# Improving EAX with Restricted 2-opt


Chen-hsiung Chan
frankch@life.nthu.edu.tw

Sheng-An Lee
shengan@lis.idv.tw

Cheng-Yan Kao
cykao@csie.ntu.edu.tw

Department of Computer Science and Information Engineering
National Taiwan University
No. 1, Sec. 4, Roosevelt Road, Taipei 10617, Taiwan
+886-2-2362-5336 ext. 401

Huai-Kuang Tsai
d7526010@csie.ntu.edu.tw

Genomic Research Center
Academia Sinica
No. 128, Sec. 2, Academia Road, Nankang, Taipei, 115, Taiwan



## ABSTRACT

Edge Assembly Crossover (EAX) is by far the most successful crossover operator in solving the traveling salesman problem (TSP) with Genetic Algorithms (GAs). Various improvements have been proposed for EAX in GA. However, some of the improvements have to make compromises between performance and solution quality. In this work, we have combined several improvements proposed in the past, including heterogeneous pair selection (HpS), iterative child generation (ICG), and 2-opt. We also incorporate 2-opt into EAX, and restricted the 2-opt local searches to sub-tours in the intermediates generated by EAX.

Our proposed method can improve the performance of EAX with decreased number of generations, error rates, and computation time. The applications of conventional 2-opt and our restricted 2-opt concurrently have additive effect on the performance gain, and this performance improvement is more obvious in larger problems. The proposed method also enhanced the solution quality of EAX. The significances of the restricted 2-opt and the conventional 2-opt in EAX were analyzed and discussed.



## Categories and Subject Descriptors

Genetic Algorithm.

## General Terms

Algorithms.

## Keywords

Combinatorial optimization, Local Search, Genetic Algorithms, Traveling Salesman Problem (TSP), Edge Assembly Crossover (EAX), Restricted 2-opt.




*GECCO'05*, June 25-29, 2005, Washington, DC, USA.



## 1. INTRODUCTION

The traveling salesman problem (TSP) is a well-known NP-hard optimization problem. The problem designates $n$ vertices as $n$ cities, and tries to find the shortest round tour visiting each city exactly once. Many problems in various fields can be formulated as TSP. For example, scheduling [1], physical mapping [2], and even protein folding [3] can be treated as TSP. Genetic algorithms (GAs) have been widely used in tackling TSP. Many works have tried to obtain the optimum tour lengths of TSPs and to minimize the computing costs with GAs.

Various operations have to be considered when using GAs to solve TSP. These include crossover, mutation, and selection. Selection in GA including selection of parent pairs and selection for survival. In most GAs, the selection for survival process simply picks the tour with shortest tour length; but more complex selection schemes are also used. The crossover operators are more challenging, because simply swap segments in two tours will not produce valid solutions to TSP. Among the numerous crossover operators, edge-based operators, i.e. EX [4], EXX [5], and EAX [6], are of particular interest. Many researches have found that EAX performs well, and a number of improvements have been proposed. Edge-based operators may lead to diversity loss of the population. That is, the children will tend to become more similar to one of the parents. Mutation operations in GAs are intended to introduce diversity into the population. A neighbor-join (NJ) operator [7] has been shown to perform well, and can enhance the performance of GA with EAX operator.

In this work, we proposed a new operator which incorporate mutation operations into EAX. The 2-opt local search has been applied to intermediates of tours generated in EAX. Unlike other improvements which either apply mutation operator to generated tours, or use 2-opt to generate initial population, our proposed method restricted the positions of 2-opt to sub-tours formed during EAX operation. The restricted 2-opt operator can increase the diversity within solutions generated by EAX. The edge replacements in EAX algorithm tend to generate children similar to one of the parents. If these children replace the dissimilar parent, the remaining individuals will become more similar, and cause the diversity loss in the population. Selection scheme has been proposed by Nagata [8] to consider diversity loss, but the proposed scheme requires more evaluations and comparisons, and is more computation intensive. Our proposed method not only introduces diversity into the population, but also decreases the overall computation time.

The rest of the paper is organized as follows. In Section 2, we will describe the edge assembly crossover (EAX) algorithm, and some issues within this algorithm. Section 3 will describe how to incorporate 2-opt local search into EAX, and the proposed method, along with some preliminary results. Finally, the proposed method will be applied to several benchmark problems. These results will be presented and discussed.

