The Natural Crossover for the 2D Euclidean TSP

Soonchul Jung and Byung-Ro Moon
School of Computer Science and Engineering
Seoul National University
Seoul, 151-742 Korea
samuel@soar.snu.ac.kr, moon@cs.snu.ac.kr

Abstract

For the traveling salesman problem various search algorithms have been suggested for decades. In the field of genetic algorithms, many genetic operators have been introduced for the problem. Most genetic encoding schemes have some restrictions that cause more-or-less loss of information contained in problem instances. We suggest a new encoding/crossover pair which pursues minimal information loss in chromosomal encoding and minimal restriction in recombination for the 2D Euclidean traveling salesman problem. The most notable feature of the suggested crossover is that it is based on a totally new concept of encoding. We also prove the theoretical validity of the new crossover by an equivalence-class analysis. The proposed encoding/crossover pair outperformed both distance-preserving crossover and edge-assembly crossover, two state-of-the-art crossovers in the literature.

1 INTRODUCTION

Given n cities and a distance matrix $D = [d_{ij}]$ where d_{ij} is the distance between city i and city j, the traveling salesman problem (TSP) is the problem of finding a permutation π that minimizes $\sum_{i=1}^{n-1} d_{\pi(i),\pi(i+1)} + d_{\pi(n),\pi(1)}$. In metric TSP the cities lie in a metric space (i.e., the distances satisfy the triangle inequality). In Euclidean TSP, the cities lie in $\Re^d$ for some d; the most popular version is 2D Euclidean TSP where the cities lie in $\Re^2$. Euclidean TSP is a sub-case of metric TSP. In this paper, we only consider 2D Euclidean TSP.

Euclidean TSP as well as the general TSP is known to be NP-hard [16]. Smith [33] showed that 2D Euclidean TSP could be solved to optimality in $O(2^{O(\sqrt{n})})$ time. Christofides [10] proposed a polynomial-time approximation algorithm for metric TSP that finds a tour of cost at most 3/2 times the optimum. In spite of two decades of efforts, Christofides's algorithm remains an algorithm with the best bound. It was proven that there is no polynomial-time approximation scheme (PTAS) for metric TSP unless P = NP [3]. Recently Arora [2] suggested a PTAS for Euclidean TSP that, for every fixed $c > 1$, finds a tour of cost at most $1 + 1/c$ times the optimum in $O(n^{d+1}(\log n)^{O(\sqrt{d}c)^{d-1}})$ time where d is the dimension. However, his implementation result was very slow even for moderate values of c and there is no evidence that it can be implemented to be practically useful.

TSP and its variants have many practical applications such as vehicle routing, PCB design, x-ray crystallography, etc. The problem is such a hot topic that one can find more than 1,700 related papers in recent five years from INSPEC[1] database. There have been studies in various fields to get reasonable suboptimal solutions for the problem. 2-Opt, 3-Opt, and Lin-Kernighan (LK) algorithm [5],[20],[19],[25] are representative local optimization algorithms. Among them, LK algorithm is the definite champion as a local optimization heuristic. General search methods such as simulated annealing [24], artificial neural network [32], genetic algorithms (GAs) [18], and tabu search [13] have been also applied to TSP.

TSP is one of the most actively studied topics in GA community, too. The most successful experimental results have been provided by hybrid GAs which incorporate local optimization heuristics into GAs [21],[7],[28],[30],[23]. A GA that does not use a local optimization heuristic is called a pure GA. Objects of crossover operators in pure GAs are different from those of crossover operators in hybrid GAs. Crossover

[1] http://www.inspec.org

operators in a pure GA should sometimes be able to produce better offsprings than parents. On the other hand, crossover operators in a hybrid GA do not have to produce better offsprings than parents because they only have to provide initial solutions for a local optimization heuristic. Thus crossover operators are free from fine-tuning in a hybrid GA and this allows more perturbation of solutions than in a pure GA. Various crossover operators such as order crossover [12], cycle crossover [31], partially matched crossover [17], edge-recombination crossover [36],[34], and matrix crossover [19] were suggested. Recently Freisleben and Merz [28] suggested distance-preserving crossover (DPX) for TSP. To the best of our knowledge, Freisleben and Merz's hybrid GA with DPX and LK local optimization [25] is practically the best GA for TSP among published algorithms.

