# A Local Search Algorithm Based on Genetic Recombination for Traveling Salesman Problem

Peng Gang[1], Ichiro Iimura[2], Hidenobu Tsurusawa[3], and Shigeru Nakayama[4]

[1] Department of Computer and Control Engineering,
Oita National College of Technology,
1666 Maki, Oita City, 870-0152, Japan
peng@oita-ct.ac.jp

[2] Prefectural University of Kumamoto,
3-1-100 Tsukide, Kumamoto 862-8502 Japan
iiimura@pu-kumamoto.ac.jp

[3] Department of Computer and Control Engineering,
Oita National College of Technology,
1666 Maki, Oita City, 870-0152, Japan
tsurusawa@oita-ct.ac.jp

[4] Department of Information and Computer Science,
Faculty of Engineering, Kagoshima University
1-21-40 Korimoto, Kagoshima, 890-0065, Japan
shignaka@ics.kagoshima-u.ac.jp

**Abstract.** Genetic Algorithms(GAs) have been applied in many different fields and optimization problem domains. It is well known that it is hard to solve the complex problems with a Simple Genetic Algorithm (SGA). Many previous studies have shown that the hybrid of local search and GAs is an effective approach for finding near optimum solutions to the traveling salesman problem (TSP). In this paper, an approach based on the Genetic Recombination is proposed and applied to the TSP. The algorithm is composed of two SGAs which only consist of the basic genetic operators such as selection, crossover and mutation. One of the SGAs is named as the Global Genetic Algorithm (GGA) and carried out in the main tours which are designed for searching the global optimal solutions. Another one is named as the Local Genetic Algorithm (LGA) and carried out in the sub tours which are designed for searching the local optimal solutions. The LGA is combined to the GGA as an operator. The local optimal solutions are recombined to the main tours for improving the search quality. To investigate the features of the proposed algorithm, it was applied to a small double circles TSP and some interesting results were presented in our experiments.

## 1 Introduction

The TSP is one of the well-studied combinatorial optimization problems [6], [18]. Many researchers from various fields have devoted to developing new algorithms for solving it. In the TSP, each distance between two cities is given for a set of $n$ cities. The goal is to find the shortest tour. There are currently three general classes of heuristics for the TSP: classical tour construction heuristics such as the Nearest Neighbor method, the Greedy algorithm and local search algorithms based on re-arranging segments of the tour[15]. Many progressive results have been presented in the previous studies during the recent years even though there are still improvable spaces with the search algorithms which have been applied to the TSP, such as ant colonies [12], local search[5], neural networks [10], simulated annealing [17], and tabu search[2], genetic algorithms[9]. It has been proved that the hybrid of different algorithms is more effective than a single algorithm. For example, the local search has been successful for improving GAs in the search processes[1],[16], [19]. Most of the works on solving the TSP focused on the efficiency on how to solving the larger TSP instances. Some of the works aimed at expanding the theories of search algorithms, especially in the GAs domain.

