An Evolutionary Algorithm for Large Traveling Salesman Problems

Huai-Kuang Tsai, Jinn-Moon Yang, Yuan-Fang Tsai, and Cheng-Yan Kao

***Abstract*—This work proposes an evolutionary algorithm, called the heterogeneous selection evolutionary algorithm (HeSEA), for solving large traveling salesman problems (TSP). The strengths and limitations of numerous well-known genetic operators are first analyzed, along with local search methods for TSPs from their solution qualities and mechanisms for preserving and adding edges. Based on this analysis, a new approach, HeSEA is proposed which integrates edge assembly crossover (EAX) and Lin–Kernighan (LK) local search, through family competition and heterogeneous pairing selection. This study demonstrates experimentally that EAX and LK can compensate for each other's disadvantages. Family competition and heterogeneous pairing selections are used to maintain the diversity of the population, which is especially useful for evolutionary algorithms in solving large TSPs. The proposed method was evaluated on 16 well-known TSPs in which the numbers of cities range from 318 to 13 509. Experimental results indicate that HeSEA performs well and is very competitive with other approaches. The proposed method can determine the optimum path when the number of cities is under 10 000 and the mean solution quality is within 0.0074% above the optimum for each test problem. These findings imply that the proposed method can find tours robustly with a fixed small population and a limited family competition length in reasonable time, when used to solve large TSPs.**

***Index Terms*—Edge assembly crossover, evolutionary algorithm, family competition, heterogeneous pairing selection, Lin–Kernighan (LK) heuristic, traveling salesman problem.**

I. Introduction

THE traveling salesman problem (TSP) is a well-known NP-hard optimization problem, which requires the determination of the shortest round trip that passes through a set of M cities, each exactly once. TSPs raise important issues because various problems in science, engineering, and bioinformatics fields such as vehicle routing [1], scheduling problems [2], integrated circuits designs [3], physical mapping problems [4], and constructing phylogenetic trees [5] can be formulated as TSPs.

Manuscript received July 8, 2003; revised January 2, 2004. This paper was recommended by Associate Editor H. Takagi.

H.-K. Tsai is with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan, R.O.C. (e-mail: d7526010@csie.ntu.edu.tw).

J.-M. Yang is with the Department of Biological Science and Technology and Institute of Bioinformatics, National Chiao Tung University, Hsinchu 30050, Taiwan, R.O.C. (e-mail: moon@cc.nctu.edu.tw).

Y.-F. Tsai is with the Department of Social Studies Education, National Taipei Teachers College, Taipei 106, Taiwan, R.O.C. (e-mail: tyf.ts@msa.hinet.net).

C.-Y. Kao is with the Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan, R.O.C., and is also with the Bioinformatics Center, National Taiwan University, Taipei 106, Taiwan, R.O.C. (e-mail: cykao@csie.ntu.edu.tw).

Digital Object Identifier 10.1109/TSMCB.2004.828283

A large number of methods have been developed for solving TSPs. Evolutionary algorithms (EAs) indicate a very promising direction [6]. They are based on ideas taken from genetics and natural selection. An evolutionary algorithm can be generally applied to solve problems. It is especially appropriate for solving difficult optimization problems, for which traditional optimization methods are less efficient. However, general problem-independent EAs are frequently inefficient in solving TSPs, especially large TSPs.

Many approaches have been presented to improve further EAs for TSPs. Among these, designing TSP-specific operators, incorporating local searches, and maintaining population diversity are considered to be promising ways to solve TSPs. TSP-specified operators, such as, position-based operators which preserve the relative positions of cities in selected parents [7], [8], interval-based operators which inherit subpaths from parents [8]–[11], edge-based operators which preserve edges in selected parents and add new edges heuristically [2], [12]–[15], and geometry-based operators which generate children by considering geometric information [16], [17], can indeed help to solve the performance for solving TSPs. Methods that incorporate domain-specific local search methods into EAs, also called *memetic algorithms* [18], have also been shown to be efficient heuristics for solving TSPs [19]–[22]. Some mechanisms are specifically designed and constructed to prevent the premature convergence of TSP [22]–[26].

Some factors that improve the quality of solutions to TSPs obtained using EAs have been studied. For instance, Reinelt [27] showed that the edge exchange search was much more effective than node reinsertion or node exchange. Whitely *et al.* [2] claimed the preservation of edges is more important than that of the positions of nodes in the design of recombination operators, and the mechanisms for adding and preserving edges should be emphasized [2], [28]. This work examines four crossover and four local search heuristics based on their qualities of their solutions and the mechanisms by which "good edges" (that appear in a known optimal tour) are preserved and added. Edge assembly crossover (EAX) and Lin–Kernighan (LK) were found herein to be the most powerful operators among those surveyed. However, for large TSPs, EAX often requires a large population and LK barely escapes from a locally optimal solution, even though the length of the search is long. In this study, "large TSP" refers to one in which the number of cities exceeds 4 000.

Accordingly, a new evolutionary algorithm, called the heterogeneous selection evolutionary algorithm (HeSEA), is proposed for TSPs. It is obtained by integrating EAX and LK through family competition and heterogeneous pairing selection. The two former genetic operators are the main

1083-4419/04$20.00 © 2004 IEEE

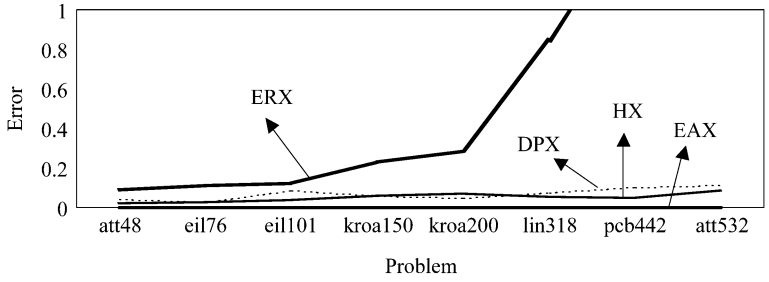

Fig. 1. Comparison of four edge-based crossovers, including EAX, HX, ERX, and DPX, with the simple GA, applied to eight TSP problems, in terms of error (1). EAX is the best method. The results obtained using HX and DPX are similar, while ERX performed poorly when the number of cities was high.

TABLE I
MAIN MECHANISMS FOR PRESERVING AND ADDING EDGES BY SOME EDGE-BASED CROSSOVERS, INCLUDING EAX, DPX, HX, AND ERX

Operator	Mechanisms	
	preserving edge	adding edge
EAX	Offspring inherits edges from parents according to the edge length and the frequencies appeared in the population	New edges are added into offspring to modify the intermediate solution by adding short edges through a spanning tree method
DPX	Only the common edges in both parents are passed into offspring	Intuitively adding short edges to make the intermediate solution feasible
HX	Offspring inherits edges from parents where the edge length is as short as possible	New edges are randomly added to become a feasible solution
ERX	Offspring inherits edges from parents as much as possible, but the maximum number of inherited edges is not guaranteed	New edges are randomly added to become a feasible solution

search operators and the latter two mechanisms are applied used to maintain the diversity of the population. HeSEA was evaluated using 16 well-known traveling salesman problems [29] in which the numbers of cities ranged from 318 to 13 509. Experimental results demonstrate that the quality of solution of the proposed EA is within 0.0074% of the optimal for test problems that involve fixed small populations and limited local search lengths. The family competition and heterogeneous pairing selection in HeSEA can significantly improve the quality of the solution to a large TSP.

The rest of this paper is organized as follows. Section II analyzes four crossover and four local search methods of EA, as applied to some TSPs, based on solution quality and the ability to add and preserve edges. Section III describes the proposed evolutionary algorithm. Section IV elucidates some characteristics of the proposed approach. Section V presents experimental results and concluding remarks.

II. ANALYSIS OF GENETIC- AND LOCAL-SEARCH METHODS

A robust evolutionary algorithm should have powerful and efficient genetic operators. Various crossover and local search methods have been proposed for solving TSPs and some of these operators perform well. However, the reason for the effectiveness, or otherwise, of these operators, is unclear. Radcliffe and Surry [28] classified crossover operators according to the behaviors of preserved edges. As well as edges inherited from parents, foreign edges introduced during the execution of genetic operators, appear to be important. This section analyzes four edge-based crossover operators and four local search methods for solving some TSPs, in terms of the quality of solution and the mechanisms by which edges are added and preserved. A simple GA is employed to analyze the performances of these surveyed operators. This simple GA uses roulette wheel selection [30] to select the paired chromosomes, and total generation replacement with elitism to select individuals of the next generation. The size of the population is set equally to the number of cities (M) and the termination criterion is that the function has been evaluated more than $10\,000M$ times. In this study, the solution quality is given as an "error" percentage (%), defined as

$$\text{Error} = \frac{\text{average-optimum}}{\text{optimum}} * 100(\%) \tag{1}$$

where *average* is the average value of the best solutions obtained by the method over 20 trials and *optimum* is the minimum length of tour so far.

