
Voronoi Quantized Crossover for Traveling Salesman Problem

Dong-II Seo and Byung-Ro Moon

School of Computer Science & Engineering, Seoul National University
Shillim-dong, Kwanak-gu, Seoul, 151-742 Korea
{diseo, moon}@soar.snu.ac.kr

Abstract

It is known that the performance of a genetic algorithm depends on the survival environment and the reproducibility of building blocks. In this paper, we propose a new encoding/crossover scheme that uses genetic distance which explicitly defines the distance between each pair of genes in the chromosome. It pursues both relatively high survival probabilities of more epistatic gene groups and diverse crossover operators for the high creativity of new schemata. The experimental results on benchmark traveling salesman problems showed remarkable improvement in tour cost and running time over state-of-the-art genetic algorithms for the problem.

1 INTRODUCTION

In the context of genetic algorithms (GAs), a specific gene pattern is called schema. Holland showed, by Schema Theorem, that highly fit schemata of short defining lengths and low orders have high survival probabilities in the traditional genetic framework [1]. These short, low-order, high-quality schemata are called *building blocks*. Bui and Moon [2, 3] claimed that, in case of multi-point crossover operators, low-order, high-quality schemata with clustered specific-symbol distributions should serve as building blocks. According to the building block hypothesis, a genetic algorithm seeks near optimal performance through the juxtaposition of building blocks [4]. The performance of a genetic algorithm thus highly depends on its survival environment and reproducibility of building blocks.

In the light of the interrelationship between genes, building blocks are gene groups that have strong *in-*

teractions (or *epistases*) among participating genes in the chromosome. Genes' interaction here means that the contribution of a gene to the chromosomal fitness depends on the values of other genes in the chromosome. The stronger the interactions of genes are, the higher the nonlinearity of the problem is; this makes the problems more difficult [5].

For a given problem representation, the survival probability of a gene group through the crossover is determined by the distribution of genes in the chromosome, while the strength of the epistasis of the gene group is an inherent property of the problem. Therefore, the strategy of locating each gene in the chromosome significantly affects the performance of genetic algorithms. In other words, the positions (or loci) of genes in the chromosome may be placed so as to ensure higher survival probabilities for more epistatic schemata. Inversion is a genetic operator devised for changing the loci of genes dynamically during the genetic process [6, 1]. The efforts to exploit gene positions dynamically are called *linkage learning* [7]. Messy genetic algorithm and fast messy genetic algorithm are examples that implicitly pursue dynamic gene repositioning [8, 9]. The chromosomal encoding with fixed gene positions is called locus-based encoding. A number of studies on *static reindexing* (or *reordering*) of gene positions in locus-based encodings showed performance improvement [10, 11, 12, 13, 14].

The representation power¹ of a genetic algorithm is highly dependent on the *chromosome topology*. That is, the higher the dimension of the chromosome topology is, the higher the representation power it has. A typical chromosome topology is a one-dimensional array. Though one-dimensional array is easy to handle and analyze, it has a poor representation power

¹Here, high representation power means low degree of distortion. Generally, in representing a graph geometrically, the higher the dimension of representation space is, the lower the degree of distortion it shows [15].

which causes great loss of information contained in the problems. To overcome this, the encoding/crossover schemes with multi-dimensional arrays were suggested and remarkable improvements were reported [16, 17, 18, 19]. Recently, Jung and Moon [20, 21] obtained successful results by applying a crossover based on 2D Euclidean encoding to the 2D Euclidean traveling salesman problem (TSP). They used phenotypes themselves for chromosomal cutting.

In a point of view, the studies of changing the loci of genes are to exploit the interactions among genes implicitly. Increasing the representation power of chromosome topologies also can be understood in this context. Jung and Moon’s study may be thought to be an extreme case of such approaches to the 2D Euclidean TSP. We keep the philosophy. In this paper, we suggest a new encoding/crossover scheme which *explicitly* exploits the interactions among genes. In the scheme, the *genic distance* between a pair of genes is defined. We apply this scheme to TSP and compare its performance with state-of-the-art methods.

The rest of this paper is organized as follows. We summarize previous approaches to TSP in Section 2 and explain the proposed genetic operators in Section 3. In Section 4, we provide experimental results. Finally, the conclusion is given in Section 5.

