A Genetic Algorithm with Priority Selection for the Traveling Salesman Problem

Cha-Hwa Lin and Je-Wei Hu

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

Abstract**—**The conventional GA combined with a local search algorithm, such as the 2-OPT, forms a hybrid genetic algorithm (HGA) for the traveling salesman problem (TSP). However, the geometric properties which are problem specific knowledge can be used to improve the search process of the HGA. Some tour segments (edges) of TSPs are fine while some maybe too long to appear in a short tour. This knowledge could constrain GAs to work out with fine tour segments without considering long tour segments as often. Consequently, a new algorithm is proposed, called intelligent-OPT hybrid genetic algorithm (IOHGA), to improve the GA and the 2-OPT algorithm in order to reduce the search time for the optimal solution. Based on the geometric properties, all the tour segments are assigned 2-level priorities to distinguish between good and bad genes. A simulation study was conducted to evaluate the performance of the IOHGA. The experimental results indicate that in general the IOHGA could obtain near-optimal solutions with less time and better accuracy than the hybrid genetic algorithm with simulated annealing algorithm (HGA(SA)).

Keywords**—**Traveling salesman problem, hybrid genetic algorithm, priority selection, 2-OPT.

I. Introduction

THE traveling salesman problem (TSP) is: Given a number of cities and the distances (costs) of traveling from any city to any other city, find a minimum-length closed tour that visits each city exactly once and then returns to the starting city. The TSP has been known to be NP-hard that can not be solved within polynomial time [4]. However, heuristic algorithms such as the genetic algorithm (GA), could obtain near-optimal solutions within reasonable time. The GA searches feasible solutions with global perspective based on an analogy to the evolutionary principle of natural chromosomes. The conventional GA incorporates with a local search heuristic algorithm, such as 2-OPT, is called the hybrid genetic algorithm (HGA) [6]-[8], which makes the GA more powerful. The 2-OPT proposed by Croes [3] is a simple and widely used local search algorithm for TSP based on edges (tour segments). The algorithm starts with an arbitrary tour and gradually improves this tour by exchanging two of the tour segments with two other ones in the tour.

Cities of the TSP have some geometric properties which are problem specific knowledge [2], [5], [10]. The properties can be utilized to improve the search process of the GA. Because most of the tour segments are too long to appear in short tours, this information can guide the GA to focus more on short tour segments than long ones. Thus, all the tour segments are divided into two separate sets: the candidate set and the non-candidate set. The tour segments in the candidate set would have higher priorities than those in the non-candidate set.

For solving TSPs more efficiently, the proposed intelligent-OPT hybrid genetic algorithm (IOHGA) consists of three strategies: the skewed production (SP), the fine subtour crossover (FSC), and the intelligent-OPT (IOPT) in addition to the exchange mutation (EM) [1]. The three strategies are devised based on the concept of the 2-level priority scheme. The SP is a construction heuristic which produces initial tours with lower costs. The FSC is a crossover operator which finds the candidate fine tour segments in parents and preserves them for descendants. The IOPT is a local search algorithm which implements the 2-OPT search process more efficiently.

A simulation study was conducted to evaluate the performance of the IOHGA. The experimental results show that in general the IOHGA can provide near-optimal solutions with less time and much better accuracy than the hybrid genetic algorithm with simulated annealing algorithm (HGA(SA)) [8].

The remainder of this paper is organized as follows. Section II introduces the concept of the minimal tour segment. The proposed IOHGA is presented in Section III. The simulation results are provided in Section IV followed by some concluding remarks in Section V.

II. Minimal Tour Segment

It is difficult to identify the possible tour segments that are short for different TSP problems. According to the analysis, a high percentage, about 45%, of the tour segments in an optimal tour of the benchmark instances in TSPLIB [10] is local optimal. Therefore, the local optimal tour segment can be utilized to identify whether tour segments are candidate fine tour segments or not. A tour segment $S_{i,j}$ with the minimum cost of all the tour segments emanating from city i is called the minimal tour segment (MTS); otherwise, it is called the non-minimal tour segment (NMTS). Thus, an MTS $S_{i,j}$ is a local optimal tour segment for city i. The tour segments of the candidate set are MTSs and those of the non-candidate set are NMTSs.

Cha-Hwa Lin is with the Department of Computer Science and Engineering, and also with the Center for General Education, National Sun Yat-sen University, Kaohsiung, 80424, Taiwan (phone: 886-7-525-4345; fax: 886-7-525-4301; e-mail: chlin@cse.nsysu.edu.tw).
Je-Wei Hu is with the Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, 80424, Taiwan (e-mail: tethu0822@hotmail.com).

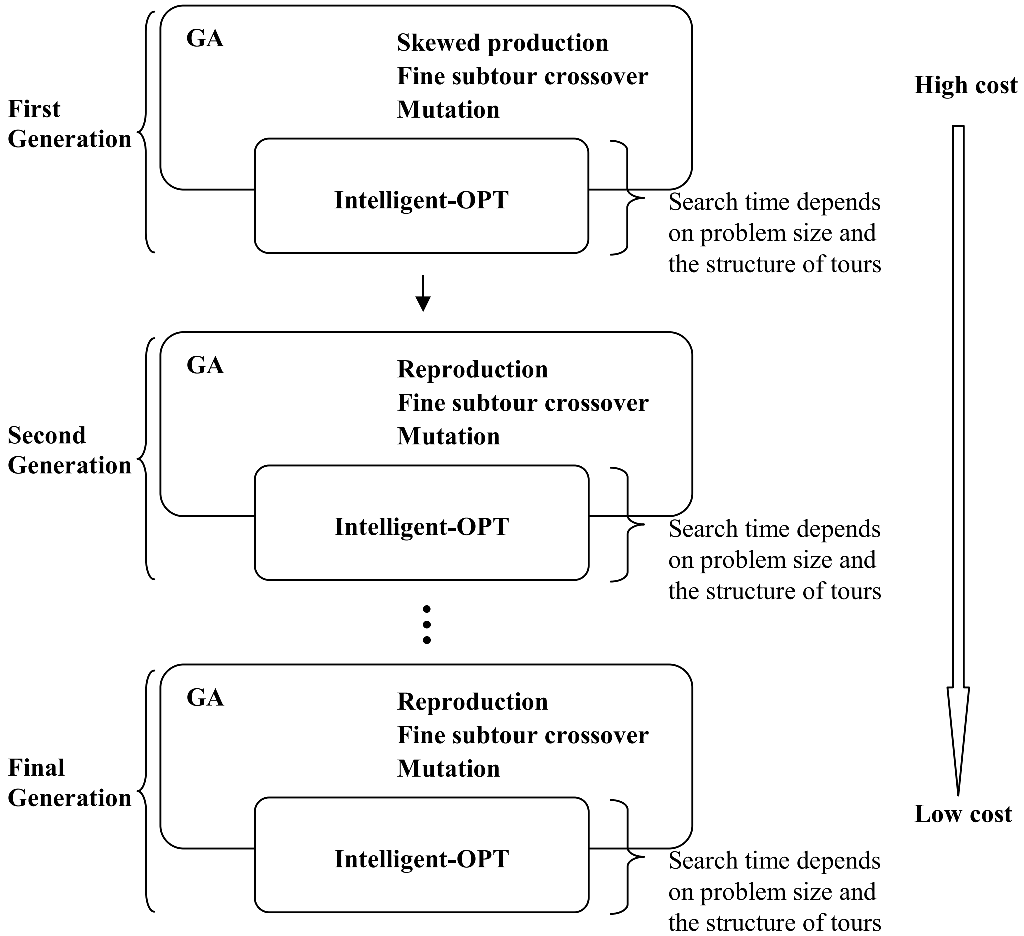


Fig. 1 The architecture of the proposed intelligent-OPT HGA

MTSs ought to have higher priority of appearing in a tour than NMTSs because MTSs might offer more contribution for the fitness of a tour than NMTSs. In the selection of tour segments, therefore, MTSs are considered first. Based on this idea, the conventional GA and 2-OPT local search algorithm can be improved.