A. Analysis of Crossover Operators

Fig. 1 presents experimental results obtained by applying the simple GA with four edge-based crossovers, EAX [36], HX [13], ERX [2], and DPX [14], on eight small TSP benchmarks—*att48*, *eil76*, *eil101*, *kroa150*, *kroa200*, *lin318*, *pcb442*, and *att532*. The crossover rate was 1.0 when the performance of the crossover operators was analyzed. In these operators, DPX has not been designed to preserve edges but to discover new edges not contained in either of the parents. Tours will be improved if local search is applied after DPX. To compare the ability of preserving edges, no local search is applied. As shown, EAX yielded the best and ERX performed the worst of the four crossovers.

Table I briefly summarizes the mechanisms by which the edges of these four edge-based crossovers were preserved and

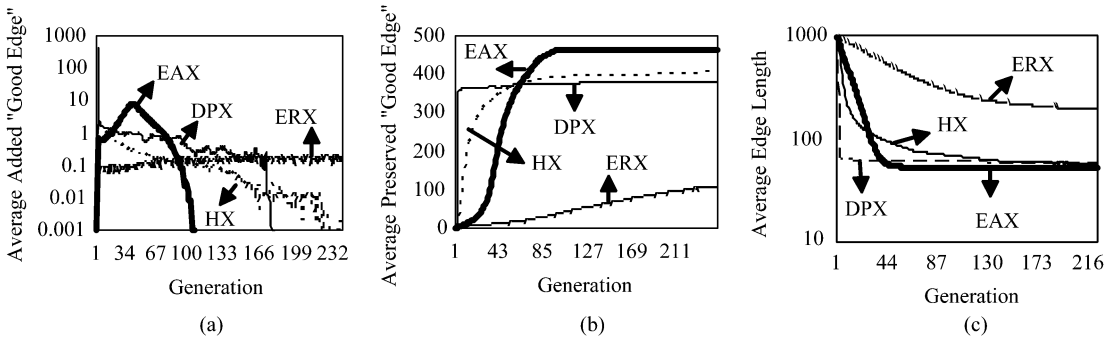

Fig. 2. Comparison of edge-based crossovers, EAX, DPX, HX, and ERX, applied to *att532*, in terms of the average numbers of (a) "good edges" added and (b) "good edges" preserved, as well as (c) the average edge length.

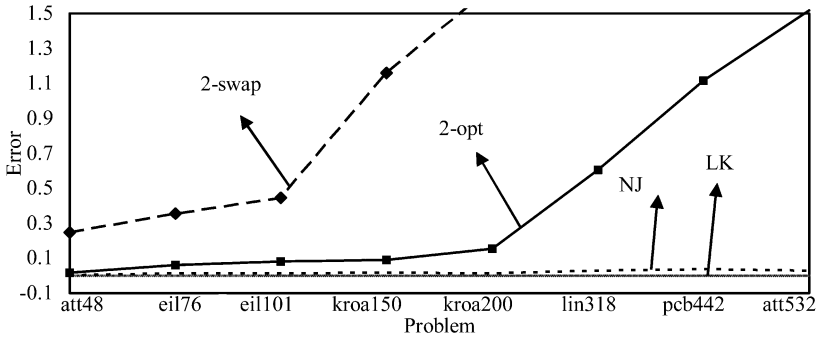

Fig. 3. Comparison of four local search methods—LK, 2-swap, 2-opt, and NJ—with the simple GA, applied to eight TSP problems in terms of "error" (1). LK is the best method and NJ yields results similar to those obtained by LK. 2-opt and 2-swap perform poorly when the number of cities is high.

added. DPX only preserves the common edges in parents. HX preserves edges by considering short edges in parents and ERX inherits common edges from parents as much as possible. EAX inherits short edges from parents and considers the frequency of edges in the current population. DPX makes the intermediate solution valid by adding short edges; ERX and HX randomly add new edges to generate feasible solutions, and EAX preserves edges heuristically and adds new edges with a greedy method, which is analogous to a minimal spanning tree.

Since the main search mechanism of edge-based crossovers is to generate offspring by preserving and adding edges, the characteristics of edge-based operators are analyzed by measuring their ability to add and preserve the "good edges" of EAX, DPX, HX, and ERX operators. A "good edge" is defined as an edge in a known optimal tour. Fig. 2 presents the results of applying these four operators to the problem *att532*. Similar results were obtained in solving the other TSP problems considered herein. Fig. 2(a) reveals that EAX can continuously add more "good edges" than other operators in the early stages (before the 70th generation). After the 70th generation, the average preserved "good edges" [Fig. 2(b)] and the average edge length [Fig. 2(c)] by EAX is better than that obtained using other operators. HX and DPX exhibit similar results for these factors and ERX yields the worst. These results agree with their capacities to add and preserve edges.

In summary, a good crossover operator for solving TSP problems should be edge-based and possess good mechanisms for adding and preserving "good edges." EAX is regarded as a good crossover operator because it meets these requirements.

B. Analysis of Local Search Methods

Some local search heuristics, such as 2-swap, 2-opt [31], neighbor-join (NJ) [32], and LK [33], have been extensively applied in GAs for solving TSPs. These techniques exchange some edges of parents to generate new children. Usually, stronger local search methods correspond to better performing GAs. The mechanisms by which these methods add and preserve edges vary. 2-swap arbitrarily changes two cities at a time, removing four edges at random and adding four edges at random. 2-opt, NJ and LK exchange edges if the generated solution is better than the original one. In each iteration, 2-opt and NJ exchange two and four edges, respectively; LK exchanges a variable number of edges.

Fig. 3 presents the experimental results obtained by applying the simple GA with the four tested local search methods—LK, NJ, 2-swap and 2-opt—on eight small TSP benchmarks. These benchmarks are *att48*, *eil76*, *eil101*, *kroa150*, *kroa200*, *lin318*, *pcb442*, and *att532*. The crossover rate is 0, according to an analysis of the performance of these methods. LK and NJ performed well. 2-opt became worse as the number of cities increased. 2-swap was the worst method.

The same strategies of analyzing crossover operators were used to analyze the performance of these methods by measuring their capacities to add and preserve "good edges" of LK, NJ, 2-opt, and the 2-swap method. Fig. 4 presents the results of these four methods, applied to problem *att532*, in terms of the addition and the preservation of "good edges" and the average edge length. Notably, similar results were obtained for the other seven test problems. Fig. 4(a) reveals that LK and NJ can add

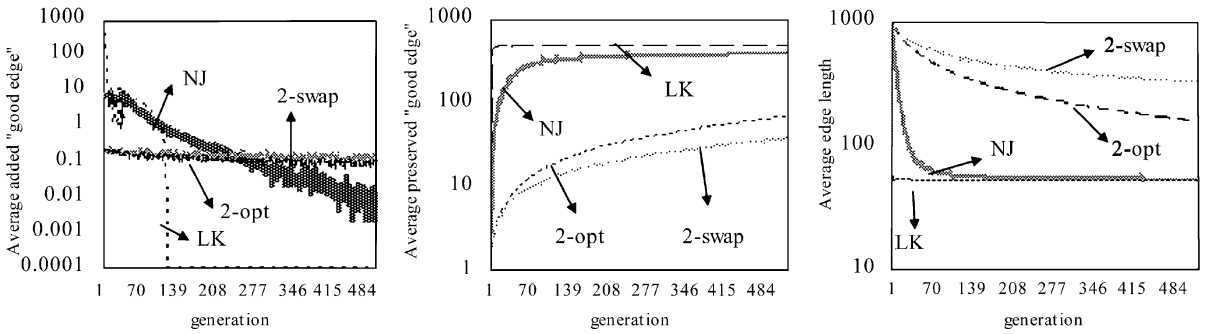

Fig. 4. Comparison of four local search methods, LK, 2-swap, 2-opt, and NJ applied to *att532*, in terms of the average numbers of (a) "good edges" added and (b) "good edges" preserved as well as (c) the average edge length.

more "good edges" than other operators in the early stages preserve more "good edges," on average, [Fig. 4(b)] in the late stages. LK is the most efficient. 2-opt and 2-swap behave similarly in adding edges and 2-swap is the worst method in preserving "good edges." These results are consistent with the authors' previous observations. In summary, a good local search method should possess good mechanisms for adding and preserving "good edges" for solving TSPs. LK is regarded as a good local search method which meets these requirements.