2 TRAVELING SALESMAN PROBLEM

Given n cities, the *traveling salesman problem* (TSP) is the problem of finding the shortest Hamiltonian cycle visiting the cities. More formally, given a set of cities $\{c_1, c_2, \dots, c_n\}$ and the distance $d(c_i, c_j)$ for every pair (c_i, c_j) , it is the problem of finding an ordering π that minimizes the following:

$$C(\pi) = \sum_{i=1}^{n-1} d(c_{\pi(i)}, c_{\pi(i+1)}) + d(c_{\pi(n)}, c_{\pi(1)}).$$

It is a well known NP-hard problem [22]. Thus one should rely on approximation algorithms that do not guarantee optimal solutions.

For decades, TSP has served as an initial proving ground for new problem solving techniques because of its difficulty, applicability, and the simplicity of definition. Various local optimization algorithms such as 2-opt, 3-opt, Lin-Kernighan (LK) algorithm, and their variants were developed and problem independent techniques such as tabu search, simulated annealing, neural networks, genetic algorithms and ant colonies were applied [23, 24].

Recently, hybrid genetic algorithms which combine lo-

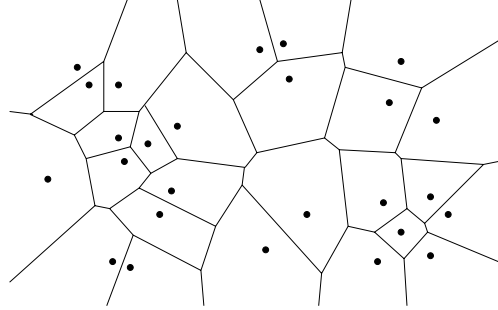


Figure 1: An example of 2D Voronoi regions

```

VQX( $n, k, d_g, p_1, p_2$ )
{
   $I \leftarrow \{1, 2, \dots, n\}; K \leftarrow \{1, 2, \dots, k\};$ 
  Select a subset  $R = \{s_1, s_2, \dots, s_k\} \subset I$ 
  at random;
  for each  $i \in I$  {
     $r[i] \leftarrow \arg \min_{j \in K} \{d_g(s_j, i)\}, s_j \in R;$ 
  }
  for each  $j \in K$  {
     $u[j] \leftarrow 0$  or  $1$  at random;
  }
  for each  $i \in I$  {
    if ( $u[r[i]] = 0$ ) then  $o[i] \leftarrow p_1[i];$ 
    else  $o[i] \leftarrow p_2[i];$ 
  }
  return  $o;$ 
}

```

Figure 2: Voronoi quantized crossover

cal optimization algorithms with the genetic framework have been successfully applied to the problem [25, 26, 20]. LK algorithm [27] is the most popular and powerful local optimization algorithm for TSP. Various crossover operators such as order crossover [28], cycle crossover [29], partially matched crossover [4], edge-recombination crossover [30], and matrix crossover [31] were used for TSP. Distance preserving crossover [32, 26], edge assembly crossover [33, 34, 35], and natural crossover [20, 21] are representative state-of-the-art crossover operators proposed recently.

3 NEW OPERATORS

3.1 Voronoi Quantized Crossover

In Voronoi quantized crossover (VQX), we adapt a chromosome topology in which every gene has a rela-

tive locus determined by the distances between genes, contrary to the others in which every gene has an absolute locus. The distance is called *genic distance*. In a point of view, the chromosome may be thought to be a “complete graph” where each vertex stands for a gene and the edge weight is determined by the epistasis between the two corresponding genes. The graph is directed if the genic distance is asymmetric. By adapting such type of chromosome topology, we aim to represent a chromosome with a minimal degree of distortion.

The name of Voronoi quantized crossover came from Voronoi quantization which is a representative vector quantization method [36]. Vector quantization is a method of approximating arbitrary multi-dimensional vectors by k code vectors each of which represents a subspace of the whole vector space. It is used mostly in data compression. The vector quantization that uses Voronoi regions [37] as the subspaces is called Voronoi quantization. The Voronoi region of a vector is defined to be the nearest neighborhood of the vector. Figure 1 shows an example of Voronoi regions associated with a set of points in 2D Euclidean space.

VQX has a simple structure. Figure 2 shows the pseudo code of VQX where n is the number of genes and k is the crossover degree ranged from 2 to n . The function $d_g : I^2 \rightarrow R$ represents the genic distance. The two parents and the offspring are denoted by p_1 , p_2 and o , respectively. Following the convention, the notation “argmin” takes the argument that minimizes the value. In VQX, the genic space defined by the genic distance d_g is divided into k Voronoi regions determined by the k randomly selected genes, then a sort of block-uniform crossover [17] is performed on the regions.

