An Improved Genetic Algorithm Using the Convex Hull for Traveling Salesman Problem

TAKENAKA Yoichi, FUNABIKI Nobuo
Division of Informatics and Mathematical Science,
Graduate School of Engineering Science, Osaka University
Toyonaka-shi, Osaka 560, JAPAN
E-mail{y-takena, funabiki}@ics.es.osaka-u.ac.jp

Abstract

In this paper, we propose an improved genetic algorithm for the traveling salesman problem(TSP) by using the "visiting order restriction theorem". The visiting order restriction theorem gives a necessary condition for the shortest tour of TSP on the Euclidean plane by using the convex hull. The convex hull for a set of points S on a plane is defined as the smallest convex polygon that encloses S. In our method, the initial tours are produced to satisfy the necessary condition of the theorem for the shortest path without incresing the computation time. The simulation results using 10 well-known benchmark problems show that our algorithm can find better tour with shorter time than Pál's algorithm.

1. Introduction

Traveling Salesman Problem (TSP) requires to find the shortest tour of visiting every given city exactly once. In this paper, the location of every city is given on a plane where the Euclidean distance metric is used. With these restrictions, this problem still belongs to NP-complete[1], and many researchers study geometrical properties such as volonoi diagrams, delaunay triangulation and convex hull to solve this problem[2, 3, 4, 5, 6]. Figure 1 shows the shortest cycle of 10-city problem by solid lines.

Due to its many important applications, a number of heuristic algorithms have been proposed to find relatively good solutions in polynominal time for TSP[4, 7, 8]. These are greedy algorithm, simulated annealing, neural network, ant colony system, genetic algorithm (GA) and so forth. Greedy algorithm is characterized by a myopic view, based only on local knowledge of the problem[4]. Simulated annealing is modeled the crystallizing process on thermal physics[9]. Neural network simulates the behavior of neurons with steepest descent method. Ant colony system is a set of cooperating agents called ants, using an indirect communication mediated by a pheromone. GA utilizes natural selection of chromosomes. GA is known as one of the most efficient algorithm for TSP.

GA needs a lot of initial chromosomes with diverseness to search good solutions efficiently. Therefore, GA usually generates initial chromosomes randomly. If we use a construction heuristic like greedy algorithm for initial chromosomes, we will encounter two problems: huge computation time to generate initial chromosomes, and lack of diverseness in initial chromosomes. For example, the greedy algorithm[3] needs $O(N \log N)$-time to generate a chromosome or a tour, and it can generate at most N different tours, where N is nunber of given cities.

In this paper, we propose a new method for generating good initial tours using the visiting order restriction theorem without increasing the computation time.

0-7803-4778-1/98 $10.00 © 1998 IEEE

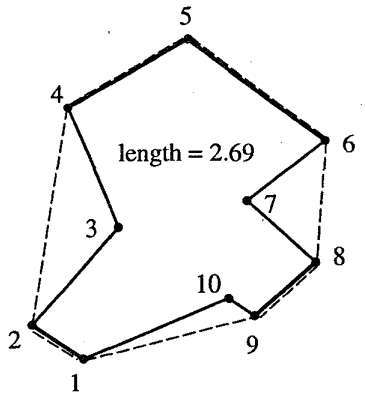

Figure 1: The shortest tour and the convex hull in the 10-city problem

2 Visiting Order Restriction theorem

Visiting order restriction theorem, proved by Flood[2], gives a necessary condition for the shortest tour of TSP on the Euclidean plane by using the convex hull. The convex hull for a set of points S in the plane is defined as the smallest convex plygon that encloses S. The dotted line in Figure 1 corresponds to the convex hull of the 10-city problem.

The theorem is descrived as followings.

Theorem :

Let S be the set of given cities in the plane, C be the convex hull of S, V be the set of cities on the boundary of C, P be the shortest tour of S. Let F be the polygon which is made up by tracing the cities of V according to the visiting order in the shortest path P. Then polygon F is equivalent to the convex hull C.

Assume N is the number of cities in S, and M is the number of cities on the boundary of the convex hull C, the theorem shows only ${}_{N-1}C_{M-1} \times (N-M)!$ tours can be the shortest tour candidates among $(N-1)!/2$ tours. For example, the convex hull of the 10-city problem C_{10} is shown in Figure 1 by the dotted lines. The cities on the boundary of C_{10} are $V = \{1, 2, 4, 5, 6, 8, 9\}$ and visiting order of the shortest path P_{10} is 1-2-3-4-5-6-7-8-9-10-1.