III. The Intelligent-OPT Hybrid Genetic Algorithm

An intelligent-OPT hybrid genetic algorithm (IOHGA) for TSPs is proposed in this study. Generally, GAs starts from a population of chromosomes (solutions) at random as the first generation of candidate solutions and evolves toward better solutions by producing a new generation of chromosomes using crossover and mutation operators based on the population of the previous generation. In IOHGA (Fig. 1), chromosomes of the first generation is generated in favor of low tour cost by the skewed production (SP) in order to have better possible solutions than random selection at the beginning. The fine subtour crossover (FSC) is applied instead of the crossover operator of the GA in an attempt to preserve worthy subtours for offspring. The exchange mutation (EM) is utilized [1] to provide maximal variations after mutation. Then, the intelligent-OPT (IOPT) local search algorithm is implemented by modifying 2-OPT to search feasible solutions in a more efficient way.

A. The Skewed Production

The purpose of skewed production is to produce low cost tours for the first generation and hence better offspring. If chromosomes of the first generation with higher quality can be produced in an intelligent way, much of the search time might be saved to find an optimal solution. The SP is thus proposed to increase the quality of population for the first generation and accelerate search processes based on geometric properties of the MTS. A comparison between a random production and a skewed production of the first generation is depicted in Fig. 2. The quality of a chromosome is evaluated by the fitness function *f*(), which can be defined as

$$f(T) = 1 / c(T)$$

where *T* is a complete tour and *c*(*T*) is the cost of a tour *T*. The fitness of a tour increases while the cost of it decreases.

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

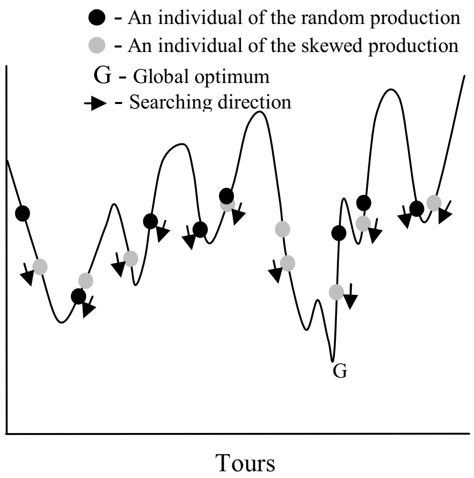


Fig. 2 Distribution of the chromosomes produced at random and in a skewed way in the solution space of a TSP

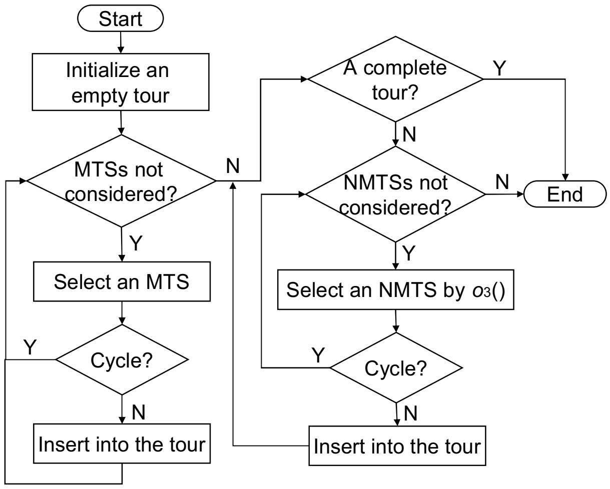
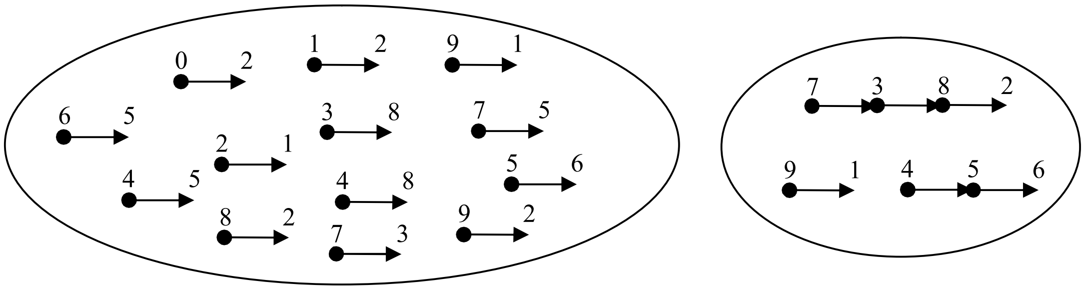


Fig. 3 The flowchart of the skewed production



(a) An *MTS_Pool* of MTSs

(b) The *tourPool* after inserting MTSs

Fig. 4 An example of collecting MTSs in the SP

In the SP, individuals should be produced to favor some characteristics in order to have higher fitness, but can not be generated without randomness because the genetic evolution needs difference among them. The preference of SPs is to select tour segments with low cost to form complete tours. The detail of how to create an individual with the preference is described in the following.

A tour is composed of several tour segments. In order to create a tour of lower cost, the determination of which tour segment is worthy to be selected to form a complete tour is needed. Because the minimal tour segments (MTS) are local optimal tour segments, MTSs might offer more contribution for the fitness of a tour than non-minimal tour segments (NMTS). For a tour to have lower cost, therefore, MTSs might be worthy to be selected as candidate tour segments. Namely, numbers of MTSs in a tour might bring it a lower cost. According to this reason, the number of MTSs in a tour is maximized to increase the probability of producing a tour of lower cost. The proposed skewed production (SP) attempts to control a GA searching the global optimum by exploiting the local optimal tour segments. The flowchart of the skewed production is shown in Fig. 3.

The Selection of MTSs

In the SP, *MTS_Pool* is a set of MTSs, *tourPool* is a temporary storage keeping track of a set of subtours which would become a complete tour where any two tour segments having a same city number are joined together at that city, and *NMTS_Pool* is a set of NMTSs. In the beginning, all the MTSs are stored into *MTS_Pool*. MTSs are repeatedly picked from *MTS_Pool* which have not been considered yet, and then inserted into *tourPool* until there is no more MTSs in *MTS_Pool* in order to maximize the number of MTSs in a tour. Moreover, MTSs would be picked up randomly to keep the variation among the resulting tours created by the SP.

An example of the selection of MTSs is shown in Fig. 4. The MTSs $S_{9,1}$, $S_{8,2}$, $S_{3,8}$, $S_{7,3}$, $S_{4,5}$, and $S_{5,6}$ are inserted into *tourPool* from *MTS_Pool* (Fig . 4a), respectively. After the insertion of $S_{5,6}$ (Fig . 4b), the *tourPool* is kept unchanged because that the insertions of $S_{0,2}$, $S_{1,2}$, $S_{2,1}$, $S_{4,8}$, $S_{6,5}$, $S_{7,5}$, and $S_{9,2}$ will result in a cyclic subtour of the *tourPool* and these MTSs are ignored in the following iterations. But there is still no complete tour in the *tourPool*. The subtours in the *tourPool* have to be connected to form a complete tour by NMTSs. The candidate tour segments for *tourPool* to form a complete tour would be

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

added into *NMTS_Pool*, which are all the NMTSs, because all the MTSs have been considered.