C. Brief Discussions of EAX and LK

Based on the solution qualities and the abilities to add and preserve "good edges," EAX and LK are the best methods for adding "good edges" in the early stage and preserving "good edges" in the late stage. However, for large problems (>4000 cities), EAX requires a large population (roughly equal to the number of cities), which is time-consuming. However, LK often finds a local optimal solution quickly but cannot easily improve the solution further, even by lengthening the search. The efficient combination of EAX and LK involves both problems of population size and search length.

A summary of the above observations leads to a new evolutionary algorithm. The proposed approach integrates EAX and LK with family competition and heterogeneous pairing selection (HpS). These two mechanisms are added to maintain the diversity of the population and increase the probability of generating better children. Sections III and IV will demonstrate experimentally that the proposed system, which successfully incorporates EAX and LK to compensate for the disadvantages of each other, can find reasonable tours robustly with a fixed small population and a limited family competition length.

III. Heterogeneous Selection Evolutionary Algorithm

Following Section II, a new evolutionary algorithm, which combines edge assembly crossover (EAX) and LK via family competition and heterogeneous pairing selection (HpS), is investigated. Fig. 5 depicts the main steps of the proposed approach. N solutions are randomly generated as the initial population. After the fitness of each solution in the population is evaluated, each sequentially becomes the "family father (s_i)," which generates a child by the following steps. The family father employs HpS to select itself (s_i) and another individual from the population, based on the edge similarity. These two individuals become the parents used by EAX, generating L children. The child with the shortest tour is designated the intermediate offspring (I_i). LK is then executed L times (where L is the family competition length) to generate a child (c_i) by mutating the mutated child I_i. Of each pair of family father (s_i) and child (c_i), the one with the fitter solution survives. Each individual (s_i) in the population sequentially follows the above steps to generate its child where $1 \leq i \leq N$. These N solutions become the new population of the next generation.

The proposed algorithm is terminated when one of the following criteria is satisfied:

1) maximum preset search time is exhausted;
2) all individuals of a population are identical;
3) all of the children generated in five continuous generations are worse than their respective parents.

Sections III-A–D describe HpS, EAX, and LK.

A. Heterogeneous Pairing Selection (HpS)

For each "family father (s_i)," HpS selects s_i and another individual from the current population based on edge similarity to perform crossover operators, such as EAX described herein. HpS is used to prevent trapping at a local optimum, by avoiding incest. In evolutionary processes, incest has two potentially negative effects—the loss of population diversity and the ineffective execution of crossover operations. The formulation and implementation of HpS is described as follows. Let $\{s_1, s_2, \ldots, s_N\}$ be the current population; $E(s_i)$ be the set of edges of the solution s_i, and $|E(s_i)|$ be the number of edges of $E(s_i)$. The number of identical edges $|T_{ij}|$ of two individuals (s_i and s_j) is defined as $|T_{i,j}| = |E(s_i) \bigcap E(s_j)|$.

For each individual s_i, let t_i be the average number of identical edges between s_i and the other individual in the population, where $t_i = (1/N-1)\sum_{j=1,j\neq i}^{N} |T_{i,j}|$, and N is the population size. Two extreme correspond to $t_i = 0$, when no edge of s_i appears in the other individuals and $t_i = M$, when all individuals are the same in the population where M is the number of cities in a TSP.

For a given individual s_i, HpS selects s_i and another individual s_j with $|T_{i,j}| \leq t_i$ for the EAX operator. This similarity-based mechanism is useful for maintaining the diversity of the population. The experimental results obtained herein are consistent with this claim.

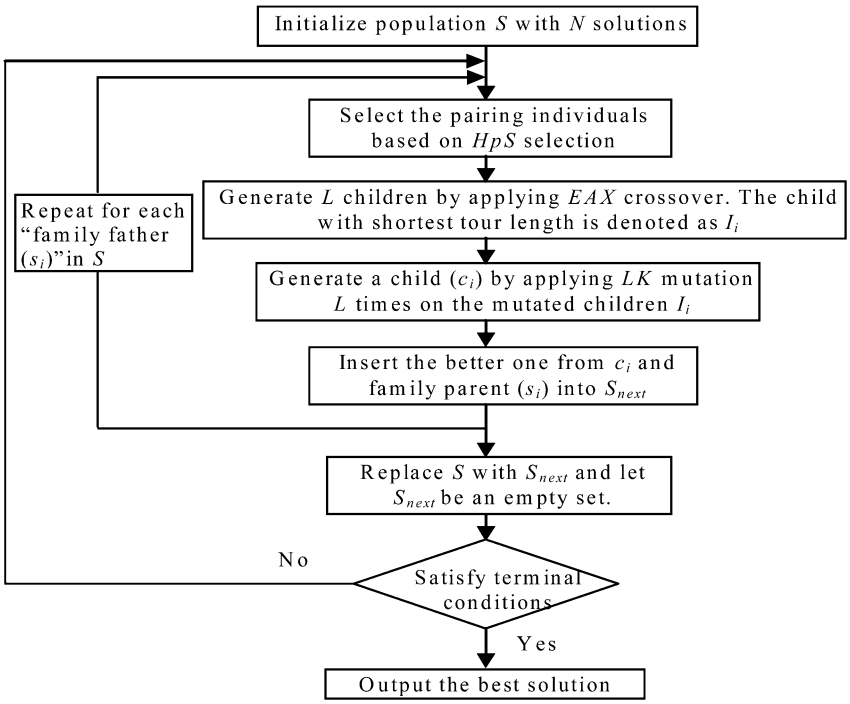

Fig. 5. Overview of the proposed evolutionary algorithm.

In practical implementation, another method is employed to calculate t_i because the time complexity of determining all t_i by calculating $|T_{i,j}|$ is $O(N^2M^2)$. At the beginning of each generation, $F(e)$, the number of instances of edge e in the current population, is calculated in advance where $e \in \{E(s_1) \bigcup E(s_2) \bigcup \ldots \bigcup E(s_N)\}$. The sum of $|T_{i,j}|$ over s_i in the population, can be reformulated as

$$\sum_{j=1, j\neq i}^{N} |T_{i,j}| = \sum_{j=1}^{N} |T_{i,j}| - |T_{i,i}| = \sum_{e \in E(s_i)} F(e) - M = \sum_{e \in E(s_i)} (F(e) - 1). \tag{2}$$

Therefore

$$t_i = \frac{1}{N-1} \sum_{e \in E(s_i)} (F(e) - 1). \tag{3}$$

Therefore, all t_i can be calculated in $O(NM)$ by looking up the precalculated table. The EAX crossover also uses the information $F(e)$, so the extra effort associated with calculating t_i is limited.

B. Family Competition

Family competition, derived from $(1 + \lambda)$-ES, is considered to be a local search procedure in HeSEA. Family competition proceeds as follows (Fig. 6). Each individual (s_i) sequentially becomes the "family father." This "family father" and another solution (s_j) selected by HpS act as parents to perform the EAX crossover operation to generate L children. The individual (I_i) with the shortest tour of these L children is chosen to perform the LK operation. LK mutates the mutated child L times. This strategy in which L solutions are generated from a single "family father" and a selection is then made, is called a family competition strategy. Each individual sequentially becomes the "family father," and LN offspring are generated in each generation.

C. Edge Assembly Crossover

EAX [12] has two important features—preserving parents' edges using a novel technique and adding new edges by a greedy method, analogous to a minimal spanning tree. Several issues, including the selection mechanism and heuristic methods, which affect the performance of EAX have been considered [34]–[38]. In this work, the main spirit of the EAX is maintained and HpS is used to replace the random pair selection (RpS), which was the original selection mechanism of the EAX genetic algorithm [12].

EAX is briefly described here. Two individuals, A and B, are chosen as the parents. EAX first merges A and B into a single graph, G. EAX traverses G to generate several *AB-cycles* by alternately selecting edges from parents A and B. According to the heuristic and random selection rules, some *AB-cycles* are selected to generate a quasisolution, which contains some disjointed subtours. Then, EAX uses a greedy method to merge these disjointed subtours into a valid solution. This solution is returned if its fitness exceeds that of its parents. Else, the procedure is repeated until a solution, that is better than both A and B, is obtained, or L children are generated, where L is the family competition length.