VQX has two main properties:

- *Convexity* — Voronoi regions are convex² [36]. Therefore, the gene groups of relatively short genic distance have higher survival probabilities than others.
- *Diversity* — It has $\binom{n}{k} 2^k$ crossover operators³.

The first property means that the survival probabilities of gene groups can be controlled by genic distance assignments. We can allow high survival probabilities for building blocks by assigning genic distance in-

²A set $S \in R^k$ is *convex* if $a, b \in S$ implies that $\alpha a + (1 - \alpha)b \in S$ for all $0 < \alpha < 1$.

³In fact, we cannot guarantee that the consequent offspring are all distinct. Different quantizations may generate the same offspring, although believed rare.

versely proportional to the strength of epistasis. The other means that VQX has a lot of crossover operators. The number of crossover operators affects the creativity of new schemata. The number of crossover operators of a typical k -point crossover is $\binom{n-1}{k}$. For $n = 100$, $k = 6$ and $n = 1000$, $k = 8$, for example, k -point crossover has about 10^{10} and 10^{20} crossover operators, respectively, while VQX has about 10^{11} and 10^{22} . However, we should mention that we do not pursue the maximal number of crossover operators.

The time complexity of VQX is $\Theta(kn)$.

3.2 Survival Probabilities

The survival probability of a gene group⁴ (or unspecific schema) in VQX is derived in this section. Given a genic distance measure d_g , a function $h : 2^I \times I \rightarrow Z^+$ is defined as

$$h(S, i) = |\{l \in I : \forall v \in S, d_g(l, v) > d_g(i, v)\}|, S \subset I, i \in I \quad (1)$$

where $I = \{1, 2, \dots, n\}$ and n is the problem size. Given a subset $R = \{s_1, s_2, \dots, s_k\} \subset I$, the Voronoi region assignment function $r : 2^I \times I \rightarrow I$ is defined as

$$r(R, i) = \arg \min_{j \in K} \{d_g(s_j, i)\}, i \in I, s_j \in R \quad (2)$$

where $K = \{1, 2, \dots, k\}$. Now, given S and i , the number of R 's of k elements that make all v 's in S have the same function value $r(R, v) = i$ is $\binom{h(S, i)}{k-1}$. Assuming that the set R is selected at random, the probability that all genes in a gene group S belong to the same region, i.e., the probability that $r(R, j)$'s are the same for all j 's in S , is derived as

$$P_{eq}(S) = \frac{\sum_{i=1}^n \binom{h(S, i)}{k-1}}{\binom{n}{k}}. \quad (3)$$

In the case of $|S| = 2$, the survival probability $P_{sur}(S)$ of a gene group S is derived as

$$\begin{aligned} P_{sur}(S) &= P_{eq}(S) + \frac{1}{2}(1 - P_{eq}(S)) \\ &= \frac{1}{2} + \frac{\sum_{i=1}^n \binom{h(S, i)}{k-1}}{2 \binom{n}{k}}. \end{aligned} \quad (4)$$

This is used in Section 3.4 to examine the relationship between genic distances and survival probabilities.

⁴Generally, a schema is defined by alleles of specific genes. In this paper, we use the term *gene group* rather than schema, because we refer only the set of genes here.