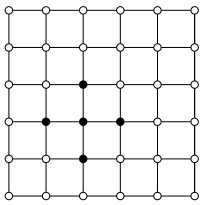
In traditional GAs, encoding a solution into a linear chromosome or into a restricted form of a 2D chromosome is considered to lose information contained in the original problem instance. We have been concerning on this topic and the information loss could be partly alleviated by gene reordering or using some multi-dimensional encodings [6],[7],[9],[29]. However, some information loss is not avoidable as far as we use such restricted forms of chromosomes. In this paper, we suggest an extreme strategy along the line. Suggested are an encoding strategy with no information loss and a matching new crossover operator that pursues minimal restriction in chromosomal cutting to achieve maximal creation power of new solutions. We also prove the theoretical validity of the suggested crossover by an equivalence-class analysis. We incorporate the new encoding and crossover strategy in hybrid GAs with LK local optimization heuristic and compare the experimental results against two representative hybrid GAs with DPX [15],[28] and edge-assembly crossover [30], respectively.

The paper is organized as follows. Section 2 explains previous approaches of GAs for TSP and Section 3 describes the new encoding/crossover pair. Section 4 presents the experimental results. Finally Section 5 makes conclusions.

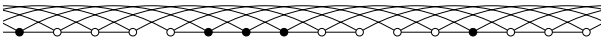
2 BACKGROUNDS

2.1 GA APPROACHES FOR TSP

GAs for TSP can be divided into order-based approach, analytic approach, and locus-based approach according to the way they encode a tour.



(a) A grid graph



(b) Its linear representation

Figure 1: An example of information loss due to linear encoding

2.1.1 Order-Based Approach

As TSP is a permutation problem, it is natural to encode a tour by enumerating the city indices in order. This approach has been dominant in GAs for TSP. In such an encoding, the chromosomal location of a city has little meaning; only the sequence is meaningful. Some representative crossovers performed on order-based encodings include cycle crossover [31], partially matched crossover [17], order crossover [12], etc.

2.1.2 Analytic Approach

In this approach, the crossover is performed by analyzing the parent tours. Edge-recombination crossover [36],[34] analyzes the cities adjacent to each city in the two parent tours and constructively generates a new tour from the information. Edge-assembly crossover (EAX) [30] generates special subcycles by alternately taking edges of the two parent tours, decomposes a parent tour using the subcycles, and repairs the intermediate individual to complete a new Hamiltonian cycle. Distance-preserving crossover (DPX) [15],[28] extracts the common edges of two parent tours and adds some edges to complete a new tour. In these schemes, the encoding is not important. Although they do not need any explicit form of chromosomes, they are still genetic in the sense that they combine characteristics of the two parents. DPX and EAX showed impressive performance; to the best of our knowledge, a hybrid GA with DPX and LK local optimization has shown the best performance among GAs for TSP. We will be comparing our proposed approach against it.

2.1.3 Locus-Based Approach

The locus-Based approach is traditional and the most popular in the field of genetic algorithms. In a locus-based encoding, every gene has a fixed location and the gene values have meaning only associated with

the locations. The locus-based approach has a merit that genes' geographical relationship affects the performance of GAs. Since genetic algorithms are motivated by natural evolution and the nature uses DNAs with locus-based encoding, this approach has been standard for most GAs. For TSP, however, the order-based approach has been dominant and, recently, the analytic approach showed impressive results. One possible reason is that TSP is a permutation problem and representing a solution by an enumeration (permutation) of cities is natural. Another reason may be that it is not easy to represent a tour by a traditional locus-based encoding.