Table 1: Reduction rates for shortest tour candidates by the theorem

Name	No. Cities	Covnvex Hull	Reduction rate
10	10	7	1/360
bayg29	29	7	1/360
att48	48	11	1/1814400
eil76	76	4	1/3
kroA100	100	12	1/19958400
pr107	107	6	1/60
pr144	144	36	$\approx 1.9 \times 10^{-40}$
kroA200	200	11	1/1814400
lin318	318	17	$\approx 9.6 \times 10^{-14}$
att532	532	12	1/19958400

The theorem indicates that only 504 tours can be the shortest tour candidates among 181440 tours in 10-city problem. Table 1 shows reduction rates of 10 benchmark problems from 10 to 532 cities[10] by the theorem.

3 Pál's genetic algorithm for TSP

A genetic algorithm (GA) starts from several trial tours (the initial tours), which are very often chosen randomly. Using some preferable simple operations (crossover and mutation), GA creates new tours from the existing ones and selects only the shorter tours for further manipulations. The power of genetic algorithms lies in their ability to survey several different parts of the problem space simultaneously.

We adopt Pál's genetic algorithm C2 for TSP[8] to evalulate our method. In the algorithm, path expression is used to describe a tour, where cities are listed according to the visiting order. For instance, the shortest tour P_{10} is described by (1,2,3,4,5,6,7,8,9,10).

The algorithm C2 is as follows:

1. Initial tours generation
   (a) Set cities in numerical order as starting tour.
   (b) Generate two random numbers i, j, where $1 \leq i, j \leq N$.

(c) Swap i-th city and j-th city in the list.

(d) Repeat step (b) and (c) in $N \times 10$ times.

2. Parents selection

First, select a parent ($P1$) with the probability p_i, where p_i denotes the probability of selecting the i-th tour in the order of increasing length. The p_i is given by:

$$\begin{aligned} p_i/p_{i+1} &= R^{1/(L-1)} \\ p_1/p_L &= R. \end{aligned} \quad (1)$$

The constant R measures the preference of the shortest tours to the longer ones for reproduction.

Then, select another parent ($P2$) from two adjacent tours of $P1$ in the circular list with the probability of $1/2$.

3. Crossover

The crossover operation builds up the offspring tour from the two parents. Starting from a randomly chosen city, the visiting order of each city is selected step by step. In each step, the next city to be visited is chosen from two candidates. Each candidate is the nearest city along the parent tour that has not yet been visited in the offspring. The choice between the two candidates c_1 and c_2 is made with the probabilities:

$$\begin{aligned} p_{c_1} &= \frac{d(s,c_2)}{d(s,c_1)+d(s,c_2)} \\ p_{c_2} &= \frac{d(s,c_1)}{d(s,c_1)+d(s,c_2)}, \end{aligned} \quad (2)$$

where s is the last visited city in offspring, $d(x,y)$ denotes distance of cities x and y, and p_{c_1}, p_{c_2} are the probabilities to be visited c_1 and c_2 respectively.

4. Mutation

The mutation operation applies 2-opt move[4] to every city. Let $succ(x)$ be the city after x, l be a integer variable, and K be a constant.

(a) $l := 1$

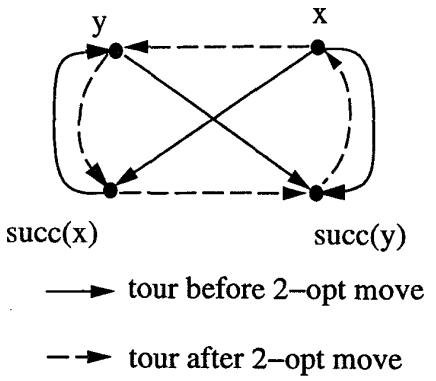

Figure 2: Transition of a tour by 2-opt move

(b) Let y be the l-th city from x. If the formula (3) stands, apply 2-opt move to cities x and y as shown in Figure 2 and go to (a).

$$\begin{aligned} &d(x, succ(x)) + d(y, succ(y)) \\ &\quad > d(x,y) + d(succ(x), succ(y)) \end{aligned} \quad (3)$$

(c) $l := l + 1$

If $l \leq K$ then go to (b), else terminate multation on city x.