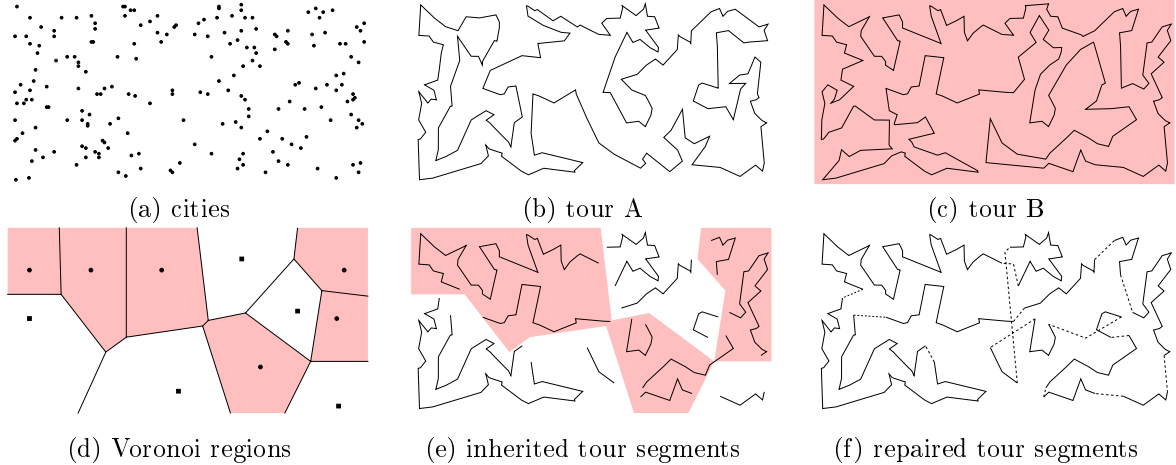


Figure 4: An example VQX for TSP (kroA200)

```

VQX'(n, k, dg, p1, p2)
{
  I ← {1, 2, ..., n}; K ← {1, 2, ..., k};
  Select a subset R = {s1, s2, ..., sk} ⊂ I
  at random;
  for each i ∈ I {
    r[i] ← arg minj ∈ K {dg(sj, i)}, sj ∈ R;
  }
  for each j ∈ K {
    u[j] ← 0 or 1 at random;
  }
  for each i ∈ I {
    if ( u[r[i]] = 0 and u[r[p1[i]]] = 0 )
      then o[i] ← p1[i];
    else if ( u[r[i]] = 1 and u[r[p2[i]]] = 1 )
      then o[i] ← p2[i];
    else o[i] ← nil;
  }
  o ← GreedyRepair(o);
  return o;
}

```

Figure 3: Modified VQX for TSP

3.3 Applying VQX to TSP

In this paper, the locus-based encoding of [11] is used; one gene is allocated for every city and the gene value represents the index of its next city in the tour. Thus, a solution may be thought to be a mapping s from the set of cities I to I . In TSP, a solution that does not construct a Hamiltonian cycle is infeasible. A solution s is feasible if and only if (i) s is one-to-one and (ii) it

has no subcycle. Directly applying the VQX of Figure 2 to TSP may produce infeasible solutions. To avoid this, we need some modification to the crossover.

Figure 3 shows the pseudo code of the modified VQX. The word *nil* is used for the genes whose values are not determined. The consequent solutions have no subcycle but may have genes of *nil* value. In other words, tour segments without any subcycle are created. We use a greedy approach to repair them. In “GreedyRepair()”, a segment is selected and connected to its nearest segment to grow into a complete tour. Note that the performance of repairing is not critical here, as a powerful local optimization heuristic follows the crossover and mutation. Figure 4 shows an example of the crossover process. It is obtained by applying VQX' (in Figure 3) with the genic distance assignment GD1 (described in Section 3.4) to kroA200, a benchmark problem taken from TSPLIB [38]. We use a random tie-breaking in applying the equation (2) in the crossover.

3.4 Genic Distance Assignments

To apply VQX to TSP, the distances among genes (genic distances) are needed. The genic distances may be assigned statically or dynamically in the genetic process. In this paper, the static assignment is used.

Intuitively, an ideal value of a genic distance is a value inversely proportional to the epistasis. This leads to the high survival probabilities of relatively more interactive gene groups because the survival probability of a gene group is inversely proportional to their genic distances in VQX. However, no practical method is known yet for exactly computing epistases; two heuristics are used in this paper. Let I be a set of city indices

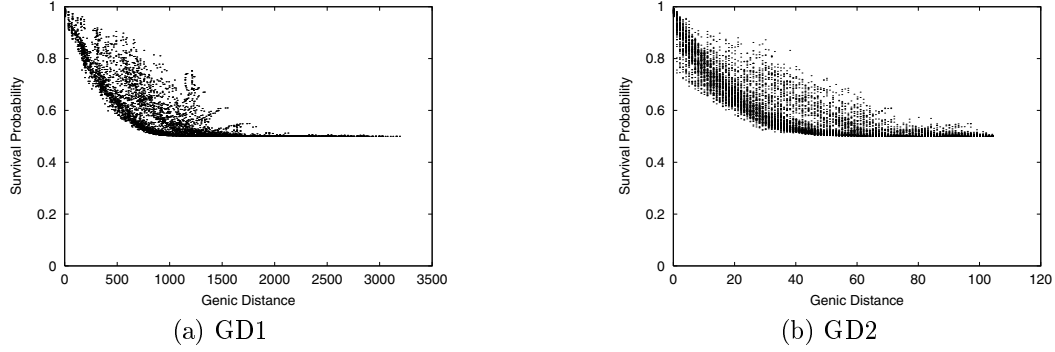


Figure 5: Survival probability versus genic distance (lin105)

and $d(i, j)$ is the distance from city $i \in I$ to city $j \in I$. The genic distance $d_g(i, j)$ from gene i to gene j is defined in two manners as

- GD1: $d_g(i, j) = d(i, j)$
- GD2: $d_g(i, j) = |\{l : d(i, l) < d(i, j), l \in I\}|$.