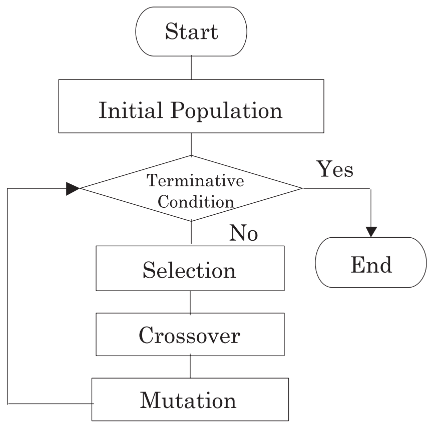


**Fig. 1.** Simple GA

In this paper, our attempt is to make a discussion on the GAs based on Genetic Recombination. The algorithm is composed of two SGAs which only consist of the basic genetic operators. One is the GGA which was applied to the main tours for searching the global optimal solutions. Another is the LGA which was applied to the sub tours for searching the local optimal solutions. The SGA was developed by John Holland and his original algorithm is approximately the same as shown in the Fig.1. [8]. In the early studies, the SGA played an important role in the development of GAs and attracted the researchers' attentions widely. Goldberg made a detailed discussion on how the SGA works with some simple optimal mathematical functions and other problems in his book[4]. Reeves discussed the differences and similarities between the SGA and the neighborhood search [3]. Computer scientist Michael D. Vose provided an introduction to what is known about the theory of the SGA. He also made available algorithms for the computation of mathematical objects related to the SGA[13]. All the studies have shown that the SGA is still valuable in heuristic search algorithm.

The paper is organized as follows. Section 2 gives a general description of the local search to the TSP. Section 3 presents the detailed operations of the proposed algorithm. Section 4 gives the analyses on the experimental results. Section 5 is the summary for this paper.

## 2 Local Search Algorithms in GAs

Every successful strategy to produce near-optimal solutions necessarily relies upon the local search algorithm. All these algorithms differ with respect to their neighborhood structures. Any such structure specifies a set of neighboring solutions that are in some sense close to that solution. The associated local improvement operator replaces a current solution by a neighboring solution of better value if possible. The local search algorithm is repeated several times, retaining the best local optimum found. Our works focus on the local search algorithms. In the hybrid of the Local Search and the GAs, the design of the local operator is very important. There are many local search heuristics which have been combined to the GAs for the TSP. For example, the well-known 2-Opt heuristic has been used to optimize the TSP tours in connection with the GAs [18]. The 2-Opt removes two edges in a tour, and then one of the resultant segments is reversed and the two segments are reconnected. If 2-Opt results in an improved tour, the change is preserved. Otherwise, the tour is returned to the original form. The 2-Opt used a pair of edges which was formed from 4 cities in the tour. It is a powerful local search operator in the GAs. Compared with the 2-Opt, K-Opt (k=3,4. . . ) uses more edges to rearrange the tours.

Moreover, some crossover methods also emphasis on the local operations. The crossover methods usually recombine two individuals to reproduce the new individuals. But most of them seldom utilize the shorter edges while they are applied to the tour. Some crossover methods aimed at the phenotype of the edges in the tour. They use more cities or a set of cities according to the visiting order in the tour. For example, YAMAMURA proposed the sub tour exchange crossover (SXX) which mutually exchanges the parts which contain the same continuous cities in two individuals. [14]. MAEGAWA proposed the edge exchange crossover (EXX) which utilizes the edges from different individuals [11]. Both of the methods are powerful for solving the TSP. More cities are used when they are applied to re-arrange the tours. However, there is a problem with few variations of the child individuals because the SXX is only available when the parts which composed of the same cities in two various individuals are found. The EXX puts emphasis on combining edges.

Our approach is to use a sub tour which contains a set of continuous cities according to the visiting order. The basic idea is to find a better sub tour to replace the original one. This operation acts like the Genetic Recombination in the Genetic Engineering field. Genetic Recombination is the process by which the combination of genes in an organism's offspring is different from the combination of genes in that organism. This definition is commonly used in classical genetics, evolutionary biology, and population genetics. Commonly, one gene or a set of a few foreign genes is taken out of the DNA of one organism and inserted into the DNA of another organism by an artificial manipulation of genes. The manipulation disrupts the ordinary command code sequence in the DNA. This disruption may make the individual better if it is applied judiciously. As we knew, there are many good phenotypes of plants that have been created in biological field with the Genetic Recombination. The technical problem is how to find the better GENES to replace the original ones. In the TSP, it is the technique related to the local search. To find the better cities in the tour, we chose a set of continuous cities from the main tour to form a sub tour, and then applied the LGA to the sub tour to find better solutions to feed into the main tour. This operation acts as the cultivation for finding the better GENES biologically. The detailed operations are described in the following section.