D. LK

LK, as investigated by Lin and Kernighan [33], is a powerful heuristic for obtaining near-optimal tours in solving TSPs. Various methods have been proposed, either to enhance the search

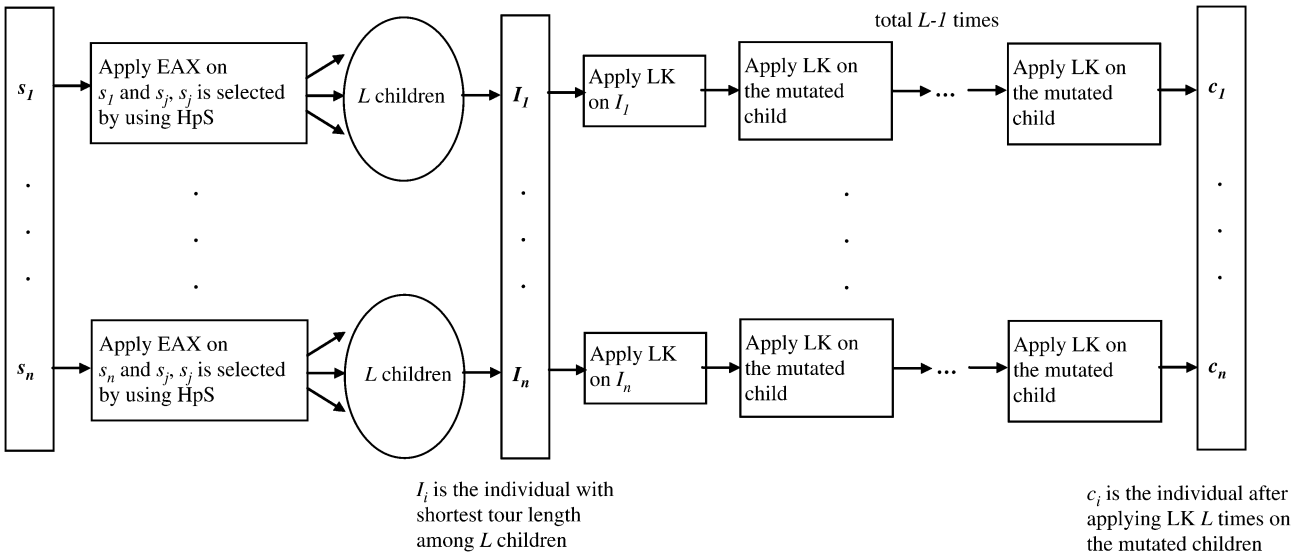

Fig. 6. Main steps in family competition.

abilities of LK [20], [39], [40] or others use LK as their local search mechanism [14], [16], [24]. The LK algorithm is rather complex. A natural way to illustrate LK is to treat LK as a variable k-opt. Given an initial tour, LK tentatively performs a sequence of flips to generate a new tour. The lengths of the initial tour and the generated tour are compared; the shorter tour is selected to perform a new flips-search. The search ends when all possible flips-searches for any city do not yield any improvement.

The concepts of *chained local optimization* [41], [42] are adapted in HeSEA. Chained local optimization is based on the idea of preserving some of the knowledge obtained in previous executions of LK. In chained-LK, LK is repeatedly efficiently applied to the preceding tour obtained by the most recent application of LK. HeSEA uses the *concorde* version of chained LK [43] as a local search method, where the family competition length (the number of kicks in the iterated LK is L).

IV. Analysis of HeSEA

This section considers some characteristics of HeSEA. First, the influences of the population size and the family competition length are discussed. Then HpS is experimentally shown to maintain the diversity of population and generate better children with a higher probability than random pair selection. Moreover, the efficiencies of combining EAX and LK are confirmed by quantifying their abilities to improve and add and preserve "good edges." The efficiencies of HpS and family competition were evaluated. Finally, HeSEA was compared to the original EAX and LK. The fairness of the analysis was ensured by setting the stop criterion to the execution of 1 000 M evaluations of the function, where M is the number of cities. 16 TSP problems from TSPLIB [29] are selected as the testing problem set, in which the numbers of cities range from 318 to 13 509. Table II lists the names of test problems, the corresponding numbers of cities and the lengths of optimal tours. All of these test problems are complete graphs.

TABLE II
Summary of 17 Tested TSPs Taken From TSPLIB [29], Including Problem Names, Number of Cities, and the Length of Optimal Tour

problem name	number of cities	length of optimal tour
lin318	318	42029
pcb442	442	50778
att532	532	27686
u574	574	36905
rat783	783	8806
pr1002	1002	259045
vm1084	1084	239297
pcb1173	1173	56892
u1432	1432	152970
vm1748	1748	336556
u2152	2152	64253
pr2392	2392	378032
pcb3038	3038	137694
fnl4461	4461	182566
frl5915	5915	565530
usa13509	13509	19982859
d15112	15112	1573084

A. Effect of the Population Size and Family Competition Length

As stated in Section II, the population size (N) and the family competition length (L) critically determine the computation time and quality of the solution. Various values of these two factors were tested on problems *att532*, *fnl4461*, and *usa13509*, to evaluate their effects in HeSEA. Typically, HeSEA yields similar curves for all test problems. Fig. 7(a) represents the relationship between the population size and the average error, defined by (1), obtained in solving problem *fnl4461*. For a particular number of evaluations of the function and the same family competition length, the average declines as the population size increases. The improvement is insignificant when population size exceeds 100. Accordingly, the population size is set to 100. Fig. 7(b) shows the effect of the family search length (L) on the quality of the solution

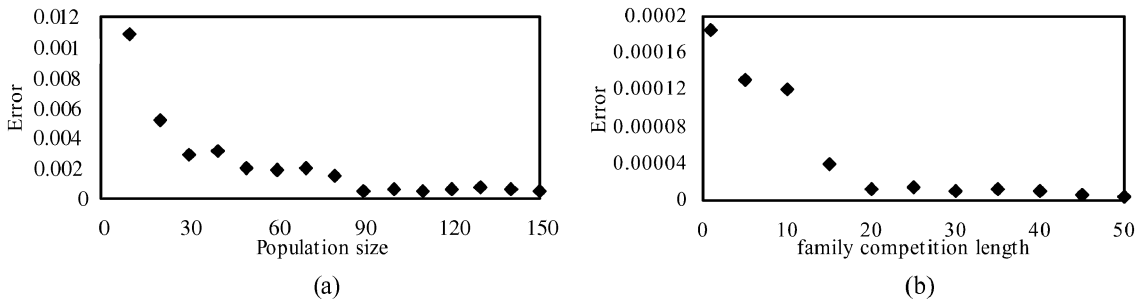

Fig. 7. Relationships (a) between the average error (1) and the population size (N) and (b) between the average error and the family competition length (L), for problem *fnl4461*. Based on the results, the population size (N) and family search length (L) are set to 100 and 20, respectively.

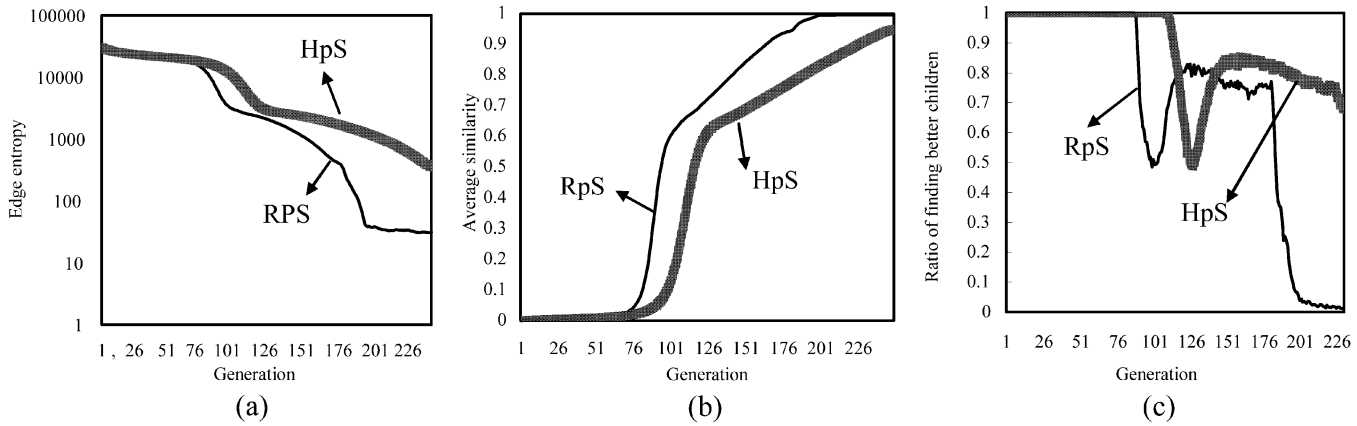

Fig. 8. Comparison of the edge entropy (see the text), the edge similarities within a population (see text), and the ability to generate better children, associated with the proposed system using RpS and those associated with the proposed system using HpS selection, as applied to problem *fnl4461*. (a) The edge entropy obtained using RpS declines faster than that of HpS. The large edge entropy implies that the population is highly diverse. (b) The average edge similarity obtained using RpS grows faster than that of HPS. (c) The curve associated with RpS declines steeply when the number of generations rises over 180, when applied to *fnl4461*, but that of HpS maintains a probability over 80%.

to problem *fnl4461*. The family search length L is set to 20 because improvements in the quality of solution are limited when L exceeds 20.