In GD1, the Euclidean distance itself is used; in GD2, the number of cities closer to city i than city j is used. Usually, d_g is asymmetric in GD2, while it is symmetric as far as d is symmetric in GD1.

Figure 5 shows the results of a simple test to observe the relationship between the genic distance and the survival probability of a gene pair. The horizontal and vertical axes of coordinates represent the genic distance and the survival probability, respectively. The equation (4) was used to acquire the survival probabilities from the genic distances obtained by applying GD1 and GD2 to lin105, an instance from TSPLIB. It shows that the survival probabilities of a close gene pairs in the genic space are high in both cases.

3.5 Heterogeneous Mating

In a preliminary examination, VQX showed faster convergence than the other crossovers in comparison; this may cause the premature convergence of the genetic algorithm. To avoid this, we use a method of mating mutually dissimilar individuals in parallel with VQX. Hollstien called this type of breeding a negative assortive mating [39]. There are various methods, sometimes called niching methods, for maintaining population diversity [40, 41].

Figure 6 shows the pseudo code of the mating used in this paper. First, m individuals are selected from the population P by roulette-wheel selection. Then the most different one from p_1 among them is selected

```

MateSelection( $P, m, p_1$ )
{
   $C \leftarrow \emptyset$ ;
  for  $i \leftarrow 1$  to  $m$  {
     $c \leftarrow \text{Selection}(P \setminus (\{p_1\} \cup C))$ ;
     $C \leftarrow C \cup \{c\}$ ;
  }
   $p_2 \leftarrow \arg \max_{c \in C} \{\text{distance}(p_1, c)\}$ ;
  return  $p_2$ ;
}

```

Figure 6: Heterogeneous mate selection

as p_2 . Hamming distance⁵ is used for the distance function “distance()”.

4 EXPERIMENTAL RESULTS

The genetic algorithm used in this paper is a steady-state hybrid genetic algorithm. Figure 7 shows the template. In the template, n is the problem size, m is the group size in mating, k is the crossover degree, and d_g is the genic distance measure. The two selected parents and the offspring are denoted by p_1 , p_2 and o , respectively. The genetic operators and their parameters used in this paper are summarized in the following:

- Population Initialization — Initial solutions are generated at random.
- Population Size — $|P| = 100$.
- Selection — Roulette-wheel selection. The fitness

⁵the number of different edges between two tours.

Table 1: Experimental results of VGA1 and VGA2

Graph (opt)	Xover	OB #	Best (%)	Avg (%)	σ/\sqrt{t}	Gen	Time (s)
att532 (27686)	VGA1	97	27686 (0)	27686.37 (0.001)	0.22	3006	103
	VGA2	95	27686 (0)	27686.69 (0.002)	0.30	3024	109
dsj1000 (18659688)	VGA1	24	18659688 (0)	18659952 (0.001)	24	2803	1026
	VGA2	52	18659688 (0)	18659809 (0.001)	13	3470	1251
d2103 (80450)	VGA1	72	80450 (0)	80470.05 (0.025)	3.22	3874	1084
	VGA2	76	80450 (0)	80467.07 (0.021)	3.04	4271	1157
pcb3038 (137694)	VGA1	11	137694 (0)	137707.22 (0.010)	1.40	12234	835
	VGA2	8	137694 (0)	137706.78 (0.009)	1.22	13021	906
fnl4461 (182566)	VGA1	0	182573 (0.004)	182607.22 (0.023)	1.97	28518	2011
	VGA2	0	182571 (0.003)	182605.88 (0.022)	2.21	28992	2057

