Genetic Algorithm For The Travelling Salesman Problem using Enhanced Sequential Constructive Crossover Operator

Prof. Dr.Hachemi Bennaceur *hamnasser@imamu.edu.sa*
Faculty of Computer and Information Sciences
Department of Computer Science
Al Imam Mohammad Ibn Saud Islamic University (IMSIU),
Riyadh, Saudi Arabia

Entesar Alanzi *eaalanzi@imamu.edu.sa*
Teacher Assistant / Department of Computer Science
Al Imam Mohammad Ibn Saud Islamic University (IMSIU),
Riyadh, Saudi Arabia

Abstract

Traveling Salesman Problem (TSP) is one of the most important combinatorial optimization problems. There are many researches to improve the genetic algorithm for solving TSP. The Sequential Constructive crossover (SCX) is one of the most efficient crossover operators for solving optimization problems. In this paper, we propose a new crossover operator, named Enhanced Sequential Constructive crossover operator (ESCX), which modifies and improves the criteria of SCX operator in construction of offspring. ESCX considers, in addition to the real cost of the traversed cities, an estimation cost of the remaining tour, and it selects the next node to build the offspring based on that evaluation. The experimental results comparing the proposed crossover operator to the SCX operator on some benchmark TSPLIB instances show the effectiveness of our proposed operator.

Keywords: Traveling Salesman Problem, Optimization Problem, Genetic Algorithm, Sequential Constructive Crossover.

1. INTRODUCTION

Traveling Salesman Problem (TSP) is one of the most important NP-Hard combinatorial problems in computer science and operations research, any problem that belongs to the NP-class can be formulated to TSP [3]. A large number of real-world problems can be modeled by TSP problem such as scheduling [5], Vehicle routing [13], industrial robotics [6], VLSI layout design, computing wiring [2], DNA sequencing [10][9]. TSP consists of a salesman and set of cities, the salesman has to find the shortest tour to visit all the cities exactly once and returns back to the starting city. TSP can be represented by a complete undirected weighted graph G (V, E) where the cities are the graph's vertices and the distances are presented by weighted edges. The main goal of TSP is to find the minimized Hamilton cycle which starts with a vertex, visits all vertices exactly once then ends at the same starting vertex [8]. The optimal solution is a permutation π of the vertices index 1, 2, 3….n such that the cost function $f(\pi)$ is minimized, where $f(\pi)$ is given by

$$f(\pi) = \sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)} + d_{\pi(n)\pi(1)}$$

Where $d_{\pi(i)\pi(i+1)}$ is the distance between city i and city i+1, and $d_{\pi(n)\pi(1)}$ is the distance between city n and the starting city [12]. For a problem of n cities, the total number of possible paths covering all cities is n!, so, this problem may be impractical even for small size problems.

The TSPs can be classified into - Symmetric traveling salesman problem (sTSP) and Asymmetric traveling salesman problem (aTSP) [2]. In the Symmetric TSP, the distance between city A and city B is the same as the distance between B and A. For n cities symmetric TSP, there are (n-1)! /2 possible solutions. The Asymmetric TSP otherwise, the distance between city A and city B is not equal to the distance between city B and A [12], and there are (n-1)! possible solutions [20].

Many algorithms have been developed for solving TSP. Genetic Algorithms (GAs) are found to be one of the best heuristic algorithms for solving TSP problems and yield approximate solutions within a reasonable time. There are many improvements that have been suggested in GA to enhance the performance in solving TSP [15], one of them is creating new crossover operators such as: [4],[14],[17],[19]. The Sequential Constructive crossover (SCX) [20] is the most efficient well-known crossover operators for solving optimization problems. The SCX constructs an offspring (individual) using better edges on the basis of their values present in the parents' structure [20].

In this paper, we propose an enhancement of the SCX operator named ESCX. It applies a more elaborated criteria in the building of the offspring. This idea is inspired from the efficient A* technique which during the exploration of the state space selects a node to expand based on an evaluation function combining the real cost of the node (the distance from the initial state to the node) and a heuristic estimating the remaining cost from the node to reach the goal state. ESCX considers for each city, in addition to the real cost of the traversed cities, an estimation cost of the remaining tour. And then it selects the next node to construct the offspring based on that evaluation. Our enhancement is expected to traverse the search space and produce good solutions in term of quality of solutions as well as solution times. We experimented on some benchmark TSPLIB instances and compared the results of our proposed crossover operator with the SCX operator. The Experimental results show the effectiveness of our proposed operator (ESCX) for solving TSP problem

The remaining parts of this paper are organized as follows: Section 2 presents the related works. Section 3 discusses the development of a genetic algorithm based on our proposed operator for the TSP, while sections 4 presents the computational experiments and comparison. Section 5 discusses the experimental results. Finally, the paper concludes with Section 6.