# 2. EDGE ASSEMBLY CROSSOVER (EAX)

The EAX algorithm was proposed by Nagata *et al*. [6]; the algorithm can be divided into several steps, which will be shortly described as follows.

**[The EAX algorithm]**

1. Two tours, A and B, will be selected for crossover. The common edges and different edges on the two tours will be identified.

2. *AB_cycles* with intermittent edges from tour A and B are formed. *AB_cycles* are closed graphs with edges from tour A and tour B alternatively. The formations of *AB_cycles* are not unique; the *AB_cycles* are constructed randomly. Details on *AB_cycle* construction have been described extensively by Nagata *et al.* [6] and Watson *et al.* [9].

3. Sets of *AB_cycles* will be selected to form an E-set. An E-set contains some but not all *AB_cycles*. Various selection schemes are possible for construction of E-set.

4. The E-set will be applied to parent A. Edges from parent A on E-set will be removed, and edges from parent B will be added. This will form an intermediate solution with several disjoint sub-tours.

5. The sub-tours in the intermediate will be modified into valid solution.

The EAX algorithm is by far the most powerful genetic algorithm operator for solving TSP. It is curious whether the EAX algorithm can be further improved. From the algorithm described above, possible improvements can be introduced at specific steps in the algorithm. For example, the construction of *AB_cycles* is not unique. Current algorithm constructs *AB_cycles* randomly. It is possible to construct *AB_cycles* based on some criteria or heuristics for better crossover. Also the selection of *AB_cycles* to form an E-set can be manipulated. Finally, the modification of intermediate to valid solution can be improved. Some issues of EAX will be shortly discussed, and previously proposed improvements will be summarized below.

## 2.1 Construction of E-sets

An E-set is a collection of *AB_cycles*. Different criteria can be used in the selection of *AB_cycles* to construct an E-set. In Nagata's original work on EAX [6], two criteria are proposed. The first criterion is simply selecting *AB_cycles* with 0.5 probability, therefore roughly half of the total *AB_cycles* will be selected. This is denoted as *EAX-Rand*. The other criterion considers the balance between GAIN (change in tour length) and DIV (frequency of a edge in the population), and can be called as *EAX-heuristic*. The weights of the two parameters (GAIN and DIV) can be adjusted; however, in current implementation and the one described by Nagata *et al.* [6] the two parameters contributed equally with equal weights. Whether changes on the weights would have effect on the performance of EAX is yet to be determined. In Nagata's doctorial thesis [10], another scheme is proposed, which used a single *AB_cycle* to construct an E-set. This method is called *EAX-1AB*. The use of single *AB_cycle* will make the child near tour A, only a small number of edges are removed and added. The performance of *EAX-1AB* has been shown to be better than *EAX-Rand* [8], since only a small number of modifications are required.

## 2.2 Modification of Intermediates to Valid Tours

The E-sets are used to mark edges to be removed from or added to the parent. The *AB_cycles* in the E-set will be applied to one of the parent. The parent is arbitrarily selected. If these *AB_cycles* were applied to parent A, edges from parent A will be removed, and edges from parent B will be added into parent A. After the removal and addition of edges, several disjoint sub-tours will form the intermediate. It is necessary to modify the intermediate so as the final tour is a valid solution to TSP. Nagata *et al*. [6] used a greedy method with heuristics to construct the final tour. The modification will select two edges from two disjoint tours, removed the two edges and reconnected the two sub-tours. They tried to minimize changes in overall tour length during the modification process. Watson *et al*. [9] has extend the heuristics used by Nagata *et al*. [6] to exhaustive enumeration of all edges in two disjoint sub-tours. It is interesting to note that not much effort has been put into the modification step of EAX. That is, the selection of which two sub-tours to join. Nagata *et al.* [6] used a minimum spanning tree for joining these sub-tours. Alternative schemes for joining these sub-tours are also possible and require further investigations. However, these sub-tours may be suitable targets for local optimization. Applying local search operators on these sub-tours may further increase the speed of convergence. This will be elaborated in Section 3.

## 2.3 Enhancements to EAX

Tsai *et al*. [11] has proposed a parent selection method, called heterogeneous selection evolutionary algorithm (HeSEA). HeSEA use the concept of family competition and heterogeneous paring selection (HpS). In HpS, the parent pairs are selected for individuals sharing edges less than a criterion, which is the average number of common edges in a population. The use of HpS can avoid crossover of highly similar parents, and retain the diversity of the population.