Table 2: Comparison of VGA with DGA, EGA, and NGA

Graph (opt)	Xover	Best (%)	Avg (%)	σ/\sqrt{t}	Gen	Time (s)
att532 (27686)	DGA	27686 (0)	27692.86 (0.025)	0.75	3971	89
	EGA	27686 (0)	27700.51 (0.052)	0.84	13934	271
	NGA	27686 (0)	27692.13 (0.022)	0.77	3563	167
	VGA2	27686 (0)	27686.69 (0.002)	0.30	3024	109
dsj1000 (18659688)	DGA	18659688 (0)	18660087 (0.002)	78	11267	1038
	EGA	18659688 (0)	18679325 (0.105)	1494	41938	1867
	NGA	18659688 (0)	18659942 (0.001)	18	3266	1114
	VGA2	18659688 (0)	18659809 (0.001)	13	3470	1251
d2103 (80450)	DGA	80450 (0)	80500.09 (0.062)	5.86	4021	630
	EGA	80450 (0)	80469.82 (0.025)	2.29	82072	8466
	NGA	80450 (0)	80472.05 (0.027)	4.89	1970	456
	VGA2	80450 (0)	80467.07 (0.021)	3.04	4271	1157
pcb3038 (137694)	DGA	137699 (0.004)	137751.44 (0.042)	4.24	20261	1408
	EGA	137694 (0)	137831.77 (0.100)	7.26	199015	28213
	NGA	137698 (0.003)	137733.10 (0.028)	3.71	20582	1734
	VGA2	137694 (0)	137706.78 (0.009)	1.22	13021	906
fnl4461 (182566)	DGA	182593 (0.015)	182822.39 (0.140)	31.80	76331	13728
	EGA	182598 (0.018)	182864.60 (0.164)	31.11	338860	160845
	NGA	182572 (0.003)	182631.82 (0.036)	3.19	84247	8832
	VGA2	182571 (0.003)	182605.88 (0.022)	2.21	28992	2057

value f_i of the solution i is calculated as

$$f_i = (C_w - C_i) + (C_w - C_b)/4 \quad (5)$$

where C_i , C_w , and C_b are the costs of the solution i , the worst solution, and the best solution in the population, respectively. The fitness value of the best solution is five times as great as that of the worst solution in the population.

- Group Size for Mating — $m = 5$.
- Crossover Degree — An empirical value $k = \lfloor \ln n + \frac{1}{2} \rfloor + 2$ is used where n is the problem size and “ln” is the natural logarithm.
- Mutation — Double-bridge kick move [27] was applied once per ten offsprings. Figure 8 shows a symbolic drawing of double-bridge kick move.

- Local Optimization — LK algorithm accelerated by don't-look bit [42] and segment tree [43] was used.

- Replacement — A variant of preselection [44] was used as in [11]. Each offspring is replaced with (i) its more similar parent if the offspring is better, (ii) the other parent if the offspring is better, (iii) the worst solution in the population, otherwise.

- Stop Condition — Until 70 percent of the population converge with the same cost as the best solution in the population. This takes account of the cases that more than one best solution of the same quality competes with each other.

The algorithms were implemented in C on Intel Pentium III 866 MHz running Linux 2.2.14.

```

VGA( $n, m, k, d_g$ )
{
  Initialize population  $P$ ;
  repeat {
     $p_1 \leftarrow \text{Selection}(P)$ ;
     $p_2 \leftarrow \text{MateSelection}(P, m, p_1)$ ;
     $o \leftarrow \text{VQX}'(n, k, d_g, p_1, p_2)$ ;
     $o \leftarrow \text{Mutation}(o)$ ;
     $o \leftarrow \text{LocalOptimization}(o)$ ;
     $P \leftarrow \text{Replacement}(P, p_1, p_2, o)$ ;
  } until (stop condition);
  return the best of  $P$ ;
}

```

Figure 7: Steady-state hybrid genetic algorithm for TSP

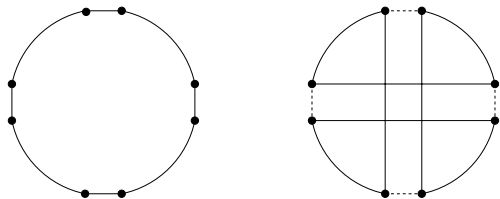


Figure 8: Double-bridge kick move

Table 1 compares two different versions of VQX. VGA1 and VGA2 represent the genetic algorithms using Voronoi quantized crossover with the genic distance assignments GD1 and GD2 (described in Section 3.4), respectively. In the table, the frequency of finding the optimal solutions (OB#), the best tour cost (Best), average tour cost (Avg), group standard deviation (σ/\sqrt{t}), average generation (Gen), and average running time (Time) over 100 ($= t$) runs are presented on att532, dsj1000, d2103, pcb3038, and fnl4461, problem instances from TSPLIB [38]. The parentheses after best and average tour costs represent the percentages above optima. VGA2 performed better than or equal to VGA1 for all instances except att532 in average cost. Distances between cities were computed in double precision mode and rounded to integer to remove uncertainties.