2. RELATED WORKS

Crossover operator is the most important operator in GA, which creates new individual(s) by combining two randomly selected parents. Crossover operator plays a critical role in GA since characteristics are exchanged between individuals of the population [17]. The researchers have studied many crossover operators like creating new crossover operators [4],[14],[17],[19] or modifying existing crossover operators [15],[16].

In [20] Ahmed proposed a new unconventional crossover operator, Sequential Constructive crossover (SCX), which has been one of the best operators for solving TSP problem. The Sequential Constructive Crossover (SCX) constructs an offspring using better edges on the basis of their values present in the parents' structure. The SCX does not depend only on the parents' structure. So, it sometimes introduces new better edges to the offspring. Hence, the chances of producing better offspring are more than those of ERX and GNX operators. A comparative study was presented among SCX, Edge Recombination Crossover (ERX) and Generalized N-point crossover (GNX) for some benchmark TSPLIB instances. The experimental results showed that the SCX is better than the ERX and GNX in term of quality of solution and solution time [20] [3].

Abdel-Moetty and Heakil [17] proposed a new crossover operator, Modified Sequential Constructive Crossover (MSCX), which is an improvement of the SCX [20]. They tested solution quality of the proposed algorithm on small input data (from 5 to 20 cities). The results showed that for a number of cities from 15 to 20, and small population size, the MSCX achieved better

results than SCX, PMX [22], OX [23], and MOX. However, the efficiency of their algorithm was experimented on only small dataset to measure the solution quality.

Thanh et al. [15] proposed two new crossover operators, MSCX_Radius and RX crossover, and a new mechanism of a combination of these operators in GA for solving the TSP. The MSCX_Radius modified the second step of MSCX [17], such that if no 'legitimate node' exists after current node p, find sequentially the r nodes, which are not visited by the parents and select the shortest distance to the node p. The experimental tests showed that the results found by MSCX_Radius were worse than MSCX, but the combination of two propose crossover operators MSCX_Radius and RX and MSCX was effective for TSP and better than the GA using only MSCX based on solution quality.

In [3], the authors studied two unconventional crossovers Sequential Constructive Crossover (SCX) and Inverse Sequence Crossover (ISX), and two mutation operators: Greedy Mutation operator and Normal Mutation operator. The authors observed that the best combination found was SCX with Greedy mutation operator together were able to traverse the search space and produce better results.

Ray et al. [18] proposed a new genetic algorithm called FRAG_GA. A new nearest fragment operator (NF) and modified order crossover (MOC) were proposed for solving symmetric TSP and microarray gene ordering problem. The NF was used for determining an appropriate number of fragments for optimizing the initial population. The MOC was used for determining an appropriate substring length for performing order crossover (OX). The experiment results showed that FRAG_GA obtained better results compared to the OXSIM [16] implementing standard GA with order crossover and simple inversion mutation, and SWAPGATSP [19].

In [14], the authors proposed a new crossover operator, Swapped Inverted Crossover (SIC) and a new operation called Rearrangement. The SIC constructed 12 individuals from two selected parents, then two best individuals were selected. Rearrangement operation was applied to all populations to find the greatest cost c_{ij} between two adjacent cities i and j among all adjacent cities on the tour, then city i was swapped with three different cities, the best position will be accepted. The experimental results showed that the proposed GA obtained better results compared to FRAG_GA [18], SWAPGATSP [19] and OXSIM [16].

Satyananda [4] proposed two new crossover operators, named Crossover operator using Nearest Neighbor algorithm (CNN) and Crossover operator using Sequential Insertion algorithm (CSI). Both operators did not depend only on the parents' structure; they could produce better offspring than their parents. A comparative study among them and PMX operator was performed for TSPLIB instances: gr21, berlin52, and eil2. The experiment results showed that the CSI obtained better results than CNN and PMX operators.

Another crossover operator called Edge Recombination Crossover (ERX) proposed by [7] which constructs an offspring from the concept of edge map, which lists all the neighborhood of each node in the parents [11]. In [11] eight crossover operators : Ordered Crossover (OX), the Partially Mapped Crossover (PMX), Edge Recombination Crossover (ERX), the Cycle Crossover (CX), Alternating Edges Crossover (AEX) and Heuristic Crossovers (HGreX, HRndX and HProX) have been compared on the vehicle routing problem (VRP) and the traveling salesman problem (TSP). The experiment results for the VRP showed that the best performances were provided by HGreX or AEX. However, for the TSP, the best performances were obtained by OX and ERX.

