, because PMBGAs proposed in the past for solving permutation problems are capable of solving only rather small problems. The paper provides many opportunities for further research in this area, some of which have been mentioned throughout the paper.

Acknowledgments

This research was 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.

David E. Goldberg and Martin Pelikan's contribution was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620-00-0163. Research funding for this work was also provided by a grant from the National Science Foundation under grant DMI-9908252. The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research, the National Science Foundation, or the U.S. Government.

References

Bosman, P. A. N., & Thierens, D. (2000). Continuous iterated density estimation evolutionary algorithms within the IDEA framework. Workshop Proceedings of the Genetic and Evolutionary Computation Conference 2000 (GECCO-2000), 197-200.

Bosman, P. A. N., & Thierens, D. (2001). Crossing the Road to Efficient ID*E*As for Permutation Problems. *Proceedings of the Genetic and Evolutionary Computation Conference - GECCO-2001*, 219-226.

Bosman, P. A. N., & Thierens, D. (2002). Permutation Optimization by Iterated Estimation of Random Keys Marginal Product Factorizations. *Parallel Problem Solving From Nature*, 331-340.

Dorigo M., Maniezzo, V. and Colorni, A. (1996). The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Trans. on Systems, Man, and Cybernetics-Part B*, Vol. 26, No. 1, 29-41.

Goldberg, D.E. (1989). *Genetic algorithms in search, optimization and machine learning*. Reading, MA: Addison-Wesley publishing company.

Goldberg, D. E. (2002). The design of innovation: Lessons from and for competent genetic algorithms, Volume 7 of Genetic and Evolutionary Computation. Kluwer.

Harik, G. (1995). Finding multimodal solutions using restricted tournament selection. *Proceedings of the 6th International Conference on Genetic Algorithms*, Morgan Kaumann, 28-31.

Helsgaun, K. (2000). An effective implementation of the lin-kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126:106-130.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: The University of Michigan Press.

Larrañaga, P., & Lozano, J. A. (Eds.) (2002). Estimation of distribution algorithms: A new tool for evolutionary computation. Boston, MA: Kluwer.

Lin, S. & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, 21:498-516.

Mahfoud, S. W. (1992). Crowding and preselection revisited. *Parallel problem solving from nature 2*, 27-36.

Muehlenbein, H., & Paass, G. (1996). From recombination of genes to the estimation of distributions I. Binary parameters. *Parallel Problem Solving from Nature,* 178-187.

Oliver, I., Smith, D., and Holland, J. (1987). A study of permutation crossover operators on the travel salesman problem. *Proc. of the 2nd Int. Conf. on Genetic Algorithms*, 224-230.

Pelikan (2002). Bayesian optimization algorithm: From single level to hierarchy. Doctoral dissertation, University of Illinois at Urbana-Champaign, Urbana, IL. Also IlliGAL Report No. 2002023.

Pelikan, M., Goldberg, D. E., & Lobo, F. (2002). A survey of optimization by building and using probabilistic models. *Computational Optimization and Applications, 21*(1), 5-20. Also IlliGAL Report No. 99018.

Robles, V., Miguel, P. D., & Larrañaga, P. (2002). Solving the traveling salesman problem with EDAs. In Larranaga, P. and Lozano, J. A. (Eds.), *Estimation of Distribution Algorithms*, 211-229, Kluwer.

Starkweather, T., McDaniel, S., Mathias, K, Whitley, D, and Whitley, C. (1991). A comparison of genetic sequence operators. *Proc. of the 4th Int. Conf. on Genetic Algorithms*, Morgan Kaufmann, 69-76.

Thierens, D. (1995). Analysis and design of genetic algorithms. Doctoral dissertation, Katholieke Universiteit Leuven, Leuven, Belgium.

Tsutsui, S., Pelikan, M., & Goldberg, D. E. (2001). Evolutionary Algorithm using Marginal Histogram Models in Continuous Domain. *Workshop Proceedings of the 2001 Genetic and Evolutionary Computation Conference*, 230-233.

Tsutsui (2003). Edge histogram based sampling and its application to ACO. *Proceedings of the Frontiers of Evolutionary Algorithms Conference 2003 (FEA 2003)*.

Tsutsui, S., & Miki, M. (2002). Solving Flow Shop Scheduling Problems with Probabilistic Model-Building Genetic Algorithms using Edge Histograms. *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution And Learning (SEAL02)*, Singapore.

Whitley, D. (1989). The GENITOR algorithm and selective pressure: Why rank-based allocation of reproductive trials is best. *Proc. of the 3rd Int. Conf. on Genetic Algorithms*, Morgan Kaufmann, 116-121.

Whitley, D., Starkweather, T., & Fuquay, D. (1989). Scheduling problems and traveling salesman problem: The genetic edge recombination operator. *Proc. of the 3rd Int. Conf. on Genetic Algorithms*, Morgan Kaufmann.