5. Natural selection

The offspring is accepted if it is better than the worst tours in term of the tour length, and it is not identical with any existing tours. When the offspring is accepted, it is inserted next to its parents.

6. Terminate condition

GA is terminated if 100 offsprings are not accepted consecutively, else go to 2.

4 Introduction of the theorem

We propose a new method of generating initial tours using the visiting order restriction theorem without increasing the computation time. In our method, initial tours satisfy the necessary condition of the theorem by the following procedure:

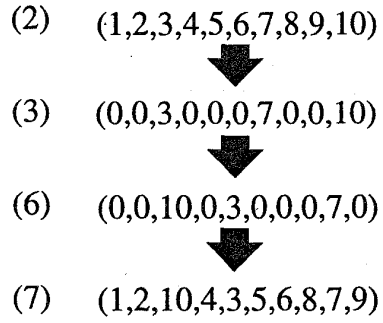

Figure 3: Process of initial tour generation

1. Find the convex hull C of the given cities S [1]. Here V is the set of cities on the boundary of C and N is the number of given cities.
2. Make a list of city labels in the ascent order.
3. Change the label of cities in V to "0".
4. Generate two random numbers i, j $(1 \leq i, j \leq N)$.
5. Swap the i-th label and j-th label in the list.
6. Repeat step 3. and 4. in $N \times 10$ times.
7. Permute "0" to the label of city labels in V by following the appearance order on C

   At the step 7., we use the clockwise appearance order to unify the direction of the tours. Figure 3 shows the process of making an inital tour for the 10-city TSP.

5 Simulation results

In order to verify the performance of our method, the simulator has been developed on a workstation SONY NEWS/5000. 10 benchmark problems from 10 to 532 were examined. We use the number of chromosomes $L = 200$, 2-opt move constant $K = 10$, and constance of bias selection $R = 1$[8]. Table 2 shows the computation time

[1]The $O(|S| \log |S|)$-time algorithm has been known to find the convex hull of S[11, 12].

Table 2: Computation time for initial tour generation

Name	Pál's Method	Our Method	Greedy
10	0.00(s)	0.00	0.00
bayg29	0.01	0.01	0.04
att48	0.01	0.02	0.22
eil76	0.02	0.03	1.35
kroA100	0.04	0.05	3.88
pr107	0.05	0.07	5.16
pr144	0.10	0.10	16.57
kroA200	0.18	0.20	61.33
lin318	0.44	0.45	244.15
att532	1.28	1.29	1132.78

to generate $L(= 200)$ initial tours in Pál's method and our method. The column named "greedy" gives the computation time to generate 200 different tours by greedy algorithm, where we show the computation time to generate N tours on the problem for $N \leq 200$.

From Table 2, the computational time to generate initial chromosomes in our method is almost same as the Pál's one. The greedy algorithm needs long computational time.

100 simulation runs with different initial states are performed in each of the benchmark problem. Table 3 and Table 4 show the simulation results by Pál's method and ours method respectively. Each tables summarize the shortest length among tours acquired in the simulation "*Shortest*", the average length "*Ave.*", the number of generated offsprings "*Iterations*", and the computation time "*Time*".

From the simulation results in Table 3 and Table 4, the shortest tours of the two methods are equal when N is smaller than 107. When N is smaller than 100, both methods can find the optimum tour of the problems. When N is larger than 200, our method is superior to Pál's method at both the shortest tour and average tour length. As for the average number of iteration steps, Pál's and our method are almost same for $N \leq 100$, while our method takes more iterations for $N \geq 100$. Because of the incrase of the iterations, the computation time of our method become $4.3\% \sim 9.8\%$ longer than Pál's method.

Table 3: Simuation results of Pál's method

Name	*Shortest*	*Ave.*	*Iterations*	*Time*
10	26907	26907	222	0(s)
bayg29	9074	9074	2806	13
att48	33524	33525	4417	40
eil76	108159	108394	7064	117
kroA100	21285	21312	7727	186
pr107	44302	44589	11550	292
pr144	58587	59019	10108	384
kroA200	29505	30413	14560	1020
lin318	45518	48102	8330	2128
att532	97087	102216	9298	3558

Table 4: Simulation results of our method

Name	*Shortest*	*Ave.*	*Iterations*	*Time*
10	26907	26907	266	0
bayg29	9074	9074	2804	13
att48	33524	33530	4543	40
eil76	108159	108461	6874	110
kroA100	21285	21311	7730	187
pr107	44302	44515	12300	318
pr144	58569	59017	11070	422
kroA200	29479	30253	15287	1064
lin318	44732	47996	8101	1894
att532	95917	101938	9993	3809