The first locus-based encoding for TSP was suggested by Homaifar *et al.* [19]. They used an $n \times n$ matrix $M = [m_{ij}]$ to represent a tour; if the element m_{ij} is 1, city j follows city i in the tour. A drawback in this approach is that the crossover takes $\Theta(n^2)$ to complete the matrix. Bui and Moon [7] suggested a locus-based "linear" encoding. The chromosome consists of an array of n integers. The i^{th} element represents the city following city i in the tour. This crossover takes $\Theta(n)$. But because both of them map each city to a location (or a set of locations) in the chromosome, it is not avoidable that cities' geographical relationships are distorted, which we believe will distort GAs' search to some extent. To alleviate this information loss, Bui and Moon [7] reordered the genes of cities and showed performance improvement. It is, however, just an alleviation of information loss. This is the motivation of this study. In this paper, we suggest an extreme approach along the line.

2.2 MULTI-DIMENSIONAL ENCODING

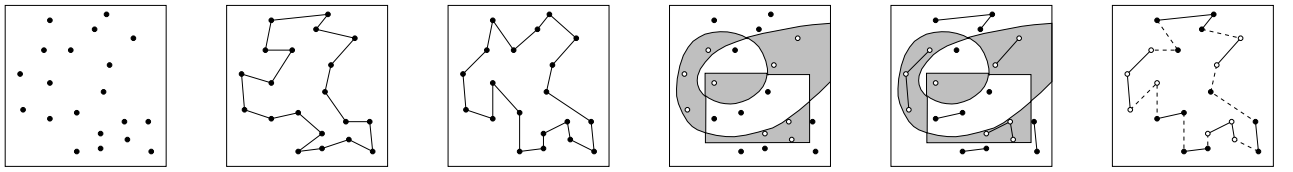
A traditional one-dimensional encoding is easily implemented but has an intrinsic, potential drawback that any solution in a problem has to be represented by a linear string. This highly probably causes information loss contained in the problems. For example, let us consider a grid graph like Figure 1(a). Each vertex has up to four neighbors in the graph. But if the graph is converted to a linear string like Figure 1(b), each vertex has at most two neighbors instead of four; it is unavoidable that considerable adjacency information of the original graph is destroyed. This is a symbolic description of information loss by linear encodings. It is known that a fairly high dimension is needed to embed a general graph with reasonable distortion [26]. A two-dimensional encoding/crossover pair can reflect more geographical linkages of genes than one-dimensional encoding/crossover pairs [29]. Cohoon and Paris [11] proposed a two-dimensional crossover which chooses a small rectangle from one parent and copies the genes in the rectangle into the offspring, with the rest of the genes copied from the other parent. Anderson *et al.* [1] suggested block-uniform crossover which tessellates a two-dimensional chromosome into $i \times j$ blocks; genes in each block are copied as a group from a uniformly-selected parent. Bui and Moon [8] suggested an n-dimensional generalization of traditional crossover which chooses k crossover points on an n-dimensional chromosome. Kahng and Moon [22] suggested a general framework for devising an n-dimensional crossover. The principal merit of two- or higher-dimensional chromosomal encodings is that they preserve more geographical linkages of genes than the traditional linear encodings [8].

Although multi-dimensional chromosomal encodings contain more information than one-dimensional chromosomal encodings, restricted chromosomal encodings like matrices or grids still cause information loss. In case of the 2D Euclidean TSP, if we have to assign each city to a slot [7] or a row of a matrix encoding [19], cities' Euclidean relationship is affected because the positions of cities are mostly not that evenly distributed. More unrestricted chromosomal encodings are desired in this context.

3 ENCODING WITHOUT INFORMATION LOSS AND NATURAL CROSSOVER

For a 2D Euclidean TSP instance, cities are located in a two dimensional space. The relationships between cities are determined by their physical locations. Restricted chromosomal encodings like matrices or grids as well as linear encodings cannot contain exact information of cities' locations.

Bui and Moon [6],[7],[9] showed that the performance of a genetic algorithm with a locus-based linear encoding could be dramatically affected by reordering genes' locations in the chromosome. Their key idea was to help closely related genes locate closely in the encoding. They showed that, due to the reordering, high-quality schemata could be transformed to better forms to survive [9]. The same problem occurs in the encoding of TSP. A reordering technique was suggested in [7] for a locus-based linear encoding for TSP. Although the cities' geographical information is better reflected as a result of reordering, the information loss is intrinsically unavoidable by mapping 2D TSP to such a one-dimensional encoding. More or less information loss is thus unavoidable as far as we use some locus-based encoding other than the original distribution itself of cities. It is the motivation of this study.