The Selection of NMTSs

For the selections of NMTSs, three objective functions are investigated. A tour might consist of not only MTSs but also NMTSs as in the example mentioned above. How to choose NMTSs which are worthy to be inserted into *tourPool* from *NMTS_Pool*? In the SP, an objective function is used to evaluate whether an NMTS is a fine tour segment worthy to be inserted into *tourPool* or not. Conventionally, the objective function $o_1()$ is defined as

$$o_1(S_{i,j}) = c(S_{i,j}) \quad and \quad i, j \in N$$

where $S_{i,j}$ is a tour segment from city i to city j, $c(S_{i,j})$ is the cost of $S_{i,j}$, and N is the set of all cities. An NMTS with a minimal cost from city i would be chosen to be inserted into *tourPool*.

In this greedy manner the resulting tour might not be optimal in the end. The NMTS chosen to be inserted into *tourPool* might not contribute to form a complete tour of a lower cost as expected. Therefore, $o_1()$ has to be modified in order to evaluate NMTSs more accurately for a better selection and a lower cost complete tour. More geometric properties could be considered for the modification of $o_1()$. Let us consider what problems this greedy manner would encounter first. In order to simplify the illustration of the non-optimal problem of this greedy manner, the sample data of *tourPool* in Fig. 4b would be used as in Fig. 5a. In the example, $ST_{i,j}$:c denotes that the cost of subtour $ST_{i,j}$ is c and T represents a complete tour. The example illustrates how subtours $ST_{4,6}$, $ST_{7,2}$, and $ST_{9,1}$ in *tourPool* (Fig. 5a) are connected with the NMTSs selected from *NMTS_Pool* (Fig. 5b) to form a complete tour based on the cost of NMTSs. The number beneath a tour segment $S_{i,j}$ in Fig. 5 is the value of $o_1(S_{i,j})$, which equals the cost of $S_{i,j}$, $c(S_{i,j})$. The selection process using $o_1()$ is listed as follows.

1. Because $o_1(S_{2,0})$ = 5 is the minimal cost among the tour segments in *NMTS_Pool* (Fig. 5b) and would not result in a cyclic subtour, $S_{2,0}$ would be chosen from *NMTS_Pool* and inserted into *tourPool*. Then, $ST_{7,2}$ is connected with $S_{2,0}$ to form $ST_{7,0}$ having the cost of 23 (Fig. 5c). The tour segments $S_{1,0}$, $S_{2,0}$, $S_{2,4}$, $S_{2,9}$, and $S_{6,0}$ either emanating from city 2 or terminating at city 0 are deleted from *NMTS_Pool* after processing $S_{2,0}$. The new *tourPool* contains subtours $ST_{4,6}$, $ST_{7,0}$, and $ST_{9,1}$ and the new *NMTS_Pool* contains NMTSs $S_{0,4}$, $S_{0,7}$, $S_{0,9}$, $S_{1,4}$, $S_{1,7}$, $S_{6,7}$, and $S_{6,9}$.
2. Similarly, $S_{0,7}$ would be chosen as the candidate tour segment, but then ignored and deleted from *NMTS_Pool* for the insertion of it into *tourPool* would result in a cyclic subtour (Fig. 5d). Therefore, *tourPool* is unchanged.
3. $S_{1,4}$ would be chosen from *NMTS_Pool* and inserted into *tourPool* (Fig. 5e). Then, subtours $ST_{4,6}$ and $ST_{9,1}$ are connected by $S_{1,4}$ to form $ST_{9,6}$ having the cost of 22. The new *tourPool* contains subtours $ST_{7,0}$ and $ST_{9,6}$ and the new *NMTS_Pool* contains NMTSs $S_{0,9}$, $S_{6,7}$, and $S_{6,9}$.
4. $S_{6,7}$ would be chosen from *NMTS_Pool* and inserted into *tourPool* (Fig. 5f). Then, subtours $ST_{7,0}$ and $ST_{9,6}$ are connected by $S_{6,7}$ to form $ST_{9,0}$ having the cost of 55. The new *tourPool* contains a subtour $ST_{9,0}$ and the new *NMTS_Pool* contains an NMTSs $S_{0,9}$.
5. Finally, $S_{0,9}$ would be chosen from *NMTS_Pool* and inserted into *tourPool* (Fig. 5g), and $ST_{9,0}$ is connected with $S_{0,9}$ to form a T having the cost of 67. The new *tourPool* results in a complete tour T with the cost of 67 and the new *NMTS_Pool* is empty.

In the above example, although the cost of $S_{2,0}$ is quite low, $S_{1,4}$, $S_{6,7}$, and $S_{0,9}$ with cost of 7, 10, and 12 respectively, might increase the cost of the tour much and reduce the quality of a tour in next steps. In order to solve the problem, another objective function is examined to evaluate the quality of tour segments. The objective function $o_2()$ is defined as

$$o_2(S_{i,j}) = \min_{x \in N} c(S_{i,x}) / c(S_{i,j}) \quad and \quad i, j \in N$$

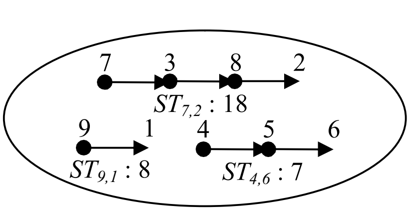
where $S_{i,j}$ is an NMTS from city i to city j, $S_{i,x}$ is an MTS from city i to city x, N is the set of all cities, $c(S_{i,j})$ is the cost of $S_{i,j}$, and the range of $o_2(S_{i,j})$ is (0, 1]. For a city i, $o_2()$ is defined to be the cost ratio of an MTS $S_{i,x}$ to an NMTS $S_{i,j}$. The larger the value of $o_2()$, the higher the possibility of an NMTS $S_{i,j}$ being a candidate fine tour segment and lower probability to be replaced in the future, because $S_{i,j}$ has less variation between it and an MTS $S_{i,x}$. In other words, the NMTS $S_{i,j}$ is worthier to be inserted into *tourPool* than the other NMTSs. Therefore, $o_2()$ identifies the tour segment with a better probability to result in a near-optimal tour.

The concept of $o_2()$ is depicted in Fig. 6, where $c_{i,j}$ is the cost of tour segment emanating from city i to city j and c_i is the cost of the MTS emanating from city i. The NMTS $S_{w,z}$ with a larger value of $o_2()$ would have a higher probability to be a candidate fine tour segment and not be replaced by others in the future, because the variation between $S_{w,z}$ and S_{w,m_w} is smaller than that between $S_{x,y}$ and S_{x,m_x}. Thus, $S_{w,z}$ is worthier to be inserted in *tourPool* than $S_{x,y}$. In Fig. 7, $S_{0,7}$ is worthier to be chosen than $S_{2,0}$ because $o_2(S_{0,7}) = 0.50 > o_2(S_{2,0}) = 0.40$.