Our objective in this paper is to show the effectiveness of our improved ESCX crossover by comparing it with the best-known crossover operator SCX. This enhancement of the exploration of the state space of the problem is expected to improve the performance of the algorithm in terms of solution quality as well as solution time.

3. GENETIC ALGORITHMS

Genetic algorithms (GAs) are based on natural evolution and genetics to solve complex optimization problems [3]. GAs can be used to find approximate solutions in a reasonable time for TSP [15]. GA uses a set of solutions (individuals) called population to start the process. The solutions are encoded by chromosomes and each of them contains a number of genes. The goodness of each chromosome is measured by the fitness function. The best solutions from the population are selected to produce the next population. The production of the new solutions occurs by using crossover and mutation operators. The main role of crossover operator is to combine two parents (individuals) to produce a new offspring (child), and the main role of mutation operator is to recover from stuck into local optimal by making a certain change in the individuals. The genetic search usually terminates when a maximum number of generations have been produced [21].

3.1 Sequential Constructive Crossover Operator (SCX)

The Sequential Constructive Crossover (SCX) constructs an offspring using better edges on the basis of their values present in the parents' structure [20]. The SCX does not depend only on the parents' structure. So, it has better chance to introduce better offspring than the ERX and GNX crossover operators [20] [3]. The algorithm for the SCX is as follows:

Step1: Start from node1 of the parent1, 'node p'.

Step2: Sequentially search both of the parent chromosomes and consider the first 'legitimate node' (the node that is not yet visited) appeared after 'node p' in each parent. If no 'legitimate node' after node p is present in any of the parent, search sequentially the nodes {2, 3, 4, …, n} and consider the first 'legitimate node', and go to Step 3.

Step3: Suppose the 'Node α' and the 'Node β' are found in 1st and 2nd parent respectively, then for selecting the next node go to Step 4.

Step4: If Cpα <Cpβ, then select 'Node α', otherwise, 'Node β' and concatenate it to the partially constructed offspring chromosome. If the offspring is a complete chromosome, then stop, otherwise, rename the present node as 'node p' and go to Step 2 [20].

3.2 Enhanced Sequential Constructive Crossover (ESCX)

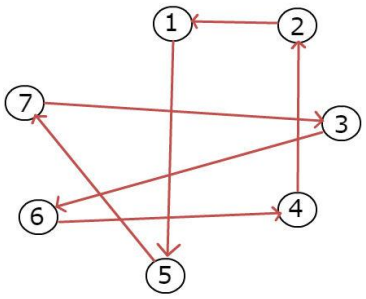
While the SCX is one of the best algorithms available for solving the TSP problem, as mentioned in related work section many attempts to improve it have been proposed. In this paper, we propose an enhancement of the SCX operator named ESCX. This idea is inspired from A* technique which selects nodes to expand based on an evaluation function lying on the real cost of the node and a heuristic estimating the remaining cost from the node to the goal to reach. ESCX applies a more elaborated criteria in building of offspring. It considers, in addition to the real cost of the traversed cities, an estimation cost of the remaining tour, and it selects the next node to construct the offspring based on that evaluation The remaining cost of the tour can be estimated using a greedy algorithm which considers the arc having minimum cost at each step till completing the tour. This process may be time-consuming as it is repeated for each node of the offspring, in our implementation the remaining cost to complete the tour is estimated to the minimum cost to leave the nodes. More precisely, assume n1 and n2 are the next not visited nodes of the parents and ESCX will select one of them to be added to the offspring. Let n3 be the last node of the current offspring, then the criteria used by ESCX to select next node to be added to the offspring is: min ($c_{n3n1}+c_{n1t}$, $c_{n3n2}+c_{n2k}$) where: c_{n1t} =min { c_{n1nj} : n_j not included in the offspring} and c_{n2k} =min { c_{n2nj} : n_j not included in the offspring}

Let us illustrate the ESCX algorithm through the example given as cost matrix in Table 1. Let a pair of selected parents be P1 :(1, 5, 7, 3, 6, 4, 2, 1) and P2 :(1, 6, 2, 4, 3, 5, 7, 1) with cost values 312 and 331 respectively. Let M[n] denotes the minimum remaining cost for node n.