B. Ability of HpS to Maintain the Diversity of the Population

To examine the ability of the HpS to maintain the diversity of the population, edge entropy and the average edge similarity of a population were considered to elucidate the search behavior of the proposed approach, with HpS and random pair selection (RpS) mechanisms. RpS, the original selection mechanism of the EAX genetic algorithm [12], is regarded as an efficient mechanism for maintaining the diversity of the population [35]. The edge entropy of a population is $-\sum_{e\in X}(F(e)/N)\log_2(F(e)/N)$, where $X = E(s_1)\bigcup E(s_2)\bigcup\ldots\bigcup E(s_N)$, $F(e)$ is the number of edges, e, in the current population and N is the population size. The average edge similarity of a population is defined as $(2/N(N-1))\sum_i^N\sum_{j=1,j\neq i}^N|T_{i,j}|$, where $|T_{i,j}|$ is the number of edges shared by a pair of individuals (s_i and s_j). A large edge entropy and a low average edge similarity imply a diverse population.

Fig. 8(a) represents the relationships between the edge entropy and the number of the generations obtained by the proposed method with RpS and HpS selection, applied to problem *fnl4461*. Although these two methods yield similar trends, the edge entropy obtained with HpS declines more slowly than that obtained with RpS. In Fig. 8(a), the edge entropy obtained with RpS approaches zero as the number of generations exceeds 180 while the edge entropy obtained with HpS always exceeds that with RpS. Fig. 8(b) indicates that the value of the edge similarity obtained using HpS increases more slowly than that obtained by RpS used to solve problem *fnl4461*. Almost all individuals in a population generated by RpS are the same when the number of the generations exceeds 180 in solving problem *fnl4461*.

Furthermore, the ability to generate better children is measured to analyze the performance of HpS applied to the EAX crossover. Fig. 8(c) shows the probabilities of generating better children in each generation by applying RpS and HpS to solve problem *fnl4461*. The probabilities of both methods are near 100% in the early stage. However, the probability obtained using RpS declines steeply, while that obtained using HpS remains at approximately 80% even after all search steps have been executed. According to these results in Fig. 8, HpS is more powerful than RpS in both maintaining the diversity of the population and in generating better children.

C. Efficiency of Combining EAX and LK

The ability to add and preserve "good edges" (the edges in the optimal tour) and the improvements obtained by the application of EAX and LK operators in HeSEA are measured to evaluate the efficiency of combining EAX and LK. Fig. 9 presents the results of testing the system on problem *fnl4461*. Fig. 8(a) reveals that LK can add more "good edges" than EAX. The abil-

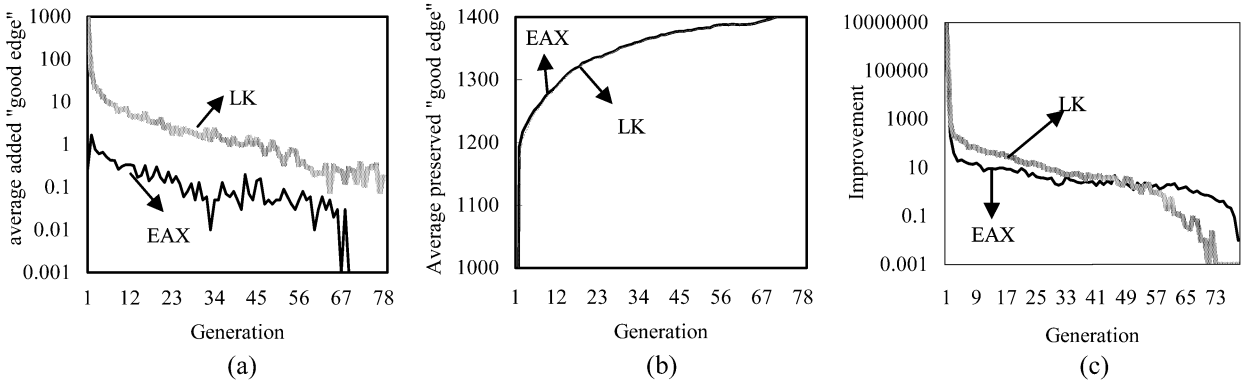

Fig. 9. Comparisons of the yielded improvements (defined in the text) and the abilities to add and preserve "good edges" associated with EAX and LK in the proposed method as applied to problem *fnl4461*. A "good edge" is defined as the edge in the optimal tour. (a) and (b) reveal the average number of added "good edges" and the average number of preserved "good edge," achieved using EAX and LK, respectively. (c) Presents the average improvement yielded by EAX and LK.

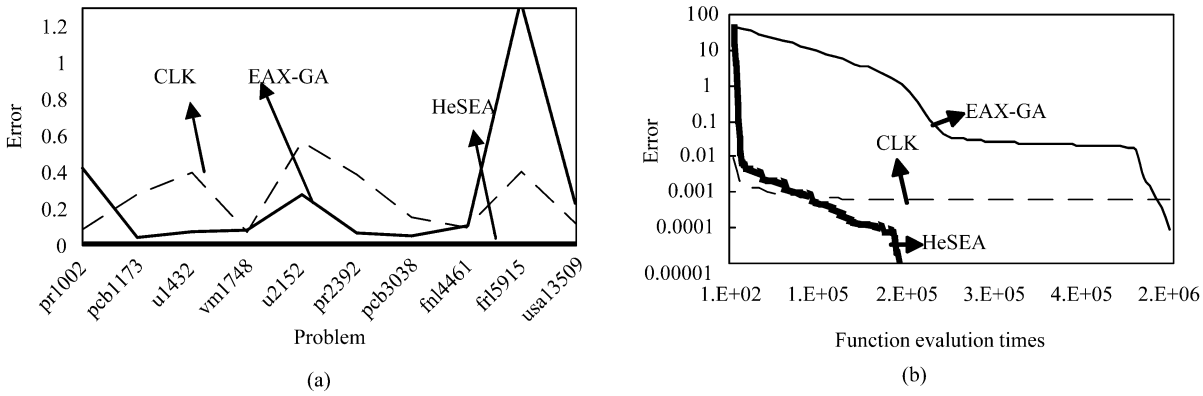

Fig. 10. Comparisons of HeSEA, EAX-GA, and CLK. "Error" is defined in (1). (a) Presents the results of testing these methods on eight TSP benchmarks. (b) Shows the search behaviors of these methods when applied to solving problem *fnl4461*. These results indicate that HeSEA yields better solutions than the approaches to which it was compared, as it takes advantages of the explorative ability of EAX and the heuristic power of CLK.

ities of these two operators to preserve "good edges" are almost identical [Fig. 8(b)], implying that they can each maintain the "good edges" generated by each other. These two operators can add "good edges" without reducing the capacity to preserve the "good edges" added by the other one. Although the seamlessness of combining EAX and LK cannot be proven theoretically, each operator compensates for the other in adding and preserving "good edges," according to the experiments conducted herein.

Fig. 9(a) and (b) shows that LK is more powerful than EAX. The improvements in the tours generated by EAX and LK are of interest. In each generation, the average improvements obtained using EAX and LK are defined as $(1/N)\sum_{i=1}^{N}(f(s_i)-f(I_i))$ and $(1/N)\sum_{i=1}^{N}(f(I_i)-f(c_i))$, respectively, where N is the population size; f is the tour length; s_i represents the ith individual in the population; I_i is the tour generated by applying EAX on s_i, and c_i is obtained by modifying I_i using LK.

Fig. 9(c) shows the improvements yielded by EAX and LK in the proposed approach. These two operators can improve the tour generated by each other. LK yields larger improvements than EAX in the earlier stage, while EAX produces higher improvements in the later stage. The experimental results indicate that LK can deliver a good solution rapidly and EAX can help LK escape from local optima. This phenomenon is consistent with their behaviors as shown in the following.

The efficiency of the proposed system is examined by comparison with the original EAX-GA [12] and CLK [43]. The original EAX-GA applies EAX and RpS selection, while CLK iteratively adopts LK only. To make the comparison fair, all parameters are set to the same values as in the papers in which the systems were originally proposed. The population size of EAX-GA is set to the number of cities. In HeSEA, the population size and family competition length are set to 100 and 20, respectively, for all problems.