Table 2 compares the performance of VQX with other state-of-the-art crossovers. DGA, EGA, and NGA represent the genetic algorithms using distance preserving crossover [32, 26], edge assembly crossover [33], and natural crossover [20, 21], respectively. The results of DGA, EGA, and NGA in the table are quoted from [21] in which the same LK implementation as ours was

Table 3: Comparison of VGA with FCGA

Graph (opt)	Xover	OB #	Avg	Gen	Time (s)
eil101 (629)	FCGA	50	629.0	15	2
	VGA2	50	629.0	6	1
lin318 (42029)	FCGA	50	42029.0	49	60
	VGA2	50	42029.0	221	29
pcb442 (50778)	FCGA	50	50778.0	39	233
	VGA2	50	50778.0	739	45
att532 (27686)	FCGA	23	27691.3	66	304
	VGA2	47	27686.9	3199	159
rat575 (6773)	FCGA	43	6773.2	55	500
	VGA2	19	6773.8	2720	53
u724 (41910)	FCGA	41	41912.3	50	845
	VGA2	50	41910.0	3089	154

used. VGA2 outperformed the others for all instances in average cost. For att532 and dsj1000, VGA2 consumed comparable running time to NGA. But their growth rates of time consumption with respect to the problem size were much lower than NGA. Thus, the speed of VGA2 for large problems pcb3038 and fnl4461 was much faster than the others. The overall results show that Voronoi quantized crossover is the most attractive among them. They also imply that GD1 and GD2 for the genic distance assignment are reasonable.

Table 3 compares the performance of VGA with FCGA [35]. FCGA stands for family competition genetic algorithm which is a combination of the family competition, near 2-opt and edge assembly crossover [33]. In the table, results over 50 runs are presented on eil101, lin318, pcb442, att532, rat575, and u724 from TSPLIB. These six instances are all those available for comparison in [35]. The results of FCGA in the table are quoted from [35]. The running time is the normalized value for Intel Pentium III 600 MHz. The average tour costs of VGA2 were better than or equal to FCGA for all instances except rat575. It is notable that the speed of VGA2 was much faster than FCGA. (In Table 2, EGA, the ancestor of FCGA, took 80 times more than VGA2 for the instance with 4461 cities.)

5 CONCLUSIONS

In this paper, we proposed a new crossover operator, named Voronoi quantized crossover (VQX), that utilizes the explicit genic distances. This allows us to exploit the interactions among genes explicitly. VQX has two main properties of convexity and diversity. These properties are believed to help improve the performance of genetic algorithms by encouraging the survival probability and the reproducibility of high-quality building blocks in the genetic process. The

experimental results supported this.

VQX may be applied to other combinatorial optimization problems than the traveling salesman problem. Of course, a measure for genetic distances must be devised for each problem. Future studies include extending VQX to various problems.

ACKNOWLEDGMENTS

The authors would like to thank Soonchul Jung for invaluable discussions on various ideas reported in this paper. This work was partly supported by SNU Statistical Research Center for Complex Systems and Brain Korea 21 Project. The RIACT at Seoul National University provided research facilities for this study.