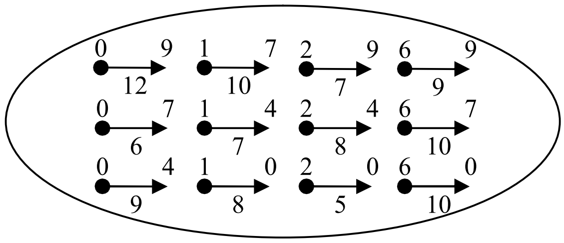
Fig. 8 shows the *tourPool* given in Fig. 5a are connected by tour segments to form a complete tour using $o_2()$. The selection process is listed as follows.

1. Because $o_2(S_{0,7})$ = 0.50 is the largest value of $o_2()$ among the tour segments in *NMTS_Pool* (Fig. 8b) and would not result in a cyclic subtour, $S_{0,7}$ is chosen from *NMTS_Pool* and inserted into *tourPool*. Then, $ST_{7,2}$ is connected with $S_{0,7}$ to form $ST_{0,2}$ having the cost of 24 (Fig. 8c). The tour segments $S_{0,4}$, $S_{0,7}$, $S_{0,9}$, $S_{1,7}$, and $S_{6,7}$ either emanating from city 0 or terminating in city 7 are deleted from *NMTS_Pool* after processing $S_{0,7}$. The new *tourPool* contains subtours $ST_{0,2}$, $ST_{4,6}$, and $ST_{9,1}$ and the new *NMTS_Pool* contains NMTSs $S_{1,0}$, $S_{1,4}$, $S_{2,0}$, $S_{2,4}$, $S_{2,9}$, $S_{6,0}$, and $S_{6,9}$.
2. Similarly, $S_{2,0}$ would be chosen as the candidate tour segment, but then ignored and deleted from *NMTS_Pool* for the insertion of it into *tourPool* would result in a cyclic subtour (Fig. 8d). Therefore, *tourPool* is unchanged.
3. $S_{6,9}$ would be chosen from *NMTS_Pool* and inserted into

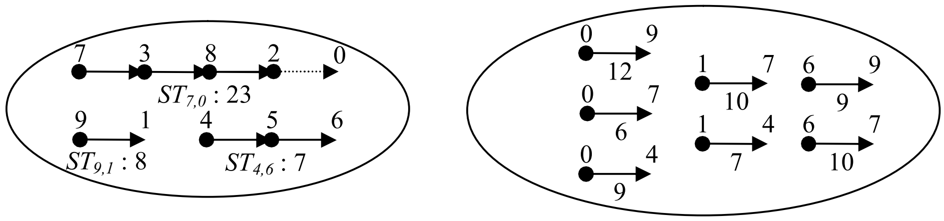
Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf



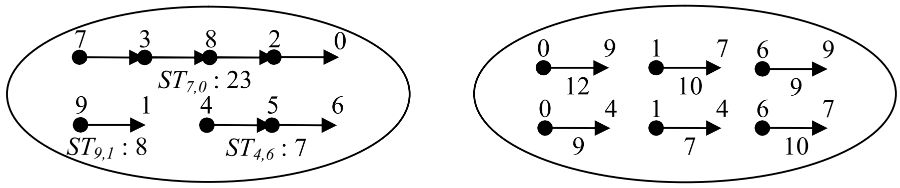
(a) Subtours $ST_{4,6}$, $ST_{7,2}$, and $ST_{9,1}$ in *tourPool* without city 0



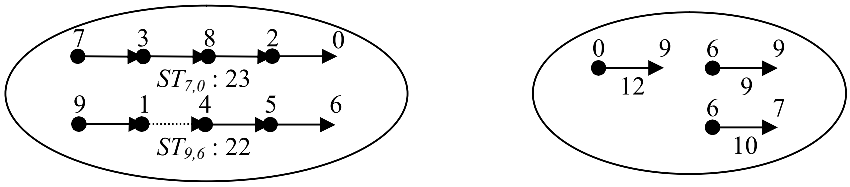
(b) The candidate tour segments in *NMTS_Pool* for (a) to form a complete tour



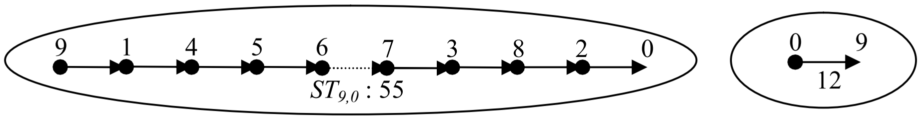
(c) *tourPool* and *NMTS_Pool* after inserting $S_{2,0}$



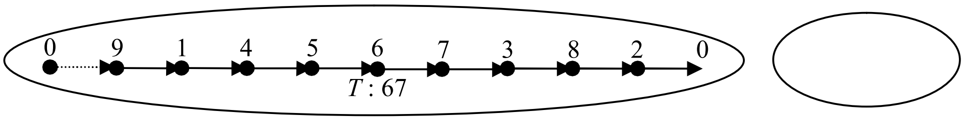
(d) $S_{0,7}$ is deleted without being inserted into *tourPool* because a cyclic subtour would be



(e) *tourPool* and *NMTS_Pool* after inserting $S_{1,4}$



(f) *tourPool* and *NMTS_Pool* after inserting $S_{6,7}$



(g) *tourPool* and *NMTS_Pool* after inserting $S_{0,9}$

Fig. 5 An example of connecting tour segments to form a complete tour by a greedy manner $o_1()$

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

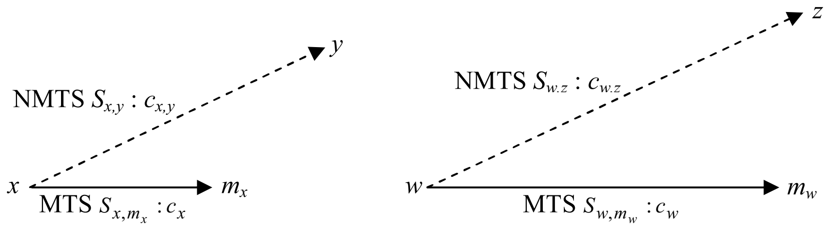


Fig. 6 The concept of the objective function $o_2()$

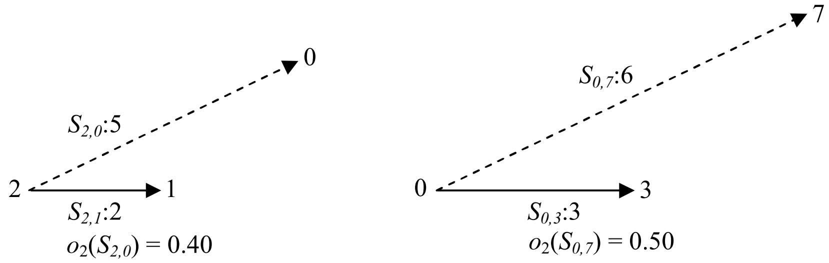


Fig. 7 An example of the objective function $o_2()$

tourPool (Fig. 8e). Then, subtours $ST_{4,6}$ and $ST_{9,1}$ are connected by $S_{6,9}$ to form $ST_{4,1}$ having the cost of 24. The new *tourPool* contains subtours $ST_{0,2}$ and $ST_{4,1}$ and the new *NMTS_Pool* contains NMTSs $S_{1,0}$, $S_{1,4}$, and $S_{2,4}$.