(a) cities (b) tour A (c) tour B (d) partitioned cities (e) intermediate sub-tours (f) new offspring

Figure 2: An example of natural crossover

We use the *graphic image itself* of a tour, which contains complete geographic information, as a chromosome; no information is lost in the encoding stage. In other words, the genotype corresponds to the phenotype. Figure 2(b) and (c) are two example chromosomes. The cutting lines for crossover are drawn on the 2D images.

3.1 THE NATURAL CROSSOVER

We name the suggested crossover as *natural crossover* (NX). We describe it in the following with Figure 2 as an example:

1. The graphic images of two tours are selected as parents (Figure 2(b) and (c)).
2. Curves are freely drawn on the 2D space where cities are located. They always partition the chromosomal space into two equivalent classes. Its validity is proven in the next subsection. As in Figure 2(d), curves partition the chromosomal space into two regions (white and gray), and every city belongs to one of the regions. Cities in one region are marked black and cities in the other region are marked white.
3. For every edge of the tour A, if both endpoints of the edge are marked black, it survives in the offspring; for every edge of the tour B, if both endpoints of the edge are marked white, it survives in the offspring. Then we have a number of disconnected sub-tours (Figure 2(e)).
4. Add random edges to make a valid tour (Figure 2(f)).

In the step 4, one may choose the shortest edges among those that do not cause a sub-cycle. We use random addition on purpose to provide more diverse perturbation. This is because we use a hybrid GA which applies local optimization on the solution. If we used the bitmap representation of a 2D image, the cost of the crossover would be too expensive. Instead we first draw the curves and examine the endpoints of edges in the tours as described in the above. The cost of the crossover thus grows linearly with respect to the number of cities.

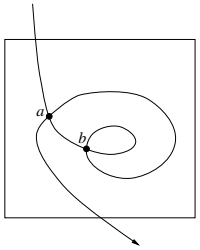


Figure 4: An example loop that contains a sub-loop

3.2 THEORETICAL VALIDITY

Given a multi-dimensional encoding, an arbitrary chromosomal cutting does not necessarily make a valid crossover. A sufficient condition for a cutting strategy to make a valid crossover is that every cutting divides the chromosomal positions into two disjoint partitions. Then one can copy the genes in one partition from one parent to the offspring and those in the other partition from the other parent. A formal method for proving the validity is to find an equivalence relation associated with the cutting and to show that chromosomal positions are divided into two equivalence-classes [22]. We prove the validity of NX using an equivalence-class analysis.

Consider a continuous two-dimensional space $D = T \times T$ where $T = [0, t]$.

Definition 1 *A* cut *is a closed curve or a curve that enters D (from the outside), moves inside D without any restriction, and exits D.*

Figure 3 shows some sample cuts.

Definition 2 *A* loop *is a trajectory starting at a point in D and returning to the same point.*

An arbitrary loop may contain one or more sub-loops

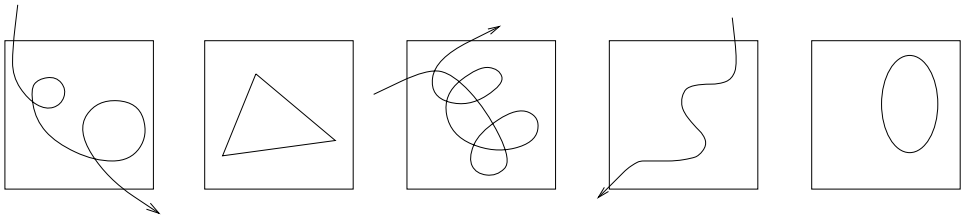


Figure 3: Sample cuts

as shown in Figure 4. In the figure, the loop starting at the point a contains another loop starting at the point b.