## 3 New Algorithm

The new algorithm is shown in Figure 2. The parts a and b in the figure are separately a SGA before the part b is implemented to the part a. The part b is the LGA in the algorithm. The two parts form the new algorithm–GGA. Part b performs as a local search operator in the GGA. The GGA is applied to the main tour for searching the global optimal solutions. To obtain the global optimal solutions in the process of the TSP, the GGA may just need to improve the order of a set of cities in the best tour with evolution of the process, especially in the latter stages of the process. We chose a set of continuous cities to create the sub tour in the experiment. The LGA was applied to the sub tour for finding the local optimal solution to replace the original part chosen from the main tour.

There are two methods for initializing the sub tour: 1. Keep the original part from the main tour as an individual in the population of the LGA, and then randomly initialize the sub tour except for the start and end cities to the population size of the LGA. 2. Only reproduce the original part from the main tour to the population size of the LGA. It means the population is composed of same individuals as the original part. The same operation in the both initialization methods 1 and 2 is that the original parts from the main tours are preserved in the population of the LGA. If the solution of the LGA is shorter than the original part, the main tour will be improved when it is recombined. If no solution is found shorter than the original part in the LGA, the original part will be recombined back to the main tour. Both of the initialization methods are available for a small sub tour in the LGA. But, the first method is not suggested for a big sub tour as it takes long time to initialize.

In the LGA, the sub tour is an open tour. The length of the tour is calculated from the start city to the end city, not including the distance between the end city and the start city. Another important point is that all individuals in the population of the LGA have the same start and end cities during the process-

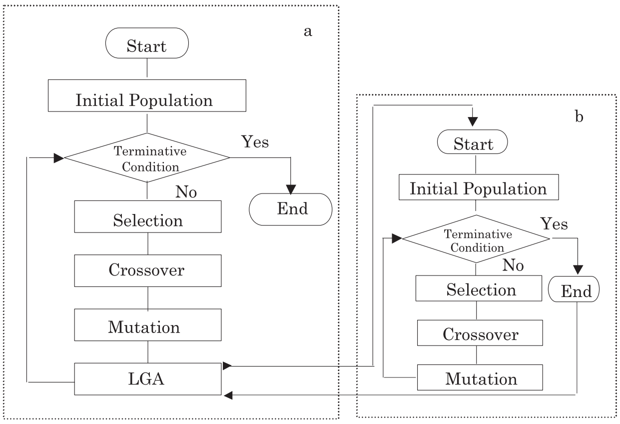


**Fig. 2.** New Algorithm

ing. This operation is for avoiding that the main tour becomes longer at the connection points after the sub tour was recombined back into it.

Because both the GGA and the LGA are complete genetic algorithms, all genetic operators for solving the TSP are probably applicable to them. In our experiments, only three genetic operators were implemented: Crossover, Mutation and Selection. The crossover is one point crossover method with two random individuals. The mutation is randomly carried out to the individuals by changing the order of two cities chosen randomly in one tour. The selection is carried out to replace the four longest individuals with the two shortest ones in the population. In the LGA, the mutation and crossover keep the diversity of the population, so the mutation rate is set higher than in the GGA. The terminative conditions are set to the total number of generations in the GGA and the LGA.

Obviously, it is computationally expensive running a big LGA in every generation of the GGA. But, it is believed that the LGA could be set as an island for improving the elite individuals of the main population in distributed or parallel processing. (This will be discussed in our future works). A small LGA is effective as an operator in the GGA. This has been confirmed in our experiments.

***Local Operations:*** The tour which consists of $n$ cities is expressed as $c$=$\{c_0,...,c_i,c_j,..., c_{n-1}\}$. The distance $d(c_i,c_j)$ is given for the pair of cities $c_i$ and $c_j$. All cities are coded using path representation.