4. $S_{2,4}$ would be chosen from *NMTS_Pool* and inserted into *tourPool* (Fig. 8f). Then, subtours $S_{0,2}$ and $ST_{4,1}$ are connected by $S_{2,4}$ to form $ST_{0,1}$ having the cost of 56. The new *tourPool* contains subtour $ST_{0,1}$ and the new *NMTS_Pool* contains an NMTS $S_{1,0}$.
5. Finally, $S_{1,0}$ would be chosen from *NMTS_Pool* and inserted into *tourPool* (Fig. 8g). Subtours $ST_{1,0}$ are connected with $S_{1,0}$ to form a tour *T* having the cost of 64. The new *NMTS_Pool* is empty. The new *tourPool* results in a complete tour *T* with a cost smaller than that of using $o_1()$.

Thus, $o_2()$ makes the selection of NMTSs better than $o_1()$. However, when NMTSs with different costs have the same value of $o_2()$, one of them would be randomly selected as the candidate tour segment which might not contribute to form a complete tour with a lower cost.

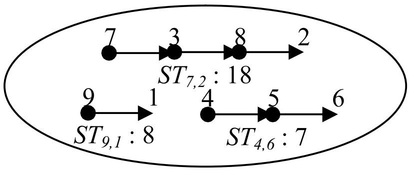
The concept of this overcorrect situation is depicted in Fig. 9. Suppose the NMTSs $S_{x,y}$ and $S_{w,z}$ have the same value of $o_2()$ = 2 which means that the variation between $S_{x,y}$ and S_{x,m_x} equals that between $S_{w,z}$ and S_{w,m_w}, but the cost $c_{x,y}$ is smaller than $c_{w,z}$. $S_{x,y}$ should have a higher probability of being a candidate fine tour segment than $S_{w,z}$. However, this information is not reflected by $o_2()$. To solve the problem, a new objective function $o_3()$ is defined as

$$o_3(S_{i,j}) = c(S_{i,j})^2 / \min_{x \in N} c(S_{i,x}) \quad and \quad i, j \in N$$

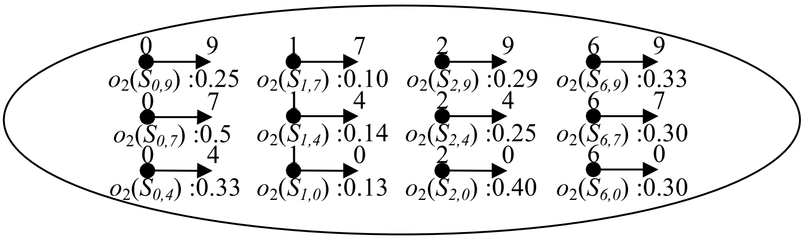
where $S_{i,j}$ is an NMTS from city *i* to city *j*, $S_{i,x}$ is an MTS from city *i* to city *x*, *N* is the set of all cities, and $c(S_{i,j})$ is the cost of $S_{i,j}$. The smaller the value of $o_3()$, the better an NMTS to be selected as a candidate tour segment and inserted into *tourPool*.

Although $o_3()$ is the inverse of $o_2()$ multiplied by the cost of the tour segment in consideration, the inverse of $o_2()$ plays a different role as the weight for a tour segment. An illustrative example to form a complete tour using $o_3()$ is shown in Fig. 10 and explained in detail as follows.

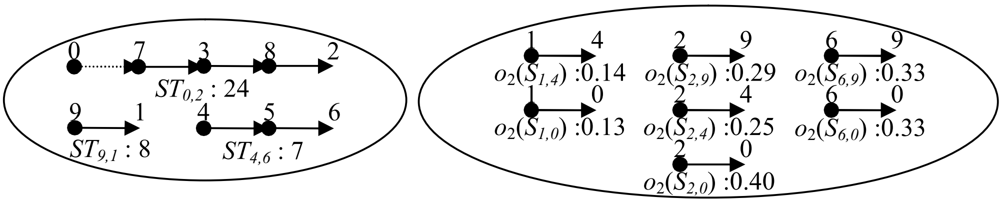
1. Because $o_3(S_{0,7})$ = 12 is the smallest value of $o_3()$ among the tour segments in *NMTS_Pool* (Fig. 10b) and would not result in a cyclic subtour, $S_{0,7}$ is chosen from the *NMTS_Pool* and inserted into *tourPool*. Then, $ST_{7,2}$ is connected with $S_{0,7}$ to form $ST_{0,2}$ having the cost of 24 (Fig. 10c). The tour segments $S_{0,4}$, $S_{0,7}$, $S_{0,9}$, $S_{1,7}$, and $S_{6,7}$ either emanating from city 0 or terminating in city 7 are deleted from the *NMTS_Pool* after processing $S_{0,7}$. The new *tourPool* contains subtours $S_{0,2}$, $ST_{4,6}$, and $ST_{9,1}$ and the new *NMTS_Pool* contains NMTSs $S_{1,0}$, $S_{1,4}$, $S_{2,0}$, $S_{2,4}$, $S_{2,9}$, $S_{6,0}$, and $S_{6,9}$.
2. Similarly, $S_{2,0}$ would be chosen as the candidate tour segment, but then ignored and deleted from *NMTS_Pool* for the insertion of it into *tourPool* would result in a cyclic subtour (Fig. 10d). Therefore, the *tourPool* is unchanged.
3. $S_{2,9}$ would be chosen from the *NMTS_Pool* and inserted into *tourPool* (Fig. 10e). Then, subtours $ST_{0,2}$ and $ST_{9,1}$ is connected by $S_{2,9}$ to form $ST_{0,1}$ having the cost of 39. The new *tourPool* contains subtours $ST_{0,1}$ and $ST_{4,6}$ and the new *NMTS_Pool* contains NMTSs $S_{1,0}$, $S_{1,4}$, and $S_{6,0}$.
4. $S_{6,0}$ would be chosen from the *NMTS_Pool* and inserted into *tourPool* (Fig. 10f). Then, $ST_{4,6}$ is connected with $S_{6,0}$ to form $ST_{4,0}$ having the cost of 17. The new *tourPool* contains subtours $ST_{0,1}$ and $ST_{4,0}$ and the new *NMTS_Pool* contains an NMTS $S_{1,4}$.



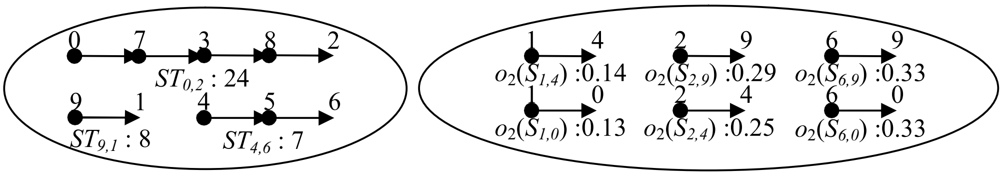
(a) Subtours $ST_{4,6}$, $ST_{7,2}$, and $ST_{9,1}$ in *tourPool* without city 0



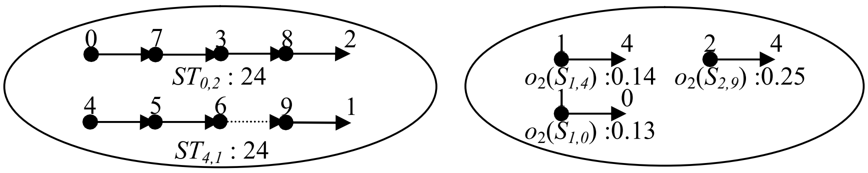
(b) The candidate tour segments in *NMTS_Pool* for (a) to form a complete tour