Fig. 10(a) summarizes the results of HeSEA, EAX-GA and CLK applied to eight benchmarks. These findings indicate that HeSEA performs more robustly than EAX-GA and CLK for test problems. With an equal number of function evaluations $(1000M)$, the proposed system yields better solutions than the other two methods when tested on the test benchmarks. The results of EAX-GA are slightly better than those of CLK when applied to smaller problems. For larger problems, such as *frl5915* and *usa13509*, EAX-GA underperformed CLK since EAX-GA requires more search time. The fitness values of the tours generated by these three methods were traced to understand their search behaviors. Fig. 10(b) presents the results of these three methods as tested on problem *fnl4461*. CLK can find good solutions efficiently in the early search stage, whereas, EAX-GA improves tours slowly and may be able to escape from local optima to achieve better solutions.

TABLE III
COMPARISONS OF HESEA WITH EAX-LK (SEE TEXT), EAX-GA AND CLK, APPLIED TO TEN TSP PROBLEMS, BASED ON THE BEST AND AVERAGE TOUR LENGTH. THE AVERAGE *ERROR* (1) IS GIVEN IN PARENTHESES AND (−) INDICATES THE AVERAGE ERROR IS ZERO

problem		HeSEA	EAX-LK	EAX-GA	CLK
pr1002	best	259045 (-)	259045 (-)	259985(0.3629)	259045(-)
(259045)	ave.	259045 (-)	259053 (0.0031)	260142.1(0.4235)	259270.6(0.0871)
pcb1173	best	56892(-)	56892(-)	56897(0.0088)	56893(0.0018)
(56892)	ave.	56892(-)	56892(-)	56916.6(0.0432)	57050.8(0.2791)
u1432	best	152970(-)	152970(-)	153012(0.0275)	153340(0.2419)
(152970)	ave.	152970(-)	152977.5(0.0049)	153086.3(0.0760)	153580.3(0.3990)
vm1748	best	336556(-)	336556(-)	336679(0.0365)	336737(0.0539)
(336556)	ave.	336556(-)	336582.5(0.0078)	336840.7(0.0846)	336803.9(0.0737)
u2152	best	64253(-)	64253(-)	64377(0.1930)	64486(0.3626)
(64253)	ave.	64253(-)	64253(-)	64432.2(0.2789)	64617(0.5665)
pr2392	best	378032(-)	378032(-)	378059(0.0071)	378592(0.1481)
(378032)	ave.	378032(-)	378032(-)	378286.9(0.0674)	379500.5(0.3885)
pcb3038	best	137694(-)	137694(-)	137717(0.0167)	137791(0.0070)
(137694)	ave.	137694(-)	1377700.3(0.0046)	137765.6(0.0520)	137906.7(0.1545)
fnl4461	best	182566(-)	182567(0.0005)	182632(0.0362)	182662(0.0526)
(182566)	ave.	182566.9(0.0005)	182592.6(0.015)	182760.0(0.1068)	182745.2(0.0982)
frl5915	best	565530(-)	565557 (0.0048)	571942(1.1338)	567555(0.3581)
(565530)	ave.	565530.5(0.0001)	565621.6(0.016)	572933(1.3090)	567817(0.4044)
usa13509	best	19983361(0.0025)	19985355(0.0125)	20018705(0.1794)	20001221(0.0919)
(19982859)	ave.	19984334(0.0074)	19988101(0.0262)	20026761(0.2197)	20006298(0.1173)

D. Efficiency of Integrating HpS and Family Competition

HeSEA was compared with EAX-LK, which replaces HpS and family competition with a random-pairing scheme and total replacement with elitism, respectively, to confirm the effectiveness of HpS and family competition. HeSEA was also compared to EAX-GA [12] and CLK [43] to evaluate the performance delivered by combining EAX and LK. Table III shows the experimental results concerning the application of these four methods to ten TSP benchmarks. The running times of all comparative methods are all set to the running time of HeSEA (Table IV).

In Table III, EAX-GA outperforms CLK when the problem size is under 4000. For problems *frl5915* and *usa13509*, EAX-GA underperforms CLK because EAX-GA may require more time to converge. EAX-LK, combining EAX and LK with random pairing and total replacement, performed better than both EAX-GA and CLK in all test TSPs. However, when the number of cities exceeds 4000, EAX-LK cannot find the optimal solutions and the average error exceeds HeSEA. HeSEA, integrating EAX and LK through HpS and family competition, not only stabilized the search behaviors but also yielded the solutions of highest quality of all methods, when applied to all tested problems, especially large ones.

In summary, the novel method seems to be able to seamlessly integrate EAX and LK to solve TSPs, as confirmed by four observations.

1) They can compensate for each other in different search stages [Figs. 8(c) and 9(b)]; that is, LK can accelerate EAX, while EAX helps LK to escape from local optima.
2) They can add "good edges" without reducing the ability to preserve the "good edges" obtained by the other [Fig. 8(a) and (b)].
3) Heterogeneous pairing selection can maintain the diversity of the population (Fig. 8).
4) The family competition and HpS in HeSEA can significantly improve the solutions for large TSPs (Table III).

V. EXPERIMENTAL RESULTS AND CONCLUSION

The proposed approach was tested on 16 TSP benchmark problems in which the numbers of cities ranged from 318 to 13 509. The method was implemented on a Pentium IV 1.2 GHz personal computer with a single processor and 1 GB RAM. Each problem was run 20 times. As in the analysis in Section IV, the population size and family competition length were set to 100 and 20, respectively. Table IV presents the experimental results obtained by applying HeSEA to these 16 problems. HeSEA found the optimal tours of all tested problems except problem *usa13509*. The optimal solution to the small problems (in which the number of cities is less than 4000) is determined in each trial. For large problems, such as *fnl4461* and *frl5915*, HeSEA found the optima at least 16 times out of 20 independent trials. The best solution found by HeSEA to problem *usa13509* is 19983361, which is only 0.0025% above the optimum (19 982 859). For each test problem, the average tour length does not exceed 0.0074% above the optima.

HeSEA was compared to six methods, including the ant colony system (ACS) [44], the Voronoi crossover genetic algorithm (VGA) [16], the compact genetic algorithm (CGA) [24], iterated LK (ILK) [20], Tabu search with LK (TLK) [45] and multitrial LK (LKH) [40]. These methods performed well on these test problems, according to a survey of the relevant literature. ACS uses an ant colony system with a 3-opt operator. VGA integrates the Voronoi-quantized crossover with the LK local search. CGA mimics the existence of solutions and combined LK local search. ILK iteratively applies LK local search to a range of initial tours. TLK uses the Tabu search with LK local search, and LKH modifies the original LK by

TABLE IV
EXPERIMENTAL RESULTS OBTAINED BY APPLYING HESEA TO 16 TSP PROBLEMS, INCLUDING AVERAGE TIME, AVERAGE NUMBER OF GENERATIONS, TOUR LENGTH, AND NUMBER OF TIME THE OPTIMAL TOUR IS IDENTIFIED IN 20 INDEPENDENT RUNS. THE AVERAGE ERROR IS AS GIVEN IN (1) AND $(-)$ REPRESENTS THAT THE AVERAGE ERROR IS ZERO

problem	time (sec.)	gen.	best tour length (error %)	avg. tour length (error %)	optimal times
lin318 (42029)	12.4	3.2	42029(-)	42029(-)	20
pcb442 (50778)	9.2	3.2	50778(-)	50778(-)	20
att532 (27686)	15.3	6.1	27686 (-)	27686 (-)	20
u574 (36905)	23.6	4.8	36905(-)	36905(-)	20
rat783 (8806)	39.1	8.4	8806(-)	8806(-)	20
pr1002 (259045)	91	12	259045 (-)	259045 (-)	20
vm1084 (239297)	80.6	10.2	239297 (-)	239297 (-)	20
pcb1173 (56892)	84.5	11.5	56892(-)	56892(-)	20
u1432 (152970)	107	11	152970(-)	152970(-)	20
vm1748 (336556)	141	11.5	336556(-)	336556(-)	20
u2152 (64253)	211	17.5	64253(-)	64253(-)	20
pr2392 (378032)	208	14.5	378032(-)	378032(-)	20
pcb3038 (137694)	612	29.7	137694(-)	137694(-)	20
fnl4461 (182566)	2349	67.8	182566(-)	182566.9(0.0005)	16
frl5915 (565530)	2773	71	565530 (-)	565530.5(0.0001)	19
usa13509 (19982859)	34984	223	19983361(0.0025)	19984334(0.0074)	0