1. Randomly chosen one main tour $c$=$\{c_0, ..., c_i, ..., c_j, ..., c_{n-1}\}$ from Population of the GGA.
2. Randomly chosen one sub tour which contains $N_s$ continuous cities in the main tour $c$. The sub tour $c_s$=$\{c_i, ..., c_j\}$. The start city is $c_i$ and the end city is $c_j$. j-i$\geq$4. This is the first individual of the LGA. The length of the first individual was set as $d_0$.

3. Reproduced the first individual to the population size of the LGA. The start city $c_i$ and the end city $c_j$ were kept unchanged. The population of the LGA was created with $P_s$ same individuals.
4. Run the LGA in every generation of the GGA. The start and end cities of all individuals of the LGA were kept unchanged during the processing.
5. The best individual in the population of the LGA was obtained when the LGA stopped. Its length was set as $d_1$.
6. The best individual was recombined back to the main tour to replace the original part.

Here we set the Ratio $R$=$d_0/d_1$.The range of $R$ is $R\geq$ 1. The $R$ is discussed in the next section. The population size $P_s$ of the LGA ranges from 4 to $n$. The minimum size of the sub tour is set to 4 cities due to the start and end cities are fixed in the sub tour during the processing. The maximum size is set to $n$ because it is the largest improvable parts in the main tour. The LGA performs as the reversion operator in the GGA when the sub tour only contains 4 cities, the start and end cities were fixed.

## 4 Results and Discussions

The tour which contains $n$ cities $c = \{c_0, ..., c_i, ..., c_j, ..., c_{n-1}\}$ is given and the distance between two cities $c_i$,$c_j$ is $d(c_i,\ c_j)$. Total distances of the main tour and the sub tour are $d_m$ and $d_s$ respectively:

$$d_m=\sum_{i=0}^{n} d(c_i,\ c_{i+1}) \tag{1}$$

Where $c_n$=$c_0$ .

$$d_s=\sum_{i=0}^{n-1} d(c_i,\ c_{i+1}) \tag{2}$$

The $d_m$ and $d_s$ are defined as the Fitness of the individuals. The shorter the distance is, the higher the Fitness is.

The parameters are shown in the table 1. Our source code is written in the Java and run on a PC (CPU Pentium 1.0GHz, 256 MB memory) with Windows 2000 Operating System.

**Table 1.** Parameters in the GGA and the LGA

| | GGA(Main Tour) | LGA(Sub Tour) |
|---|---|---|
| Population size: | 100 | 80 |
| Cities | 24 | 4∼ 24 |
| Generations | 1000 | 0,10,20,30,40,50,60,70,80,90,100 |
| Crossover Rate | 75% | 75% |
| Mutation Rate | 2% | 2.5% |
| Selection | 2 shortest individuals replace the 4 longest ones | 2 shortest individuals replace the 4 longest ones |

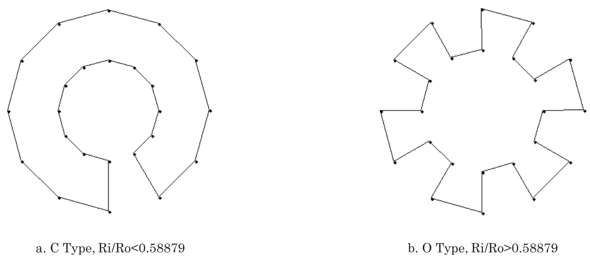


**Fig. 3.** TSP instances

The TSP instance of the double concentric circle which contains only 24 cities was used in the experiments(Fig.3.). The ratio of the inner radius(Ri) and the outer radius(Ro) is $r$=Ri/Ro. If $r < 0.58879$, the optimal tour is C type (Fig.3.a.). If $r \geq 0.58879$, the optimal tour is O type(Fig.3.b.).