Nagata [8] has proposed a new method to handle diversity loss usually observed in EAX. A number of children are generated using *EAX-Rand*, *EAX-1AB*, or *EAX-dMSXF* [12]. The selection for survival did not directly rely on the tour lengths of the children. A new evaluation function is used. This evaluation function considers the change in tour length between parent A and child C, and also the number of different edges among parent A, B, and child C [8]. This method is called *EAX-Dis*. *EAX-Dis* can obtain more optimum solutions and smaller errors comparing to *EAX-Rand*, *EAX-1AB*, and *EAX-dMSXF*. However, *EAX-Dis* requires more generations to converge. Also, more children are generated for evaluation in each generation, therefore the computing cost of *EAX-Dis* is higher than other selection schemes.

Watson *et al*. [9] have implemented EAX and attempted to enhance EAX performance. Two methods are proposed. First, they used iterative child generation (ICG). In ICG, if *EAX-heuristic* failed to produce child with better tour length, a new child is generated using *EAX-Rand*. The process continues until a better child is generated or the specified iteration limit is reached. The second enhancement is the use of 2-opt as an mutation operator. The 2-opt local search was applied to individuals in the new generation. Both methods can improve the performance of EAX with smaller error rates.

## 3. PROPOSED IMPROVEMENT

In this work, we proposed incorporation of 2-opt operator into EAX. Previous works by Watson *et al*. [9] apply 2-opt after the children are generated and replaced their parents. The 2-opt searches are performed on the entire tour. In this work, the 2-opt operators are applied to sub-tours in the intermediates generated by EAX. This operator is called restricted 2-opt because the 2-opt searches are performed on restricted regions of the graph.

The proposed method include the following major steps.

**[Proposed Method]**

1. Selection of parent pairs using HpS.

2. Crossover with *EAX-1AB/EAX-Rand* using ICG. The first child is generated with *EAX-1AB*, and consequent children are generated with *EAX-Rand* iteratively. The limit of iteration number is 20. That is, at most 20 children will be generated for evaluation within one crossover.

3. During EAX crossover, 2-opt local search is applied to sub-tours generated in EAX intermediates. These intermediates were generated after the *AB_cycles* have been applied to one of the parent, but haven't been modified into valid tours. The scope of these 2-opt searches are restricted within the sub-tours. For each sub-tours, 100 2-opt operations are performed.

4. After the child is generated, conventional 2-opt searches are applied to the entire tour of child. For each child, 100 2-opt operations are performed.

**Table 1. The enhancements of restricted 2-opt to EAX.**

| | | Err. | Gen. | Opt. | Time (s) |
|---|---|---|---|---|---|
| eil101 | EAX | 0.006% | 39.98 | 48/50 | 0.391 |
| | 2-opt | 0.000% | 25.08 | 50/50 | 0.394 |
| kroA200 | EAX | 0.055% | 80.96 | 33/50 | 1.558 |
| | 2-opt | 0.027% | 57.06 | 44/50 | 1.414 |
| lin318 | EAX | 0.233% | 116.30 | 12/50 | 4.004 |
| | 2-opt | 0.103% | 87.92 | 24/50 | 3.697 |
| pcb442 | EAX | 0.020% | 136.42 | 26/50 | 7.067 |
| | 2-opt | 0.014% | 105.44 | 37/50 | 6.321 |

Our implementation of 2-opt randomly selects two cities, and reverses the sub-tour between the two cities instead of swap the two cities. For example, in a tour (3, 8, 6, 4, 1, 10, 2, 7, 9, 5), if cities 6 and 7 are selected, the resulting new tour after 2-opt mutation will become (3, 8, **7**, **2**, **10**, **1**, **4**, **6**, 9, 5) instead of (3, 8, **7**, 4, 1, 10, 2, **6**, 9, 5) as in the other implementation. Cities with different ordering in the tour are in bold and underlined. The space complexities of our 2-opt implementation is small, because the operations are performed inline, only new tours with better fitness scores will be retained and replace the original tour.

The proposed method has been applied to four small TSP problems with numbers of cities ranging from 101 to 442. The GA is implemented in C++, running on Pentium 4 3.0 GHz SMP machine. The SMP feature of the machine is not used. The population size is set to 100, and the limit on iteration is set to 20. For each TSP problem, 50 trials are conducted. Table 1 lists the error rates, number of generations, number of optimums, and CPU times for these problems. From Table 1 we can see that the incorporation of restricted 2-opt indeed improve the performance of *EAX-1AB*, with decreased number of generations, decreased CPU time, and decreased error rates.