Fact 1 *For any singleton $S = \{A\}$ where A is a cut, and two points $x, y \in D - \{A\}$, the following two claims hold:*

1. *If there is a (continuous) path from x to y with an even number of crosses (intersections) with A, any other path from x to y also has an even number of crosses with A.*
2. *If there is a path from x to y with an odd number of crosses with A, any other path from x to y also has an odd number of crosses with A.*

Proof: (by mathematical induction on the number of loops in the cut A)

- If there is no loop, it is obvious that the space $D - \{A\}$ is divided into two partitions with respect to A. If a path from x to y has an even number of crosses with A, it is obvious that x and y are in the same partition and any other (continuous) path from x to y also has an even number of crosses with A. Similarly, it is also obvious that, if a path from x to y has an odd number of crosses with A, any other path from x to y also has an odd number of crosses with A.
- Assume that, for $k > 1$, the claims are true when the number of loops are less than k. Let the cut A contain k loops and let A' be the cut A minus an arbitrary sub-loop not containing any sub-loop. It is clear that there exists at least one such sub-loop. Then the number of loops in A' is $k - 1$ and the claims hold w.r.t. A' by the inductive assumption.
- Consider the case of Claim 1 that there is a path from x to y with an even number of crosses with A. i) If x and y are in the same side (inside or outside) w.r.t. the removed loop, the path (actually any path from x to y) has an even number (including 0) of crosses with the removed loop, and consequently the path must have an even number of crosses with A'. Then, by the inductive assumption, any path from x to y has an even number of crosses with A'. Therefore, after returning the removed loop, any path from x to y has an even number of crosses with A. ii) If x and y are in the opposite sides w.r.t. the removed loop, the path (actually any path from x to y) has an odd number of crosses with the removed loop, and consequently the path must have an odd number of crosses with A'. Then, by the inductive assumption, any path from x to y has an odd number of crosses with A'. Therefore, after returning the removed loop, any path from x to y has an even number of crosses with A. Claim 2 can be similarly proved. ∎

Fact 2 *For any set S of multiple cuts and two points $x, y \in D - S$, the following two claims hold:*

1. *If there is a path from x to y with an even number of crosses with S, any other path from x to y also has an even number of crosses with S.*
2. *If there is a path from x to y with an odd number of crosses with S, any other path from x to y also has an odd number of crosses with S.*

Proof: Omitted by space limitation.

Definition 3 *For a set S of cuts and two points $x, y \in D$, we say $(x, y) \in R_S$ if and only if there exists a continuous path from x to y which makes an even number of crosses with S.*

Since it is clear that R_S is reflexive, commutative, and transitive, R_S is an equivalence relation. Now consider an arbitrary point $x \in D - S$ and the equivalence class $x/R_S = \{y \in D - S | (x, y) \in R_S\})$. From Fact 2, $D - S - x/R_S$ also constitutes an equivalence class w.r.t. the relation R_S. We thus have the following conclusion.

Fact 3 *All cities of a 2D-Euclidean-TSP instance are divided into two equivalence classes with respect to one or more cuts of NX.*

Figure 5 shows three cutting examples and the corresponding equivalence classes (represented by whites

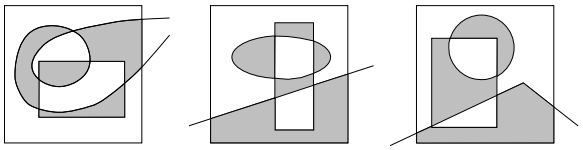


Figure 5: Example cuttings and corresponding equivalence classes

---

create a population of size N;

do {
  choose $parent_1$ and $parent_2$ from the population;
  $offspring \leftarrow$ crossover($parent_1$, $parent_2$);
  mutation($offspring$);
  local-improve($offspring$);
  replace(population, $offspring$);
} **until** (stopping condition)

return the best individual;

Figure 6: A typical steady-state hybrid genetic algorithm

and grays).