### 4.1 Distributions of Optimal Solutions

Fig.4. shows the numbers of the optimal solutions which were obtained in 20 runs. A peak appeared around the half city numbers of main tour. The Fitness is shown in Fig.5. The process stopped at the 1000th generation. The process converged faster with the increase of city numbers in the sub tour at the beginning. This happened because there are much more improvable spaces in a big sub tour in the LGA. The stable convergences appeared when the city numbers of the sub tour range around the half that of the main tour. The result is not satisfied when the LGA performs as the reversion operator in the experiments.

### 4.2 Computation Time

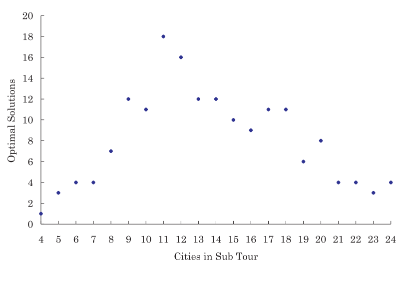


**Fig. 4.** Distribution of optimal solutions.

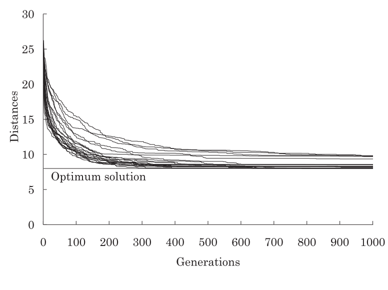


**Fig. 5.** Fitness.

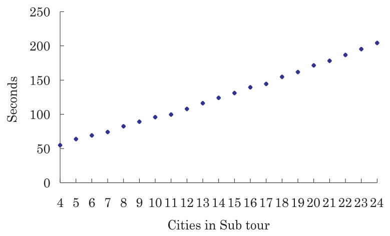


**Fig. 6.** Computation time with cities increase

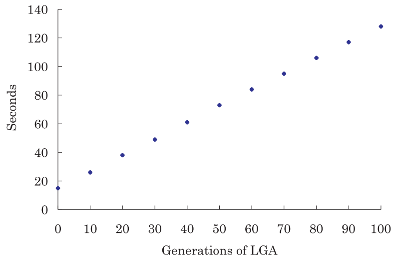


**Fig. 7.** Computation time with generations increase.

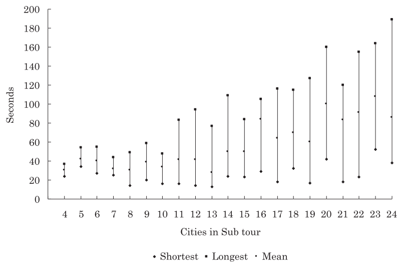


**Fig. 8.** Time of the optimal solution

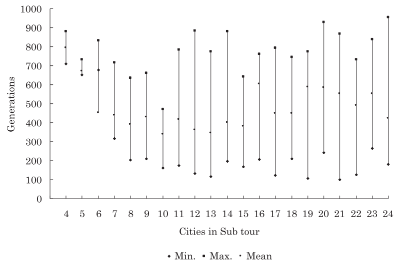


**Fig. 9.** Generation of optimal solutions.

The GAs usually take a long time to run to reach a good result. Consequently it increases the computation time greatly by combining the LGA to the GGA. The computation time was examined by increasing the city numbers and the generations in the LGA respectively. The results are shown in Fig.6. and Fig.7. The computation time linearly became longer with the increase of the city numbers and the generations in the LGA. Less city numbers and generations in the LGA took shorter time but it might not be sufficient for finding a better sub tour to improve the main tour.

### 4.3 Time and Generations for obtaining the Optimal Solutions

Fig.8. is the figure of the computational time for obtaining the optimal solutions. There are 3 points on every vertical line . The low, upper and central points show the shortest, longest and mean time for obtaining the optimal solutions respectively in 20 runs. The time became longer for finding the optimal solution while the city numbers of the sub tour increased. In Fig.9., the three points on the lines show the earliest, latest and mean generations for reaching the optimal solutions respectively in 20 runs. The earliest generations appeared when the city numbers in the sub tour ranged around the half city numbers of the main tour. Some optimal solutions appeared early when the sub tours contained more cities.