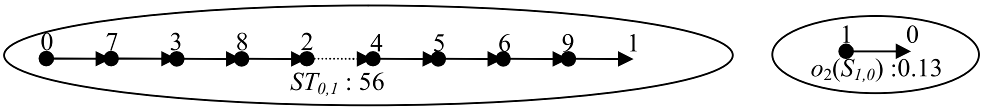
(c) *tourPool* and *NMTS_Pool* after inserting $S_{0,7}$



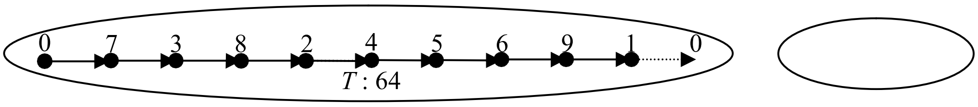
(d) $S_{2,0}$ is deleted without being inserted into *tourPool* because a cyclic subtour would be



(e) *tourPool* and *NMTS_Pool* after inserting $S_{6,9}$



(f) *tourPool* and *NMTS_Pool* after inserting $S_{2,4}$



(g) *tourPool* and *NMTS_Pool* after inserting $S_{1,0}$

Fig. 8 The subtours are connected by tour segments to form a complete tour using $o_2()$

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

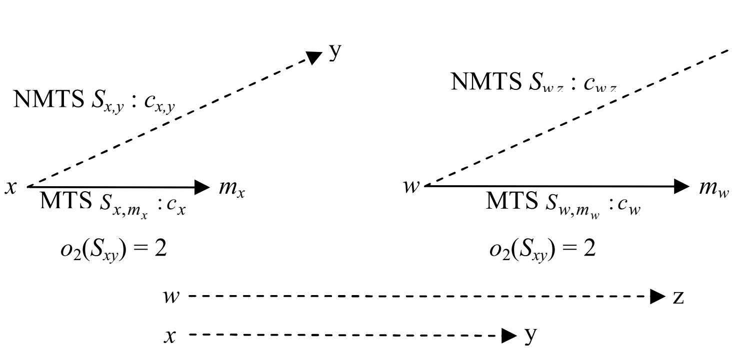


Fig. 9 The concept of the overcorrect situation

5. Finally, $S_{1,4}$ would be chosen from the *NMTS_Pool* and inserted into *tourPool* (Fig. 10g). Subtours $ST_{0,1}$ and $ST_{4,0}$ are connected with $S_{1,4}$ to form a tour *T* having the cost of 63. The new *NMTS_Pool* is empty. The new *tourPool* results in a complete tour *T* with a cost smaller than that of using $o_1()$ or $o_2()$.

B. The Fine Subtour Crossover

The fine subtour crossover (FSC) is a genetic operator using priority selection from a repository of candidate fine subtours. A candidate tour may consist of both fine and bad tour segments. The FSC attempts to retain segments worth preserving for the offspring. The idea behind the strategy is that the probability for a low cost tour to be a near-optimal solution might be increased if it contains more fine tour segments. The recognition of fine tour segments plays an important role in the process of the FSC.

An MTS $S_{i,m}$ has more likelihood of being a fine tour segment than an NMTS for $S_{i,m}$ is a local optimum of all the tour segments emanating from city *i*. To obtain a low cost tour, the priority of preserving an MTS for the offspring is thus higher than that of preserving an NMTS.

In addition to MTSs, some NMTSs might be fine tour segments based on the problem inputs. More information can be obtained for making decisions from the tour structures. The structure similarity between lower cost tours that lead to near-optimal solutions implies that the common tour segments among tours might worth to be inherited by the offspring. Tour segments in two distinct tours F and S have the same terminal cities are called common tour segments; otherwise, they are non-common tour segments. For example, tour segments $S_{x,y}$ in F and $S_{y,x}$ in S are considered as common tour segments. The common NMTSs in parents might be fine tour segments for the next generation.

Therefore during the genetic evolution process, the useful information by analyzing geometric properties of tour structures can be obtained to decide whether a tour segment is a candidate fine tour segment and also their priorities. The priority order of a tour segment being viewed as a candidate fine tour segment and inherited by the offspring is listed as: 1. Common MTSs, 2. Non-common MTSs, 3. Common NMTSs, and 4. Non-common NMTSs.

How to choose among tour segments if they have the same order as a candidate fine tour segment? Based on the geometric properties, non-common MTSs of the two parents are considered according to their costs while the common NMTSs and non-common NMTSs of the two parents are considered by $o_3()$ defined in Section III. *A*.

The FSC is much similar to the complete subtour exchange crossover (CSEX) [7]. The same purpose of both the FSC and the CSEX is to find the tour segments which are worthy to be preserved for descendants. The difference between the FSC and the CSEX is that the FSC emphasizes on that MTSs have higher priority than NMTSs to be inherited from parents, while the CSEX considers all tour segments are equally fair.

C. The Intelligent-OPT

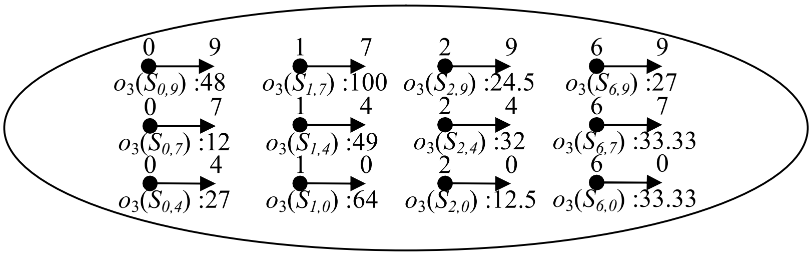
Although some derivations of 2-OPT, such as LK heuristics [9], are proposed to improve the efficiency of the 2-OPT search process, the complexities of the algorithms are increased. In contrast, the proposed intelligent-OPT (IOPT) could not only improve the efficiency of the 2-OPT search process but also maintain the simplicity of the algorithm. The flowchart of the intelligent OPT is shown in Fig. 11.

In 2-OPTs, the recombination would take place only when it can reduce the cost of the tour. Two tour segments might be chosen and checked without doing any exchanges. The random selections might spend a lot of time for nothing. Therefore, random selections of the 2-OPT search process are considered less efficient than directed selections. The proposed IOPT attempts to guide the selection of 2-OPT search process to promising directions such that the search time of finding a local optimal solution could be reduced.