4 EXPERIMENTAL RESULTS

Figure 6 shows a typical steady-state hybrid genetic algorithm. Denote by DGA, EGA, and NGA the genetic algorithms using DPX, EAX, and NX, respectively. DPX and EAX were implemented based on the original papers [15],[28],[30]. We do not have an efficient implementation for drawing fully free curves in NX. Instead, we used a number of simple figures. k figures are chosen at random among circles, ellipses, rectangles, and straight lines allowing duplication. In this experiment, k was set to 5. Because the crossovers have different characteristics one another, some parameters of the GAs were set differently for each crossover.

- Population Size — 50 for DGA and NGA. The population size of EGA was given differently problem by problem as in the original paper [30]; the population sizes for lin318, pcb442, att532, rat783, dsj1000, and pcb3038 are 300, 450, 500, 800, 500, and 300, respectively.
- Selection operator — Tournament selection.
- Mutation operator — DGA and NGA used the double-bridge kick move which was known to be very effective from the literature [27],[21]. Figure 7 shows an example of the double-bridge kick move. The mutation probability is 0.1. EGA did not use mutation as recommended in [30].
- Local optimization — Both DGA and NGA used the LK algorithm. The LK used here is an advanced version incorporating the techniques of *don't-look bit* [4] and *segment tree* [14] which cause dramatic speed-up. EGA did not explicitly use a local optimization algorithm; but the crossover itself contains local optimization.
- Replacement operator — The replacement operator in [7] was used where i) the more similar parent (by Hamming distance) to the offspring is replaced if the offspring is better, ii) if not, the other parent is replaced if the offspring is better, iii) if not again, the worst in the population is replaced.

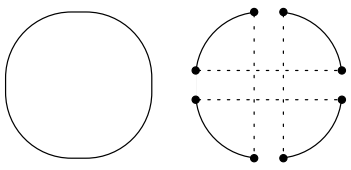


Figure 7: A double-bridge move

Table 1 shows the experimental results of the three GAs on six TSPLIB[2] benchmark problems of sizes ranging from 318 to 3038 cities. In the table, the parentheses below each graph name contain the optimal tour cost of the graph. The column "Best" and "Ave" show the best and the average results over 100 runs, respectively. Figures in parentheses next to the best and the average represent the percentages above the optimal costs. The column "$\sigma/\sqrt{n}$" shows the group standard deviation of 100 runs. The column "Gen" represents the average generation when the final tour appeared. The "CPU" shows the average CPU seconds on Pentium III 450MHz. The bold-faced numbers represent the best among the three algorithms.

NGA overall showed better solutions than the others. For lin318, att532 and dsj1000, the average results of NGA were better than those of the others within 5% risk. Although the average result of DGA was better than that of NGA for pcb3038, it is not as statistically stable as the above. If two algorithms produce the same result, a reasonable tie-breaking is giving preference to the algorithm consuming less running time. In this context, NGA performed best for five instances out of six. In the best solutions, all of the algorithms found the optimal solutions for the first four instances; for the two largest instances dsj1000 and pcb3038, NGA

[2] http://www.iwr.uni-heidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB.html

Table 1: The Experimental Results of the Three GAs with Different Crossovers over 100 runs

Graph	GA	Best(%)	Ave(%)	$\sigma/\sqrt{n}$	Gen(CPU)‡
lin318	DGA	**42029**(0.000)†	42033.44(0.011)	1.35	482(**35**)
(42029)	EGA	**42029**(0.000)	42033.56(0.011)	1.72	4196(50)
	NGA	**42029**(0.000)	**42029.00**(0.000)	0.00	412(36)
pcb442	DGA	**50778**(0.000)	**50778.00**(0.000)	0.00	612(53)
(50778)	EGA	**50778**(0.000)	50782.16(0.008)	1.15	7559(164)
	NGA	**50778**(0.000)†	**50778.00**(0.000)†	0.00	534(**31**)
att532	DGA	**27686**(0.000)	27697.58(0.042)	0.48	1490(106)
(27686)	EGA	**27686**(0.000)	27699.28(0.048)	0.87	11850(287)
	NGA	**27686**(0.000)†	**27695.61**(0.035)	0.71	1436(**76**)
rat783	DGA	**8806**(0.000)	**8806.00**(0.000)	0.00	1831(53)
(8806)	EGA	**8806**(0.000)	8811.61(0.064)	0.47	21690(975)
	NGA	**8806**(0.000)†	**8806.00**(0.000)†	0.00	1148(**35**)
dsj1000	DGA	18660436(0.004)	18663449(0.020)	275	4773(1305)
(18659688)	EGA	18660605(0.005)	18685949(0.141)	1535	18647(1003)
	NGA	**18660188**(0.003)	**18660752**(0.006)	152	1851(**733**)
pcb3038	DGA	137705(0.008)	**137760.55**(0.048)	4.28	12650(1880)
(137694)	EGA	137730(0.026)	138067.49(0.271)	10.97	15855(2548)
	NGA	**137695**(0.001)	137765.02(0.052)	4.55	10119(**816**)