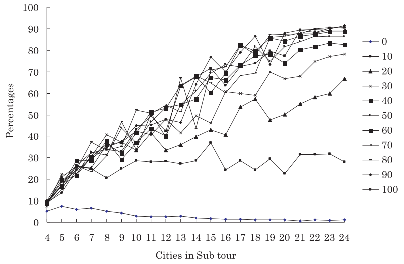


**Fig. 10.** Rates, Generations and population sizes.

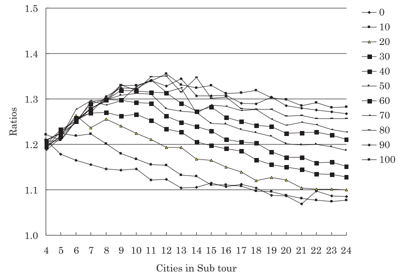


**Fig. 11.** Ratios, Generations and population sizes.

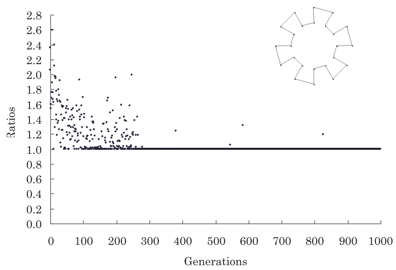


**Fig. 12.** Generation of Optimal Solution: 252nd

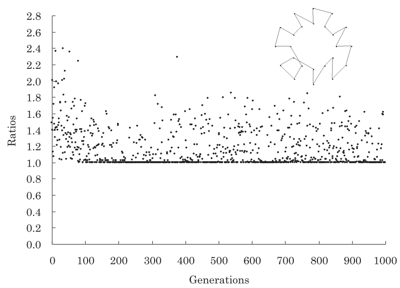


**Fig. 13.** Generation of Local Solution : 151st

But, it took much longer computation time when the city numbers increased in the sub tour as mentioned in 3.1 and 3.2..

### 4.4 Recombination Rates and Ratios

Fig.10. is the figure of the recombination rates. We suppose the number of the individuals which were recombined with the LGA is $N_r$ and the number of the individuals which were searched by the LGA is $N_s$. The Recombination Rate $=N_r/N_s\times$ 100. The Recombination Rates increased with the increase of the generations and the city numbers in the sub tours. More generations are advantageous for finding a better sub tour to carry out the recombination, but it takes longer time to run the LGA. Same trend appeared with the changes of the city numbers because there are more improvable spaces in the bigger sub tour. The part chosen from the main tour is easier to improve. Fig.11. is the Ratio figure. The Ratio $R$=$d_0/d_1$ and $R\geq$1. The peaks of the ratios appeared when the city numbers in the sub tour ranged from 8 to 13. The higher Ratio means a sub tour was found to be shorter in the LGA than the original part from the main tour. The result is correlative with the distribution of the optimal solutions in Fig.4.

### 4.5 Dynamics of the LGA

Fig.12. and Fig.13. show the dynamics of the LGA. Fig.12. presents the Ratio distributions of a single GGA in which the optimal solution was obtained at the 252nd generation. Because the original part from the GGA was preserved in the LGA and the Ratio $R{=}d_0/d_1$, the value of the $R\geq 1$. It indicates that the LGA created a shorter sub tour and recombined it to the GGA when $R >$ 1. The LGA worked efficiently and more Ratios are bigger than 1 before the optimal solution was obtained at the 252nd generation. When the population converged to the optimal solution, the LGA could not create a better sub tour than the original part from the main tour and the Ratios became 1. Fig.13. shows the Ratio distributions of another GGA in which the premature convergence occurred at the 151st generation. There are many points distributed over 1 after the premature convergence occurred. It indicates that the LGA still worked efficiently even though the GGA reached the premature convergence.