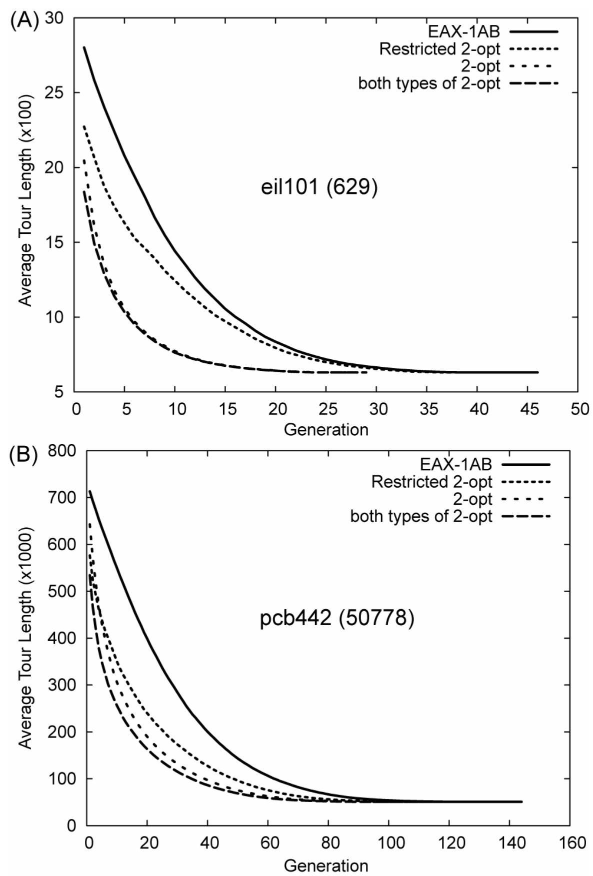


**Figure 1. The plot between generations and average tour length for (A) eil101 and (B) pcb442. The effect of restricted 2-opt is more prominent in larger problem, and the two types of 2-opt have additive improvements.**

It should be interesting to compare conventional 2-opt and our proposed restricted 2-opt. Figure 1 shows the plot between average tour length and generation for two problems eil101 and pcb442. The two types of 2-opt can both enhance the performance of *EAX-1AB* to different degrees; both converged more rapidly than original *EAX-1AB*. The use of restricted 2-opt and conventional 2-opt concurrently also have additive effects on performance gain, though this increase in convergence speed is not as significant as when the 2-opt operators are applied separately. This may be due to the overhead and overlap of the two types of 2-opt operators. In Figure 1 (A), the contribution of conventional 2-opt is dominating, the restricted 2-opt does not provide much additional enhancements. However, for pcb442 in Figure 1 (B), restricted 2-opt provides more improvements comparing to eil101. From these preliminary results, one may speculate that restricted 2-opt is more effective in larger problem. And the contributions of the two types of 2-opt operations may overlap to certain degree.

If the problem size is small as in eil101, it is possible that the conventional 2-opt operating on the entire tour can cover 2-opt searches operating on sub-tours. That is, the two types of 2-opt operations may work on the same region and overlapped with each other. Therefore the optimized tour generated by one type of two opt may be disrupted by the other type of 2-opt. For larger problems, restricted 2-opt can optimize the sub-tours more effectively, whereas conventional 2-opt has higher probability to bypass or disrupt local optimum tours. The contribution of restricted 2-opt becomes more obvious as the problem size increases. One may also speculate that the two types of 2-opt operation work on different stages of evolution.

Overall, the restricted 2-opt can contribute to additional performance gain of EAX even with ICG and conventional 2-opt applied. These contributions are more obvious for larger problems as shown in Figure 1. The curves for conventional 2-opt and restricted 2-opt are intersecting at one point in Figure 1 (B); the two types of 2-opt have different characteristics. Therefore, they may compensate each other at different stage of the evolution; however, their contributions also largely overlapped. This will be further examined in Section 4.

# 4. EXPERIMENTS

## 4.1 The experimental setup

We applied restricted 2-opt to several benchmark problems and compared the performances of *EAX-1AB*, *EAX-1AB* with restricted 2-opt, *EAX-1AB* with conventional 2-opt, and *EAX-1AB* with both types of 2-opt. The population sizes are set to 300, the ICG iteration is still 20. When applied, the 2-opt operations will be repeated 100 times on the sub-tours and the complete tour, respectively. The computer system used is identical to the one described in Section 3. For each problem, 50 trials are performed, and the results are averaged over 50 runs. Each run will terminate if the best tour length does not change for over 20 generations.