TABLE V
COMPARISON OF THE PROPOSED METHOD (HESEA) WITH SIX METHODS—THE ANT COLONY SYSTEM (ACS) [44], THE VORONOI CROSSOVER GENETIC ALGORITHM (VGA) [15], THE COMPACT GENETIC ALGORITHM (CGA) [24], ITERATIVE LK (ILK) [20], MULTI-TRIAL LK (LKH) [40] AND TABU SEARCH WITH LK [45]—AS APPLIED TO 14 TSP PROBLEMS, IN TERMS OF THE AVERAGE TOUR LENGTH AND CPU TIME. THE LENGTH OF THE OPTIMAL TOUR OF EACH PROBLEM IS GIVEN IN PARENTHESES IN THE FIRST COLUMN. THE ERROR IS AS DEFINED IN (1). EACH METHOD WAS IMPLEMENTED ON A DIFFERENT MACHINE (SEE TEXT)

problem		HeSEA	ACS	VGA	CGA	ILK	Tabu with LK	LKH
lin318 (42029)	error (%)	(-)	(-)	(-)	(-)	N/A	N/A	(0.1085)
	time (sec.)	2.3	537	29	12.1			1.4
att532 (27686)	error (%)	(-)	(0.1163)	(0.0025)	(-)	N/A	N/A	(-)
	time (sec.)	9.2	810	109	112			6.9
rat783 (8806)	error (%)	(-)	(0.3622)	N/A	(-)	N/A	N/A	(-)
	time (sec.)	15.3	1280		111			2.2
pr1002 (259045)	error (%)	(-)	N/A	N/A	(-)	(0.1482)	(0.8794)	(-)
	time (sec.)	91			145	298	1211.4	7.5
vm1084 (239297)	error (%)	(-)	N/A	N/A	N/A	(0.0217)	(0. 3932)	(0.0068)
	time (sec.)	80.6				377	597	12.6
pcb1173 (56892)	error (%)	(-)	N/A	N/A	N/A	(0.0088)	(0.6996)	(0.0009)
	time (sec.)	84.5				159	840	11.8
u1432 (152970)	error (%)	(-)	N/A	N/A	N/A	(0.0994)	(0.4949)	(-)
	time (sec.)	107				224	775	6.9
u2152 (64253)	error (%)	(-)	N/A	N/A	(-)	(0.1743)	(0.7517)	(0.0495)
	time (sec.)	211			1772	563	1624	135
pr2392 (378032)	error (%)	(-)	N/A	N/A	N/A	(0.1495)	(0.6492)	(-)
	time (sec.)	208				452	1373	26.2
pcb3038 (137694)	error (%)	(-)	N/A	(0.0093)	N/A	(0.1213)	(0.8708)	(0.0068)
	time (sec.)	612		906		572	1149	226
fnl4461 (182566)	error (%)	(0.0005)	N/A	(0.0219)	(0.0068)	(0.1358)	(0.9898)	(0.0027)
	time (sec.)	2349		2057	33887	889	1018	528
frl5915 (565530)	error (%)	(0.0001)	N/A	N/A	(0.0042)	(0.0168)	(0.9053)	(0.0350)
	time (sec.)	2773			34817	2615	4211	1301
usa13509 (19982859)	error (%)	(0.0074)	N/A	N/A	(0.0437)	(0.1638)	(0.8897)	(0.0065)
	time (sec.)	34984			137 hr.	10694	5852	19573
d15112 (1573084)	error (%)	(0.0137)	N/A	N/A	N/A	(0.0205)	(0.8339)	(0.0192)
	time (sec.)	62371				8378	3592	34759

improving its search abilities. The results obtained using these methods were directly taken from original studies or the results of the "8th DIMACS Implementation Challenge: The Traveling Salesman Problem" (http://www.research.att.com/~dsj/chtsp/).

The average error (%) and the average time (in seconds) required to solve the problem are used as measures of the performance of the compared methods. The values in parentheses represent the percentage average error as defined in (1). The *av-*

erage CPU time is only for reference, since each approach is executed on different machines. The LKH package was downloaded and executed on a single machine (Pentium IV 1.2 GHz personal computer). ACS was executed on an SGI Challenge L server with 200 MHz CPU; VGA was executed on a Pentium III 866 MHz; CGA was executed on a Pentium III 350 MHz; ILK was executed on a Silicon Graphics 196 MHz MIPS R1000, while TLK was executed on a Pentium III 800 MHz. Table V reveals that the speed of HeSEA is satisfactory and comparable with that of other methods for solving large TSPs, such as problems *usa13509* and *d15112*.

Table V compares the results obtained using HeSEA with those obtained using the surveyed methods. HeSEA yields results comparable with those yielded by other approaches when applied to these test problems. With respect to the quality of the solution, HeSEA can find the optimal solution in each trial in solving problems that involve fewer than 4 000 cities. For the large problems, the average solution obtained by the proposed method is less than 0.014% above the optima value. A comparison of HeSEA to the heuristic methods ILK and Tabu, reveals that the proposed method is somewhat slower than the others; however, the quality of the solution to each problem is higher than obtained by the two heuristic approaches. VGA and CGA, two GA-based approaches, yield better solutions than heuristics but slightly worse solutions than the proposed method. Nevertheless, LKH is faster than the proposed method. An advantage of HeSEA is that the quality of the solution seems to be more stable than that obtained by LKH, when used to solve these test problems. In summary, HeSEA is a robust method for solving TSPs within a reasonable time.

This work demonstrates that HeSEA is a stable method for solving TSP's. An evolutionary algorithm for solving TSPs should comprise mechanisms for preserving "good edges" and inserting new edges into offspring, as well as mechanisms for maintaining the population diversity. In the proposed approach, edge assembly crossover and Lin–Kernighan local search preserve "good edges" and add new edges. The proposed method can seamlessly integrate EAX and LK to improve the overall search. Heterogeneous edge selection and family competition maintain the diversity of the population and help to generate good children with a high probability.

Experiments on 16 benchmark TSPs confirm that the proposed approach is robust and is highly competitive against the algorithms found in a survey of the literature. The proposed method can yield stable solutions to all test TSPs; specifically, it finds the optimum in each run in which the number of cities is under 4 000, and in 16 of 20 independent runs to solve problems *fnl4461* and *frl5915*. The average solution to problem *usa13509* is only 0.0074% above the optimum. We believe that HeSEA is a robust tool for solving large TSPs and has potential applications.

REFERENCES

[1] G. Clarke and J. W. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Oper. Res.*, vol. 12, pp. 568–581, 1964.

[2] D. Whitely, T. Starkweather, and F. D'Ann, "Scheduling problems and traveling salesman: The genetic edge recombination operator," in *Proc. 3rd Int. Conf. Genetic Algorithms*, 1989, pp. 133–140.

[3] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 498–516, 1983.

[4] F. Alizadeh, R. M. Karp, L. A. Newberg, and D. K. Weisser, "Physical mapping of chromosomes: A combinatorial problem in molecular biology," in *Proc. 4th ACM-SIAM Symp. Discrete Algorithms (SODA)*, 1993, pp. 52–76.

[5] C. Korostensky and G. H. Gonnet, "Using traveling salesman problem algorithms for evolutionary tree construction," *Bioinformatics*, vol. 16, no. 7, pp. 619–627, 2000.

[6] H. J. Bremermann, M. Rogson, and S. Salaff, "Search by evolution," in *Biophysics and Cybernetic Systems*. Washington, DC: Spartan, 1965, pp. 157–167.

[7] I. M. Oliver, D. J. Smith, and J. R. C. Holland, "A study of permutation crossovers on the TSP," in *Proc. 2nd Int. Conf. Genetic Algorithm Their Applications*, 1987, pp. 224–230.

[8] *Handbook Genetic Algorithms*, L. Davis, Ed., Van Nostrand, New York, 1991, pp. 332–349.

[9] H. Mühlenbein, M. Gorges-Schleuter, and O. Krämer, "Evolution algorithms in combinatorial optimization," *Parallel Comput.*, vol. 7, pp. 65–85, 1988.

[10] D. E. Goldberg and R. Lingle, Jr, "Alleles, loci and the TSP," in *Proc. 1st Int. Conf. Genetic Algorithms Their Applications*, 1985, pp. 154–159.

[11] L. Davis, "Applying adaptive algorithms to epistatic domains," in *Proc. Int. Joint Conf. Artificial Intelligence*, 1985, pp. 162–164.

[12] Y. Nagata and S. Kobayashi, "Edge assembly crossover: A high-power genetic algorithm for the traveling salesman problem," in *Proc. 7th Int. Conf. Genetic Algorithms (ICGA)*, 1997, pp. 450–457.