† Win by tie-breaking with running time.
‡ CPU seconds on Pentium III 450Mhz

found visibly better solutions than the others. Under the above tie-breaking rule, NGA performed best for all six instances. It is also notable that NGA was visibly faster than the others. On the average, NGA was about 1.6 and 7.1 times faster than DGA and EGA, respectively. These results clearly show that NGA searches a problem space more effectively than DGA and EGA.

EGA was inferior to the others. Actually EGA showed poorer performance than in the original paper [30]. Watson *et al.* [35] also implemented EAX based on the original paper and made extensive experiments for att532. They reported the average tour length of 27709 and the best tour length of 27693 for 30 runs; we got 27699.28 and 27686 for 100 runs, respectively.

5 CONCLUSIONS

We suggested the natural encoding/crossover pair which minimized the restriction in the representation of problem instances of TSP. The proposed encoding/crossover pair uses a 2D tour image itself as a chromosome. It is an extreme strategy in the locus-based approach that no information is lost at all in the course of encoding. Its theoretical validity as a crossover was proved using an equivalence-class analysis.

The proposed crossover outperformed two representative crossovers in the literature. It seems to be due to the enhanced information-preserving ability of the proposed encoding and the minimal restriction of the crossover, but we do not have a solid explanation on how the crossover helps the search. It may be worth studying on the behavior of the natural crossover with focus on, e.g., the dynamics of schemata.

ACKNOWLEDGEMENTS

This work was partly supported by SNU Research Fund and Brain Korea 21 Project.

References

[1] C. A. Anderson, K. F. Jones, and J. Ryan. A two-dimensional genetic algorithm for the ising problem. *Complex Systems*, 5:327–333, 1991.

[2] S. Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998.

[3] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and intractability of approximation problems. In *33rd IEEE Symp. on Foundations of Computer Science*, pages 13–22, 1992.

[4] J. L. Bentley. Experiments on traveling salesman heuristics. In *1st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '90)*, pages 91–99, 1990.

[5] H. Braun. On traveling salesman problems by genetic algorithms. In *Workshop on Parallel Problem Solving from Nature*, pages 129–133, 1990.

[6] T. N. Bui and B. R. Moon. Hyperplane synthesis for genetic algorithms. In *Fifth International Conference on Genetic Algorithms*, pages 102–109, July 1993.

[7] T. N. Bui and B. R. Moon. A new genetic approach for the traveling salesman problem. In *IEEE Conference on Evolutionary Computation*, pages 7–12, 1994.

[8] T. N. Bui and B. R. Moon. On multi-dimensional encoding/crossover. In *Sixth International Conference on genetic Algorithms*, pages 49–56, 1995.

[9] T. N. Bui and B. R. Moon. Graph partitioning and genetic algorithms. *IEEE Transactions on Computers*, 45:841–855, 1996.

[10] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. In *Symposium on New Directions and Recent Results in Algorithms and Complexity*, page 441, 1976.

[11] J. P. Cohoon and W. Paris. Genetic placement. *IEEE Trans. on Computer-Aided Design*, CAD-6(6):956–964, 1987.

[12] L. Davis. Applying adapting algorithms to epistatic domains. In *International Joint Conference on Artificial Intelligence*, 1985.