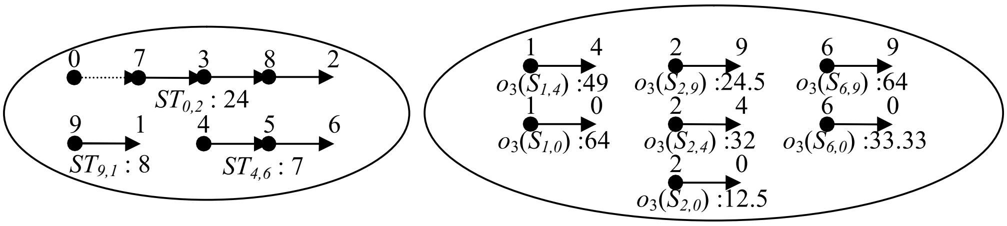
The information of geometric properties could be used for guiding the selection of the two edges in a more intelligent way. A tour is composed of MTSs and NMTSs. By observation, an exchange of an MTS with an NMTS in the 2-OPT search process might be inappropriate because the exchange would increase the cost of the tour. Furthermore, an exchange of an MTS with another MTS in the 2-OPT search process would gain no benefit because an MTS is a local optimum of all the tour segments emanating from the same city. An NMTS is more appropriate to be exchanged with some other tour segment

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

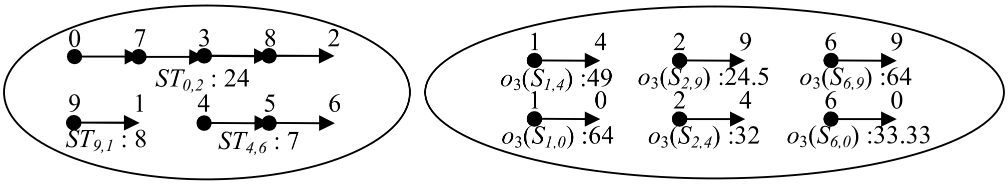
(a) Subtours $ST_{4,6}$, $ST_{7,2}$, and $ST_{9,1}$ in *tourPool* without city 0 (same as Fig. 8a)



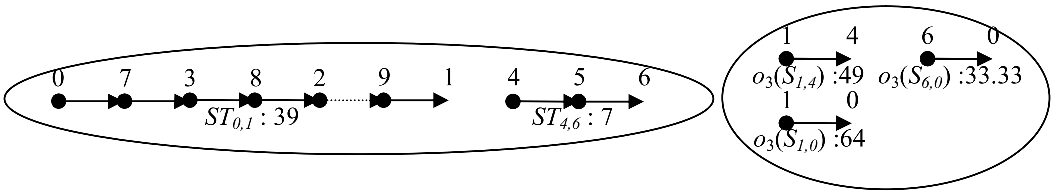
(b) The candidate tour segments in *NMTS_Pool* for (a) to form a complete tour



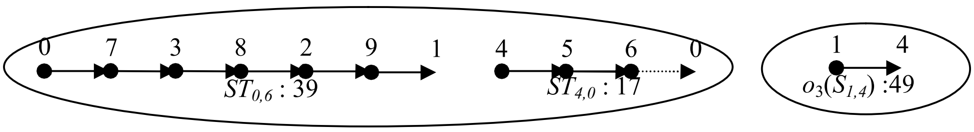
(c) *tourPool* and *NMTS_Pool* after inserting $S_{0,7}$



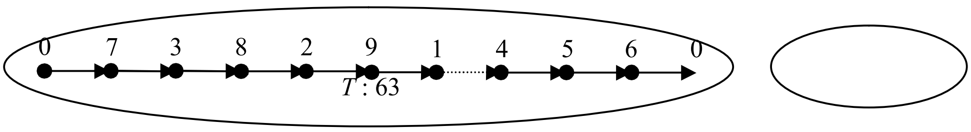
(d) $S_{2,0}$ is deleted without being inserted into *tourPool* because a cyclic subtour would be formed



(e) *tourPool* and *NMTS_Pool* after inserting $S_{2,9}$



(f) *tourPool* and *NMTS_Pool* after inserting $S_{6,0}$



(g) *tourPool* and *NMTS_Pool* after inserting $S_{1,4}$

Fig. 10 The subtours are connected by tour segments to form a complete tour using $o_3()$

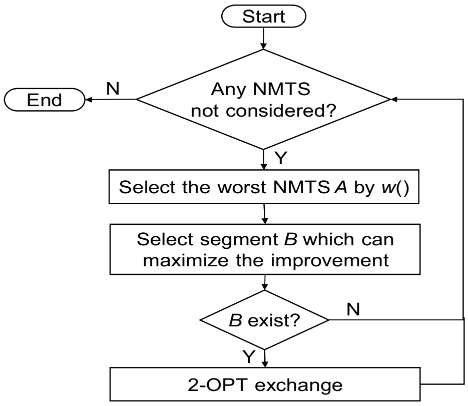


Fig. 11 The flowchart of the intelligent OPT

which might be an MTS or an NMTS based on this reason. The IOPT would first find an NMTS which is worthy to be replaced, and then find some tour segment that could reduce the tour cost maximally by utilizing the 2-OPT exchange. The process is repeated until the stopping criterion is reached.

Which NMTS in a tour should be chosen for replacement? For example, suppose an MTS $S_{a,m}$ emanates from city a and an NMTS $S_{a,b}$ emanating from city a to city b has the highest cost. That is, the cost variation between $S_{a,m}$ and $S_{a,b}$ is maximum in considering all the tour segments emanating from city a only. For city a, therefore, the NMTS $S_{a,b}$ is the worst tour segment and ought to be displaced in a tour. The NMTS $S_{a,b}$ if replaced might make more improvement on the fitness of a tour than any other tour segments emanating from the same city. Furthermore, in order to choose the worst tour segment to be replaced among all the NMTSs of a tour, the cost variation between an MTS $S_{a,m}$ and any $S_{a,b}$ of NMTSs should be normalized. The cost variation of a tour segment $S_{i,j}$ after normalization, $w(S_{i,j})$, can be expressed as

$$w(S_{i,j}) = c(S_{i,j}) / \min_{m \in N} c(S_{i,m}) \quad and \quad i, j \in N$$

where N is the set of all cities, $S_{i,j}$ is a tour segment emanating from city i to city j, and $c(S_{i,j})$ is the cost of tour segment $S_{i,j}$. An NMTS with the largest value of w() is the worst tour segment and would be replaced first in the IOPT search process to reduce the cost of a tour.

The value of w() is the cost ratio of an NMTS to an MTS, which represents the normalized variation between them. In a word, the purpose of the IOPT search process is to refine the worst tour segment of the tour repeatedly towards an optimal solution.

IV. Simulation Results

A prototype system of the proposed IOHGA has been constructed and implemented. In the simulation study, the algorithms of the IOHGA written in JAVA were performed on an Intel 80*86 desktop computer with a Celeron 2.80 GHz processor and 480 MB RAM, running under Windows XP. To demonstrate the effectiveness of the IOHGA, 14 benchmark files (instances) from 51-city to 439-city of TSPLIB [10] were tested in comparison with the hybrid genetic algorithm with simulated annealing algorithm (HGA(SA)) proposed by Katayama and Narihisha [8] using the same parameter settings as the HGA(SA). The benchmark instances are Euclidean cases of TSP problems. The HGA(SA) employed the complete subtour exchange crossover and incorporated with the simulated annealing algorithm as the metaheuristic.

The error rate of the obtained near-optimal solution is calculated by

$$Error \;\; Rate = \frac{Cost_{\,obtained} - Cost_{\,optimal}}{Cost_{\,optimal}} \times 100\ (\%)$$

In the experiments of the HGA(SA), the population size is set to be 10, the crossover rate is set to be 1.0, and the number of generations (NG) is set to be 200. Because the experiments of HGA(SA) were implemented on an S-4/5workstation (microSPARC Ⅱ, 110 MHz), the CPU time of the HGA(SA) has to be adjusted for comparison purpose. An original CPU time of HGA(SA) is multiplied by 110 and then divided by 2800 to obtain a relative CPU time compatible with the Intel 80*86 desktop computer. The CPU time of instances kroA100, kroA150, and rd100 is not given in the HGA(SA) .

The comparisons of the minimal, the maximal, the average error rates, the number of times that obtained the optimal solution in 10 runs (Opt/10 Runs), and the CPU time between the two algorithms are shown in Figs. 12 to 16, respectively. The IOHGA yield better accuracy than those of the HGA(SA) in general except for *pr107*. Moreover, the IOHGA spends less CPU time to obtain best solutions than the HGA(SA) except for *lin318* and *pr439*.