[13] J. J. Grefenstette, "Incorporating problem specific knowledge into genetic algorithms," *Genetic Algorithms Simulated Annealing*, pp. 42–60, 1987.

[14] B. Freisleben and P. Merz, "New genetic local search operators for the traveling salesman problem," in *Proc. Parallel Problem Solving Nature IV (PPSN IV)*, 1996, pp. 890–899.

[15] G. Tao and Z. Michalewicz, "Inver-over operator for the TSP," in *Parallel Problem Solving From Nature V (PPSN V)*, 1998, pp. 803–812.

[16] D. Seo and B. R. Moon, "Voronoi quantized crossover for traveling salesman problem," in *Proc. Genetic Evolutionary Computation Conf. (GECCO)*, 2002, pp. 544–552.

[17] S. Jung and B. R. Moon, "The nature crossover for the 2D euclidean TSP," in *Proc. Genetic Evolutionary Computation Conf. (GECCO)*, 2000, pp. 1003–1010.

[18] P. A. Moscato, "On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Toward Memetic Algorithms," Caltech, Pasadena, CA, Tech. Rep. California Technical Institute concurrent computation program report 826, 1989.

[19] P. Moscato and M. G. Norman, "A "Memetic" approach for the traveling salesman problem: Implementation of a computational ecology for combinatorial optimization on message-passing systems," *Parallel Comput. Transput. Applicat.*, pp. 187–194, 1992.

[20] 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. New York: Wiley, 1997, pp. 215–310.

[21] P. Merz and B. Freisleben, "Genetic local search for the TSP: New results," in *Proc. IEEE Int. Conf. Evolutionary Computation (CEC)*, 1997, pp. 159–164.

[22] ——, "Memetic algorithms for the traveling salesman problem," *Complex Syst.*, vol. 13, no. 4, pp. 297–345, 2001.

[23] S. Ronald, "Preventing diversity loss in a routing genetic algorithm with hash tagging," *Complexity Int.*, vol. 2, 1995.

[24] R. Baraglia, J. I. Hidalgo, and R. Perego, "A hybrid heuristic for the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 5, pp. 613–622, Dec. 2001.

[25] L. J. Eshelman and J. D. Schaffer, "Preventing premature convergence in genetic algorithms by preventing incest," in *Proc. 4th Int. Conf. Genetic Algorithms (ICGA)*, 1991, pp. 115–122.

[26] K. T. Mak and A. J. Morton, "Distances between traveling salesman tours," *Discrete Appl. Math.*, vol. 58, pp. 281–291, 1995.

[27] G. Reinelt, *The Traveling Salesman: Computational Solutions for TSP Applications*. New York: Springer-Verlag, 1994, vol. 840, Lecture Notes in Computer Sciences.

[28] N. Radcliffe and P. Surry, "Fitness variance of formae and performance prediction," in *Proc. 3rd Workshop Foundations Genetic Algorithms*, 1994, pp. 51–72.

[29] G. Reinelt, "TSPLIB-a traveling salesman problem library," *ORSA J. Comput.*, vol. 3, no. 4, pp. 376–384, 1991.

[30] J. E. Baker, "Reducing bias and inefficiency in the selection algorithm," in *Proc. 4th Int. Conf. Genetic Algorithms*, 1987, pp. 14–21.

[31] S. Lin, "Computer solutions of the traveling salesman problem," *Bell Syst. Tech. J.*, vol. 23, pp. 2245–2269, 1965.
[32] H. K. Tsai, J. M. Yang, and C. Y. Kao, "Solving traveling salesman problems by combining global and local search mechanisms," in *Proc. Congr. Evolutionary Computation*, 2002, pp. 1290–1295.
[33] S. Lin and B. Kernighan, "An effective heuristic algorithms for the traveling salesman problem," *Oper. Res.*, vol. 21, pp. 498–516, 1973.
[34] S. H. Chen, S. F. Smith, and S. C. Guerra, "The GENIE is out! (who needs fitness to evolve?)," in *Proc. Congr. Evolutionary Computation (CEC)*, 1999, pp. 2102–2106.
[35] J. Watson, C. Ross, V. Eisele, J. Denton, J. Bins, C. Guerra, D. Whitely, and A. Howe, "The traveling salesrep problem, edge assembly crossover, and 2-opt," in *Proc. Parallel Problem Solving From Nature V (PPSN V)*, 1998, pp. 823–832.
[36] Y. Nagata and S. Kobayashi, "An analysis of edge assembly crossover for the traveling salesman problem," in *Proc. IEEE Int. Conf. Systems Man, Cybernetics*, 1999, pp. 628–633.
[37] ——, "An proposal and evaluation of new crossover "Edge assembly crossover" for the traveling salesman problem," *J. Jpn. Soc. Artif. Intell.*, vol. 14, no. 5, pp. 848–859, 1999.
[38] H. K. Tsai, J. M. Yang, and C. Y. Kao, "A genetic algorithm for traveling salesman problems," in *Proc. Genetic Evolutionary Computation Conf. (GECCO)*, 2001, pp. 687–693.
[39] D. Applegate, R. Bixby, V. Chvatal, and W. J. Cook, "Finding Tours in the TSP," Dept. Computat. Appl. Math,, Rice University, Houston, TX, Tech. Rep. TR99-05, 1999.
[40] K. Helsgaun, "An effective implementation of the Lin-Kernighan traveling salesman heuristic," *Eur. J. Oper. Res.*, vol. 126, no. 1, pp. 106–130, 2000.
[41] O. Martin, S. W. Otto, and E. W. Felten, "Large step Markov chain for the traveling salesman," *Complex Syst.*, vol. 5, no. 3, pp. 229–326, 1991.
[42] O. Martin and S. W. Otto, "Combining simulated annealing with local search heuristic," *Ann. Oper. Res.*, vol. 63, pp. 57–75, 1996.
[43] Concorde Package. [Online]. Available: http://www.math.princeton.edu/tsp/concorde/co991 215.tgz
[44] T. Stützle and M. Dorigo, "ACO algorithms for the traveling salesman problem," *Evol. Alg. Eng. Comput. Sci.*, pp. 163–183, 1999.
[45] M. Zachariasen and M. Dam, "Tabu search on the geometric traveling salesman problem," in *Proc. Metaheuristics Int. Conf.*, 1995, pp. 571–587.

Huai-Kuang Tsai received the B.S., the M.S., and the Ph.D. degrees in computer science and information engineering from the National Taiwan University, Taipei, Taiwan, R.O.C. in 1996, 1998, and 2003, respectively.

Since October 2003, he has been postdoctoral Fellow at National Taiwan University. His research interests include evolutionary computation, bioinformatics, combinatorial optimization, and data mining.

Jinn-Moon Yang received the M.S. degree from the National Central University, Chung-Li, Taiwan, R.O.C., in 1994, the M.B.A. degree from Tamkang University, Taipei, Taiwan, and the Ph.D. degree in computer science and information engineering from National Taiwan University, Taipei, Taiwan, R.O.C. in 2001.

From 1985 to 1998, he was with for PanChiao Telecommunications, Ministry of Transportation and Communications. From 1998 to 2001, he worked for Chunghwa Telecom Training Institute, Taipei. He has been an Assistant Professor with the Department of Biological Science and Technology Computer Science and Institute of Bioinformatics, National Chiao Tung University, Taipei, since 2001. He has published more than 40 technical papers in various journals and conference records. His research interests include evolutionary computation, bioinformatics, structural biology, rational drug-design, and machine learning.

Yuan-Fang Tsai received the B.S., the M.S., and the Ph.D. degrees from the Department of Hydraulics and Ocean Engineering, National Cheng Kung University, Tainan, in 1992, 1994, and 1999, respectively.

He has been an Assistant Professor in the Department of Social Studies Education, National Taipei Teachers College, Taipei, Taiwan, since 2004. His major interests are in hydraulics engineering, numerical simulation, bioinformatics, and evolutionary computation. He has published more than 30 technical papers in various journals and conference records.

Cheng-Yan Kao was born in Taipei, Taiwan, R.O.C. in 1948. He received the B.S. degree in mathematics from National Taiwan University, Taipei, in 1971, and the M.S. degree in computer science, the M.S. degree in statistics, and the Ph.D. degree in computer science, from the University of Wisconsin, Madison, in 1976, 1978, and 1981, respectively.

He was with the Ford Aerospace and the Unisys Corporation and for General Electric from 1980 to 1989 at the Johnson Space Center, NASA, Houston, TX. He has been a Professor with the Department of Computer Science and Information Engineering, National Taiwan University, since 1990. He has published more than 40 technical papers in various journals and conference records. His research interests include evolutionary computation, bioinformatics, optimization, and artificial intelligence.