## 4.2 Results and discussion

The results of the experiments are listed in Table 2. In Table 2, 5 problems with number of cities ranging from 575 to 1173 are listed. The optimum tour lengths are also provided in parenthesis under the problem name. The error rate (Err.) is the difference between averaged tour length and optimum tour length divided by optimum tour length. Number of generations (Gen.) is the averaged number of generations over 50 trials. Number of optimums (Opt.) is the number of trials reaching optimum solution. For each problem, "EAX" is *EAX-1AB* without any 2-opt operations, "R. 2-opt" is *EAX-1AB* with restricted 2-opt operation applied to sub-tours, "2-opt" is *EAX-1AB* with conventional 2-opt operation applied to the complete tour, and "both" is *EAX-1AB* with both types of 2-opt operations applied.

For all problems except rat783, the error rate decreased with restricted 2-opt. The two types of 2-opt operations in rat783 do not performed well when applied individually. However, with both 2-opt operations applied simultaneously, the error rate decreased from 0.008% to 0.005%. The numbers of generations decreased in all problems, and the number of optimum solutions increased except for rat575. Again, with both 2-opt applied, number of optimums reached increased for rat575. With all types of 2-opt (restricted 2-opt, conventional 2-opt, and both), the CPU time used are reduced in all problems. The additive effects of the two types of 2-opt are also observed in these problems. Combining both types of 2-opt can enhance the performance of *EAX-1AB* in all aspects. Most improvements to EAX have to make trade-offs between better solution quality (error rate, number of generations and probability of reaching optimum) and computation time. However, with restricted 2-opt, both the solution quality and computing efficiency can be improved.

**Table 2. Comparisons among different 2-opt operators. "R. 2-opt" refers to "restricted 2-opt".**

| | | Err. | Gen. | Opt. | Time (s) |
|---|---|---|---|---|---|
| rat575 | EAX | 0.031% | 175.66 | 3/50 | 40.873 |
| (6773) | R. 2-opt | 0.032% | 158.04 | 3/50 | 40.135 |
| | 2-opt | 0.030% | 147.72 | 0/50 | 37.117 |
| | both | 0.029% | 141.12 | 4/50 | 39.208 |
| u724 | EAX | 0.003% | 200.46 | 44/50 | 61.898 |
| (41910) | R. 2-opt | 0.000% | 176.48 | 50/50 | 55.157 |
| | 2-opt | 0.000% | 168.30 | 46/50 | 52.944 |
| | both | 0.000% | 158.24 | 50/50 | 53.027 |
| rat783 | EAX | 0.008% | 212.22 | 31/50 | 73.898 |
| (8806) | R. 2-opt | 0.011% | 188.36 | 32/50 | 68.477 |
| | 2-opt | 0.009% | 179.44 | 35/50 | 64.034 |
| | both | 0.005% | 169.26 | 40/50 | 64.507 |
| vm1086 | EAX | 0.084% | 291.86 | 0/50 | 155.732 |
| (239297) | R. 2-opt | 0.072% | 260.66 | 0/50 | 138.381 |
| | 2-opt | 0.071% | 255.04 | 3/50 | 147.220 |
| | both | 0.065% | 239.08 | 3/50 | 130.779 |
| pcb1173 | EAX | 0.025% | 301.22 | 2/50 | 168.325 |
| (56982) | R. 2-opt | 0.019% | 271.80 | 1/50 | 150.422 |
| | 2-opt | 0.019% | 267.28 | 1/50 | 148.401 |
| | both | 0.015% | 253.28 | 3/50 | 145.030 |

The plot of tour length v.s. number of generations for vm1086 is illustrated in Figure 2. The convergence of the four operators (EAX, EAX with restricted 2-opt, EAX with conventional 2-opt, and EAX with both types of 2-opt) are shown in different line styles. The plot in the first 50 generations has been scaled and placed in the bottom of Figure 2. It is obvious from the plot that the two 2-opt operators have different characteristics and different convergence speed in different stage of the evolution. A vertical dashed line is plotted in the bottom plot. In the left of the dashed line, restricted 2-opt converge more rapidly than typical 2-opt. After the initial burst, restricted 2-opt lag behind conventional 2-opt till the end. With both types of 2-opt operators applied, the evolution converged more rapidly than when both 2-opt operators applied individually.