The CPU time of either IOHGA or HGA(SA) increases quickly as more cities are considered because the search space of the TSP problem is much larger. In HGA, most CPU time is spent by the local search heuristic. Therefore, the HGA(SA) adopted a small population size in order to reduce the CPU time. But from the perspective of genetic evolution, a small population size would make a premature convergence on a local optimum. Furthermore, if the population size is larger the premature convergence resulted from the genetic assimilation problem would not affect the IOHGA search process as much which might bring even better results.

V. Conclusion

In this paper, a new hybrid genetic algorithm using priority selection is presented for solving the traveling salesman problem. To improve the search process for the optimal solution without considering long tour segments as often, the proposed IOHGA constructs the geometric properties of the problem in a 2-level priority scheme as the underlining concept for devising three strategies: the skewed production, the fine subtour crossover, and the intelligent-OPT. Simulation results show that the accuracy of IOHGA is much better than that of HGA(SA). By using the 14 problem instances taken from TSPLIB, the average error rate of IOHGA is reduced from 8.46%/14 = 0.60% to 4.48%/14 = 0.32%; while the average CPU time is reduced from 322.81/11=29.34s to 295.96/11=26.90s when the

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

Open Science Index, Computer and Information Engineering Vol:2, No:6, 2008 publications.waset.org/14198.pdf

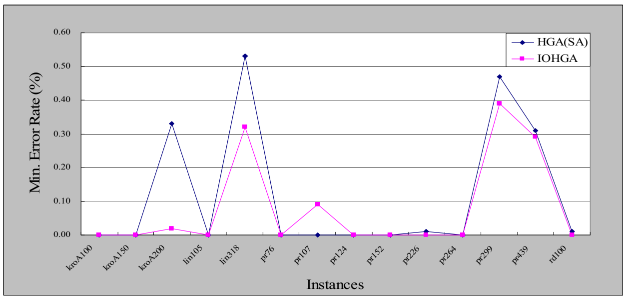


Fig. 12 The minimal error rates

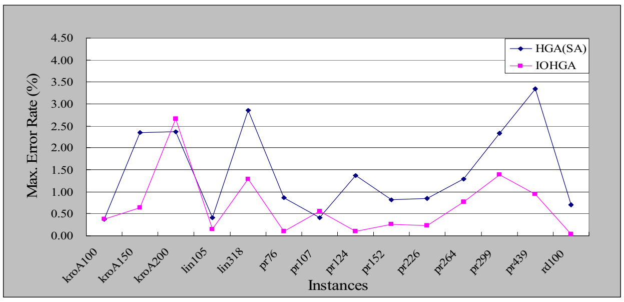


Fig. 13 The maximal error rates

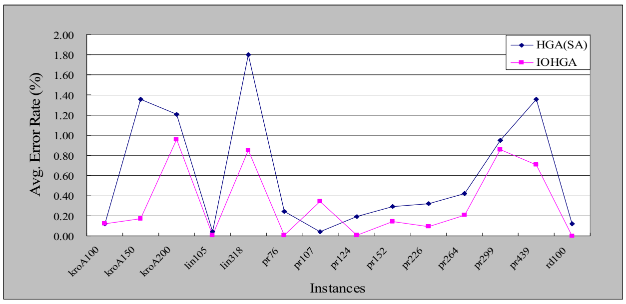


Fig. 14 The average error rates

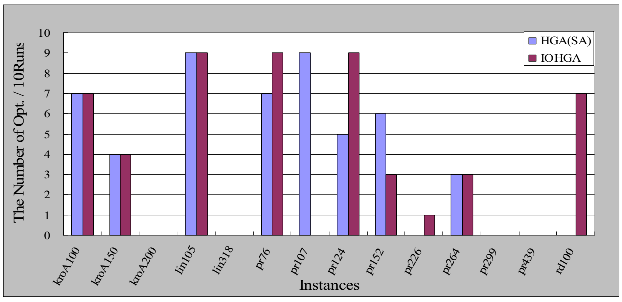


Fig. 15 The number of times obtaining optimal solution in 10 runs

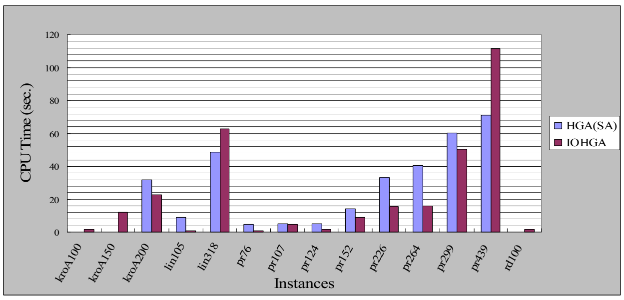


Fig. 16 The CPU time

three instances with unknown CPU times in HGA(SA) are excluded. Thus, the IOHGA improves the accuracy by 47%. Furthermore, the CPU time of IOHGA is reduced by 8.32%. If the case is not focused on the optimal solution, the IOHGA can provide near-optimal solutions more effectively.

The IOHGA might be incorporated with some clustering algorithm and applied to mobile agent planning problems in a real-time environment. However, the IOHGA could not handle large scale TSPs very well. Pruning hopeless expensive tour segments might reduce the searching space. Besides, more priority levels for classifying the tour segments could be considered in future studies.

References

[1] W. Banzhaf, “The “Molecular” Traveling Salesman,” *Biological Cybernetics*, 1990, 64: 7–14.

[2] Burkowski, F. J., “Proximity and priority: applying a gene expression algorithm to the Traveling Salesperson Problem,” *Parallel Computing*, Vol. 30, pp. 803-816, May 2004.

[3] G.A. Croes, “A Method for Solving Traveling Salesman Problems,” *Operations Research*, 1958, Vol. 6, No. 6, pp. 791-812.

[4] M.R. Garey, D.D. Johnson, and R.E. Tarjan, “The Planar Hamiltonian Circuit Problem is NP-complete,” *SIAM Journal of Computing*, 1976, 5:704-714.

[5] Grefenstette, J.J., “Incorporating Problem Specific Knowledge into Genetic Algorithms,” in *Genetic Algorithms and Simulated Annealing*, L. Davi, edtor, pp. 42-60, Morgan Kaufmann Publishers, 1987.

[6] Kido, T. “Hybrid Searches Using Genetic Algorithms,” in: *Genetic algorithm*, H. Kitano, editor, Sangyo Tosho, pp. 61-88, 1993.

[7] K. Katayama, H. Sakamoto, and H. Narihisa,, “An Efficiency of Hybrid Mutation Genetic Algorithm for Traveling Salesman Problem,” *In Proceedings of the Second Australia-Japan Workshop on Stochastic Models in Engineering, Technique & Management*, Gold Coast, Australia, 1996, pp. 294-301.

[8] K. Katayama and H. Narihisa, ”An Efficient Hybrid Genetic Algorithm for the Traveling Salesman Problem,” *Electronics and Communications in Japan, Part 3*, 2001, Vol. 84, No. 2, pp. 76-83.

[9] S. Lin and B.W. Kernighan, “An Effective Heuristic Algorithm for the Traveling Salesman Problem”, *Operations Research*, 1973, Vol. 21, No. 2, pp. 498-516.

[10] G. Reinelt, http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/index.html.