Figure 4 shows the relationship between *Iteration* and precision of the tours by Pál's and our methods, where the precision of tours is calculated by:

$$\text{precision of tours} = \frac{\text{ave. tour length} - \text{optimum tour length}}{\text{optimum tour length}} \quad (4)$$

From Figure 4, although pricision of the initial tours by our method's is worse than Pál's method's, our method can improve it faster than Pál's method. The same result is given on the other benchmark problems. Our method can improve the precision of the tours at the early stage of iterations. Thus, ours can provide better tour with shroter computation time than Pál's method.

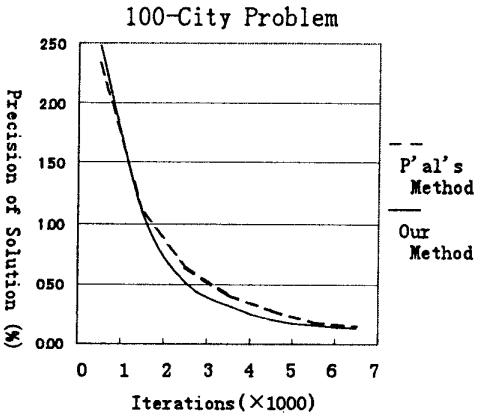

Figure 4: Relationship between generation number and precision of solutions

6 consideration

Pál's method uses 2-opt move in the mutation operation, which removes crossings in the tour to optimize the tour locally. On the other hand, the visiting order restriction theorem can globally optimizes the tour by restricting the visiting order of the cities on the convex hull. This theorem can be proved by using "Monge" property (the quadrangle inequality) and the fact the optimum tour has no crossings. Therefore, our method may produce less number of global crossings, which cannot be removed by 2-opt move, than the random tour, This conjecture may become true when the problem size become larger, because the number of cities between two adjacent cities on the convex hull become larger[11]. This conjecture is consistent with the simulation result that the difference on the tour precision between two methods become wider as the problem size is larger.

7 Conclusion

In this paper, we propose an improved genetic algorithm for the traveling salesman problem (TSP) by using the "visiting order restriction theorem". The visiting order restriction theorem gives a necessary condition for the shortest tour

of TSP on the Euclidean plane by using the convex hull. We propose a new method of generating initial tours to satisfy the visiting order restriction theorem without increasing the computation time. The simulation results using 10 well-known benchmark problems show that our algorithm can find better tours with shorter time than Pál's algorithm.

References

[1] R. Garey and S. Johnson, *Computers and Intractability , a guide to the theory of NP-completeness*, Freeman and Company, March 1991.

[2] M. M. Flood, "The traveling salesman problem", *Operations Research*, Vol. 4, pp. 61–75, 1956.

[3] G. Reinelt, *The travelling salesman*, Vol. 840 of *LNCS*. Springer-Verlag, Berlin, 1994.

[4] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shimoys, *The travelling salesman problem*, John Wiley & Sons, New York, 1985.

[5] V. Deineko, R. Van Dal, and G. Rote, "The convex-hull-and-line traveling salesman problem: a solvable case", *Information Provessing Letters*, Vol. 51, , 1994.

[6] V. G. Deineko and G. J. Woeginger, "The convex-hull-and-k-line travelling salesman problem", *Information Processing Letters*, Vol. 59, pp. 295–301, 1996.

[7] G. Zweig, "An effective tour construction and improvement procedure for the traveling salesman problem", *Operations Research*, Vol. 43, No. 6, pp. 1049–1057, 1995.

[8] K. F. Pal, "Genetic algorithm for the traveling salesman problem based on a heuristic crossover operation", *Biol. Cybern.*, Vol. 69, pp. 539–546, 1993.

[9] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing", *Science*, Vol. 220, No. 4598, pp. 671–680, May 1983.

[10] B. Bixby and G. Reinelt, Traveling salesman problem library [online], available at ftp://softlib.cs.rice.edu/pub/tsplib/tsplib.tar or ftp://elib.zib-berlin.de/pub/packages/mp-testdata /tsp/tsplib/.

[11] F. P. Preparata and M. I. Shamos, *Computational geometry: an introduction*, Springer-Verlag, 1988.

[12] J. O'Rourke, *Computational geometry in C*, Cambridge university press, 1993.