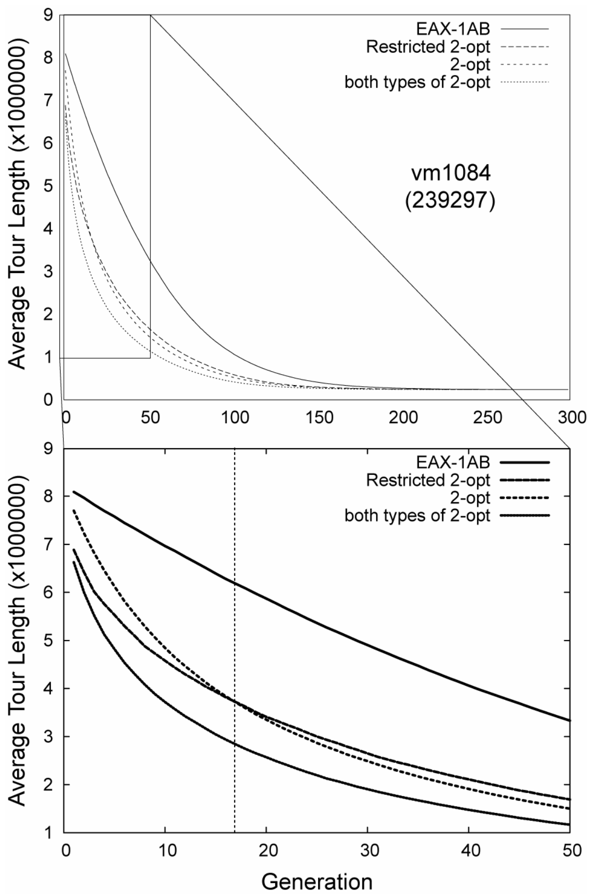


**Figure 2. The tour length v.s. generation plot for vm1084. The first 50 generations are highlighted in the bottom plot. The distinctive characteristics of the two 2-opt operators can be seen in this figure.**

The differences in the convergence behaviors of conventional 2-opt and restricted 2-opt imply that more efficient operators can be constructed with the two operators combined. Taking into consideration of the decay and progression of the two operators, better GAs can be implemented effortlessly. The 2-opt operator is easy to implement and the overhead is small. Implementation of 2-opt is straight forward. From our experiments we can found that the locality of 2-opt can be refined to yield better performance. We usually refer 2-opt as a "local search" strategy in the context of solution space; 2-opt search solutions share large similarity with current solution. However, in the context of a single tour, we can refer the 2-opt operating on the entire tour length as a "global 2-opt"; and the restricted 2-opt operating on sub-tours as "local 2-opt". Combination of both 2-opt operators can effectively optimize the overall and subtle parts of the tour. With unlimited iterations, the "global" or conventional 2-opt can also achieve the same level of optimization as that with both operators applied together. However, the overhead will be too large.

The incorporation of 2-opt into the EAX algorithm can thus enhance the performance and solution quality even with conventional 2-opt mutation operator applied. At early stage of evolution, the restricted 2-opt incorporated into EAX can drive local sub-tours to local optimum more rapidly. One may replace the randomness of 2-opt operators with some heuristics. The size of the problem can be taken into consideration, and concurrent 2-opt operations operating on various regions or lengths of the tours may achieve interesting effects, much like our current implementation with both types of 2-opt operators. However, this may add overhead and complexity to the GA, and should be carefully evaluated and verified.

One interesting question regards whether the EAX algorithm (with 2-opt mutation operator incorporated) can be further improved? We do not have definite answer to this question yet. However, from the analysis of EAX algorithm we can point out some possible steps in EAX that may be targets for improvements. For example, the formation of *AB_cycles* is random in current implementation of EAX [6, 9]; additional heuristics for better *AB_cycle* formation may provide fruitful results. Another possible step is the select of *AB_cycles* for construction of E-set. Nagata *et al.* [6], Nagata [8], and others [12] have proposed several selection scheme, but ICG [9] is much simpler and works well so far. Though strictly speaking ICG is not a new selection scheme. A simple and effective selection scheme is critical to EAX algorithm. Finally, the modification of intermediates into valid tours can also be refined. The selection of cities for joining two sub-tours, and selection of sub-tours to be joined can be considered for possible further enhancements. However, these are beyond the scope of this paper.