## 5 Summary

A local search genetic algorithm based on the genetic recombination was discussed in this paper. The LGA acted as a local search operator in the GGA. A good result was presented when the city numbers of the sub tour were set around the half city numbers of the main tour. But, we think it would be reasonable running a small LGA for a big TSP instance, because half city numbers of a big TSP still forms a new big TSP instance. It may be more effective in distributed and parallel processing running the LGA as an island to improve the main tour. This will be discussed in our future works.

## References

1. A. Homaifar, S. Guan, and G. E. Liepins, A New Approach to the Traveling Salesman Problem by Genetic Algorithms, in Proceedings of the 5th International Conference on Genetic Algorithms, pp. 460–466, Morgan Kaufmann, (1993).
2. C.N. Fiechter, A Parallel Tabu Search Algorithm for Large Traveling Salesman Problems, Discrete Applied Mathematics and Combinatorial Operations Research and Computer Science, vol. 51, pp. 243–267, (1994).
3. C.R.Reeves. Genetic algorithms and neighborhood search. in Evolutionary Computing, AISB Workshop, Pages 115-130. Leeds, U.K. (1994).
4. David E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learnin, Appendix D, Addison-Wesley, Boston, MA, (1989).
5. D. S. Johnson and L. A. McGeoch, The Traveling Salesman Problem: A Case Study in Local Optimization, in Local Search in Combinatorial Optimization, (E. H. L. Aarts and J. K. Lenstra, eds.), Wiley and Sons, New York, (1996).
6. E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization. New York: Wiley and Sons, (1985).

7. G. Reinelt, The Traveling Salesman: Computational Solutions for TSP Applications. Vol. 840 of Lecture Notes in Computer Science, Springer Verlag, Berlin, Germany, (1994).
8. Holland, J.H. Adaptation in Natural and Artificial Systems, Univ. of Michigan Press (1975)
9. J.Grefenstette, R. Gopal, B. Rosimaita, and D. V. Gucht, Genetic Algorithms for the Traveling Salesman Problem, in Proceedings of an International Conference on Genetic Algorithms and their Applications, pp. 160–168, (1985).
10. J.Y. Potvin, The Traveling Salesman Problem: A Neural Network Perspective,ORSA Journal on Computing, vol. 5, pp. 328–348, (1993).
11. Keiji Maekawa,Hisashi Tamaki,Hajime Kita and Yoshikazu Nishikawa:A method for the traveling salesman problem based on the genetic algorithm,Transactions of the Society of Instrument and Control Engineers,Vol.31,No.5,pp.598-605,May (1995)(In Japanese).
12. L. M. Gambardella and M. Dorigo, Ant Q: A Reinforcement Learning Approach to the Traveling Salesman Problem, in Proc. 12th International Conference on Machine Learning, pp. 252–260, Morgan Kaufmann, (1995).
13. Michael D. Vose The Simple Genetic Algorithm: foundations and theory,MIT Press, (1999).
14. M.Yamamura, T.Ono, S.Kobayashi, Character-Preserving Genetic Algorithms for Traveling Salesman Problem Journal of The Japanese Society for Artificial Intelligence Vol.7, No.6, pp. 1049-1059,(1992) (In Japanese).
15. P.C. Kanellakis and C.H. Papadimitriou. Local search for asymmetric traveling salesman problem (1980).
16. R. M. Brady, Optimization Strategies Gleaned from Biological Evolution, Nature, vol. 317, pp. 804–806, *Operations Research*, 28(5):1066-1099, (1985).
17. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Optimization by Simulated Annealing, Science, vol. 220, pp. 671–680,(1983).
18. S.Lin, and B. Kernighan, An Efficient Heuristic Procedure for the Traveling Salesman Problem. Operations Research., 21:498-516, (1973).
19. T. G. Bui and B. R. Moon, A New Genetic Approach for the Traveling Salesman Problem, in Proceedings of the First IEEE Conference on Evolutionary Computation, pp. 7–12, (1994).