REFERENCES

- [1] J. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, 1975.
- [2] T.N. Bui and B.R. Moon. Analyzing hyperplane synthesis in genetic algorithms using clustered schemata. In *Parallel Problem Solving from Nature*, pages 108–118. 1994.
- [3] T.N. Bui and B.R. Moon. GRCA: A hybrid genetic algorithm for circuit ratio-cut partitioning. *IEEE Transactions on CAD*, 17(3):193–204, 1998.
- [4] D.E. Goldberg. *Genetic Algorithms in Search, Optimization, Machine Learning*. Addison-Wesley, 1989.
- [5] S.A. Kauffman. Adaptation on rugged fitness landscapes. In D.L. Stein, editor, *Lectures in the Sciences of Complexity*, pages 527–618. Addison-Wesley, 1989.
- [6] J.D. Bagley. *The Behavior of Adaptive Systems which Employ Genetic and Correlation Algorithms*. PhD thesis, University of Michigan, 1967.
- [7] G.R. Harik. *Learning Gene Linkage to Efficiently Solve Problems of Bounded Difficulty Using Genetic Algorithms*. PhD thesis, University of Michigan, 1997.
- [8] D.E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems*, 3(5):493–530, 1989.
- [9] H. Kargupta. *SEARCH, Polynomial Complexity, and the Fast Messy Genetic Algorithm*. PhD thesis, University of Illinois at Urbana-Champaign, 1995.
- [10] T.N. Bui and B.R. Moon. Hyperplane synthesis for genetic algorithms. In *5th International Conference on Genetic Algorithms*, pages 102–109, 1993.
- [11] 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.
- [12] T.N. Bui and P. Eppley. A hybrid genetic algorithm for the maximum clique problem. In *6th International Conference on Genetic Algorithms*, pages 478–484, 1995.
- [13] T.N. Bui and B.R. Moon. Genetic algorithm and graph partitioning. *IEEE Transactions on Computers*, 45(7):841–855, 1996.
- [14] O.T. Sehitoglu and G. Üçoluk. A building block favoring reordering method for gene positions in genetic algorithms. In *Genetic and Evolutionary Computation Conference*, pages 571–575, 2001.
- [15] 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.
- [16] J. Cohoon and D. Paris. Genetic placement. In *IEEE International Conference on Computer-Aided Design*, pages 422–425, 1986.
- [17] C. Anderson, K. Jones, and J. Ryan. A two-dimensional genetic algorithm for the Ising problem. *Complex Systems*, 5:327–333, 1991.
- [18] T.N. Bui and B.R. Moon. On multi-dimensional encoding/crossover. In *6th International Conference on Genetic Algorithms*, pages 49–56, 1995.
- [19] A.B. Kahng and B.R. Moon. Toward more powerful recombinations. In *6th International Conference on Genetic Algorithms*, pages 96–103, 1995.
- [20] S. Jung and B.R. Moon. The natural crossover for the 2D Euclidean TSP. In *Genetic and Evolutionary Computation Conference*, pages 1003–1010, 2000.
- [21] S. Jung and B.R. Moon. Toward minimal restriction of genetic encoding and crossovers for the 2D Euclidean TSP. *IEEE Transactions on Evolutionary Computation* (conditionally accepted).
- [22] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, 1979.

- [23] D.S. Johnson and L.A. McGeoch. The traveling salesman problem: A case study in local optimization. In *Local Search in Combinatorial Optimization*, pages 215–310. John Wiley & Sons, 1997.
- [24] G. Reinelt. *The Traveling Salesman: Computational Solutions for TSP Applications*. Springer-Verlag, 1994.
- [25] 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.
- [26] P. Merz and B. Freisleben. Genetic local search for the TSP: New results. In *IEEE Conference on Evolutionary Computation*, pages 159–164, 1997.
- [27] S. Lin and B. Kernighan. An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21:498–516, 1973.
- [28] L. Davis. Applying adapting algorithms to epistatic domains. In *9th International Joint Conference on Artificial Intelligence*, pages 162–164, 1985.
- [29] 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.
- [30] 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.
- [31] A. Homaifar, S. Guan, and G. Liepins. A new approach on the traveling salesman problem by genetic algorithms. In *5th International Conference on Genetic Algorithms*, pages 460–466, 1993.
- [32] B. Freisleben and P. Merz. New genetic local search operators for the traveling salesman problem. In *Parallel Problem Solving from Nature*, pages 890–900. 1996.
- [33] Y. Nagata and S. Kobayashi. Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem. In *7th International Conference on Genetic Algorithms*, pages 450–457, 1997.
- [34] 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 *Parallel Problem Solving from Nature*, pages 823–834. 1998.
- [35] H.K. Tsai, J.M. Yang, and C.Y. Kao. A genetic algorithm for traveling salesman problems. In *Genetic and Evolutionary Computation Conference*, pages 687–693, 2001.
- [36] A. Gersho and R.M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.
- [37] G.F. Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième Memoire: Recherches sur les paralléloèdres primitifs. *Journal für Reine und Angewandte Mathematik*, 134:198–287, 1908.
- [38] TSPLIB. <http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/>.
- [39] R.B. Hollstien. *Artificial Genetic Adaptation in Computer Control Systems*. PhD thesis, University of Michigan, 1971.
- [40] D.E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Second International Conference on Genetic Algorithms*, pages 41–49, 1987.
- [41] S.W. Mahfoud. *Niching Methods for Genetic Algorithms*. PhD thesis, University of Illinois at Urbana-Champaign, 1995.
- [42] J.L. Bentley. Experiments on traveling salesman heuristics. In *First Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 91–99, 1990.
- [43] M.L. Fredman, D.S. Johnson, L.A. McGeoch, and Ostheimer G. Data structures for traveling salesmen. *Journal of Algorithms*, 18:432–479, 1995.
- [44] D. Cavicchio. *Adaptive Search Using Simulated Evolution*. PhD thesis, University of Michigan, 1970.