# 5. CONCLUSION

Our results suggest that the powerful EAX algorithm can be further refined and improved with efficient and easy to implement 2-opt operator.

We have proposed a simple modification to the EAX algorithm. We have combined all known improvements with EAX, including HpS parent selection, iterative child generation, and external 2-opt mutation operator. We also incorporated 2-opt local search into EAX. The 2-opt local search applies to sub-tours generated in the intermediates of EAX. The combination of 2-opt and EAX have been proposed before by Watson *et al.* [9], but the 2-opt mutation operator works on children generated by EAX, not within the EAX algorithm. Also, conventional 2-opt were applied to the entire tour ("global" to the entire tour). Our restricted 2-opt is restricted to sub-tours of intermediates ("local" in terms of the whole tour). The smaller scope of "local" or restricted 2-opt may effectively speed up the optimization of local sub-tours. Also the mutation operations with EAX (both "global" and "local") can increase the population diversity, avoid common diversity loss in EAX algorithm. Because the characteristics of conventional 2-opt and restricted 2-opt are different, the two types of 2-opt operators can be combined. Restricted 2-opt works better in early generations; while conventional works better in later generations. With both 2-opt applied, solution quality and computation time are both improved. We also analyzed the EAX algorithm and point out several possible means to further improve it.

# 6. REFERENCES


[1] T. Yamada and N. Ryohei, Scheduling by Genetic Local Search with Multi-Step Crossover, In *Parallel Problem Solving from Nature - PPSN IV*, *LNCS 1141*, W. Ebeling*, et al.*, Eds., Springer-Verlag, 1996, 960-969.

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

[3] H. Bohr and S. Brunak, A Traveling Salesman Approach to Protein Conformation, *Complex Systems*, *3*, 9-28, 1989.

[4] D. Whitley, T. Starkweather, and D. Fuquay, Scheduling Problems and Traveling Salesman: The Genetic Edge Recombination Operator, *Proc. of the 3rd Int. Conference on Genetic Algorithms*, 133-140, 1989.

[5] K. Maekawa, N. Mori, H. Kita, and H. Nishikawa, A Genetic Solution for the Traveling Salesman Problem by Means of a Thermodynamical Selection Rule, *Proc. 1996 IEEE Int. Conference on Evolutionary Computation*, 529-534, 1996.

[6] Y. Nagata and S. Kobayashi, Edge Assembly Crossover: A High-power Genetic Algorithm for the Traveling Salesman Problem., *Proc. of the 7th Int. Conf. on Genetic Algorithms*, 450-457, 1997.

[7] H.-K. Tsai, J.-M. Yang, and C.-Y. Kao, Solving Traveling Salesman Problems by Combining Global and Local Search Mechanisms., *Proc. of the 2002 Congress on Evolutionary Computation*, 12920-12925, 2002.

[8] Y. Nagata, The EAX Algorithm Considering Diversity Loss, In *Parallel Problem Solving from Nature - PPSN VIII*, *LNCS 3242*, X. Yao*, et al.*, Eds., Springer-Verlag, 2004, 332-341.

[9] J. Watson, C. Ross, V. Eisele, J. Denton, J. Bins, C. Guerra, D. Whitley, and A. Howe, The Traveling Salesrep Problem, Edge Assembly Crossover, and 2-opt, In *Parallel Problem Solving from Nature - PPSN V*, *LNCS 1498*, A. E. Eiben*, et al.*, Eds., Springer-Verlag, 1998, 823-832.

[10] Y. Nagata, *Genetic Algorithm for Traveling Salesman Problem using Edge Assembly Crossover: its Proposal and Analysis.*, Ph.D. Thesis, Tokyo Institute of Technology, 2000.

[11] H.-K. Tsai, J.-M. Yang, Y.-F. Tsai, and C.-Y. Kao, An Evolutionary Algorithms for Large Traveling Salesman Problems, *IEEE Trans. Systems, Man, and Cybernetics - Part B: Cybernetics*, *34*, 1718-1729, 2004.

[12] K. Ikeda and S. Kobayashi, Deterministic Multi-Step Crossover Fusion: A Handy Crossover Composition for GAs, In *Parallel Problem Solving from Nature - PPSN VII*, *LNCS 2439*, J. J. Merelo*, et al.*, Eds., Springer-Verlag, 2002, 162-171.