[13] C. N. Fiechter. A parallel tabu search algorithm for large traveling salesman problems. *Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science*, 51:243–267, 1994.

[14] M. L. Fredman, D. S. Johnson, L. A. McGeoch, and G. Ostheimer. Data structures for traveling salesmen. In *4th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '93)*, pages 145–154, 1993.

[15] B. Freisleben and P. Merz. New genetic local search operators for the traveling salesman problem. In *4th Conference on Parallel Problem Solving from Nature*, pages 616–621, 1996.

[16] M. R. Garey, R. L. Graham, and D. S. Johnson. Some NP-complete geometric problems. In *8th Annual ACM Symposium on Theory of Computing*, pages 10–22, 1976.

[17] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[18] J. Grefenstette, R. Gopal, B. Rosmaita, and D. Gucht. Genetic algorithms for the traveling salesman problem. In *First International Conference on Genetic Algorithms and Their Applications*, pages 160–168, 1985.

[19] A. Homaifar, S. Guan, and G. Liepins. A new approach on the traveling salesman problem by genetic algorithms. In *Fifth International Conference on Genetic Algorithms*, pages 460–466, July 1993.

[20] P. Jog, J. Suh, and D. Gucht. The effect of population size, heuristic crossover and local improvement on a genetic algorithm for the traveling salesman problem. In *Third International Conference on Genetic Algorithms*, pages 110–115, 1989.

[21] D. S. Johnson. Local optimization and the traveling salesman problem. In *17th Colloquium on Automata, Languages, and Programming*, pages 446–461, 1990.

[22] A. B. Kahng and B. R. Moon. Toward more powerful recombinations. In *Sixth International Conference on genetic Algorithms*, pages 96–103, 1995.

[23] K. Katayama and H. Narihisa. Iterated local search approach using genetic transformation to the traveling salesman problem. In *Genetic and Evolutionary Computation Conference*, pages 321–328, 1999.

[24] S. Kirkpatrick, C. D. Jr. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[25] S. Lin and B. Kernighan. An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21(4598):498–516, 1973.

[26] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. In *Foundations of Computer Science*, pages 577–591, 1994.

[27] O. Martin, S. W. Otto, and E. W. Felten. Large-step Markov chains for the traveling salesman problem. *Complex Systems*, 5(3):299–326, 1991.

[28] P. Merz and B. Freisleben. Genetic local search for the TSP: New results. In *IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, pages 159–164, 1997.

[29] B. R. Moon, Y. S. Lee, and C. Kim. VLSI circuit partitioner with a new genetic algorithm framework. *Journal of Intelligent Manufacturing*, 9(5):401–412, 1998.

[30] Y. Nagata and S. Kobayashi. Edge assembly crossover: A high-power genetic algorithm for the travelling salesman problem. In *7th International Conference on Genetic Algorithms*, pages 450–457, 1997.

[31] I. Oliver, D. Smith, and J. Holland. A study of permutation crossover operators on the traveling salesman problem. In *Second International Conference on Genetic Algorithms*, pages 224–230, 1987.

[32] J. Y. Potvin. The traveling salesman problem: A neural network perspective. *ORSA Journal on Computing*, 5:328–348, 1993.

[33] W. Smith. *Finding the Optimum N-city Traveling Salesman Tour in the Euclidean Plan in Subexponential Time and Polynomial Space.* Manuscript.

[34] T. Starkweather, S. McDaniel, K. Mathias, D. Whitley, and C. Whitley. A comparison of genetic sequencing operators. In *Fourth International Conference on Genetic Algorithms*, pages 69–76, 1991.

[35] J. Watson, C. Ross, V. Eisele, J. Denton, J. Bins, C. Guerra, D. Whitley, and A. Howe. The traveling salesrep problem, edge assembly crossover, and 2-opt. In *Fifth Conference on Parallel Problem Solving from Nature*, 1998.

[36] D. Whitley, T. Starkweather, and D. Fuquay. Scheduling problems and traveling salesman: The genetic edge recombination operator. In *Third International Conference on Genetic Algorithms*, pages 133–140, 1989.