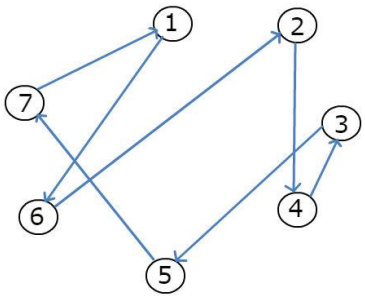
Node	1	2	3	4	5	6	7
1	999	75	99	9	35	63	8
2	51	999	86	46	88	29	20
3	100	5	999	16	28	35	28
4	20	45	11	999	59	53	49
5	86	63	33	65	999	76	72
6	36	53	89	31	21	999	52
7	58	31	43	67	52	60	999

TABLE 1: The Cost Matrix.

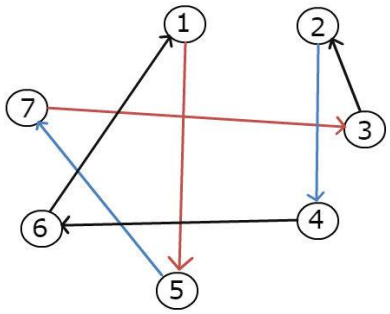
Start from 'node 1' as the 1st gene. The 'legitimate' nodes after 'node 1' in P1 and P2 are 'node 5' and 'node 6' respectively with c_{15}=35 and c_{16}=63 then we see M [5] = 33 and M [6] = 21. Since c_{15}+M [5] < c_{16}+ M [6], we accept 'node 5'. The 'legitimate' node after 'node 5' in both P1 and P2 is 'node 7'. So, we accept the 'node 7', and the partially constructed chromosome will be (1, 5, 7). The 'legitimate' node after 'node 7' in P1 is 'node 3', but none in P2. So, for P2, we consider the first 'legitimate' node in the set {2, 3, 4, 5, 6, 7}, that is, 'node 2'. Since c_{72} = 31+ (M [2]=29) **>** c_{73} = 43 + (M [3]=5), we accept 'node 3'. Thus, the partially constructed chromosome will be (1, 5, 7, 3). Again, the 'legitimate' node after 'node 3' in P1 is 'node 6', but none in P2. So, for P2, we consider the first 'legitimate' node in the set {2, 3, 4, 5, 6, 7}, that is, 'node 2'. Since c_{36} = 35+ (M [6]=31) **>** c_{32} = 5 +(M [2]=29), we accept the 'node 2'. The partially constructed chromosome will be (1, 5, 7, 3, 2). The 'legitimate' nodes after 'node 2' in P1 is none but in P2 is 'node 4'. So for P1, we consider the first 'legitimate' node in the set {2, 3, 4, 5, 6, 7}, that is 'node 4'. We accept 'node 4', which will lead to the partial chromosome (1, 5, 7, 3, 2, 4). The 'legitimate' node after P1 and P2 is none. So, we consider the first 'legitimate' node in the set {2, 3, 4, 5, 6, 7}, that is, 'node 6'. We accept the 'node 6'. Thus the complete offspring chromosome will be (1, 5, 7, 3, 2, 4, 6) with value **290** which is less than the value of both parent chromosomes. Notice that the applying of SCX crossover operator to the two parents P1 and P2 produces the offspring (1, 5, 7, 3, 6, 4, 2) with value **312**. The crossover is shown in Figure 1. Parent1 and Parent2 are shown as (a) and (b), while their possible offspring is shown as (c). In Figure 1(c), red edges are the edges which are selected from Parent1, and blue edges are selected from the second parent. Black edges are the newly constructed edges.



(a) Parent1 : (1, 5, 7, 3, 6, 4, 2)



(b) Parent2 : (1, 6, 2, 4, 3, 5, 7)



(c) Offspring: (1, 5, 7, 3, 2, 4, 6)

FIGURE 1: Example of Enhanced Sequential Constructive crossover operator.

3.3 Structure of Genetic Algorithm

Our proposed GA implementing ESCX crossover operator starts by generating randomly an initial population. It uses the tournament selection technique to select parents for the crossover operation. And it uses the traditional basic mutation operation which consists of swapping randomly two selected nodes of the individual. It can be summarized as follows:

```
Genetic Algorithm ()
        { Initialize random population;
        Evaluate the population;
        Generation = 0;
        While termination condition is not satisfied
                { Generation = Generation +1;
                /*  Tournament selection */
                Parents' selection process;
                ESCX Crossover operator with probability of crossover (Pc);
                /* Mutation: selects two nodes randomly and swaps them */
                Mutation procedure with probability of mutation (Pm);
                Evaluate the population;}
        Return the fitness of the best individual found }
```

4. COMPUTATIONAL EXPERIMENTS

The genetic algorithms using Sequential Constructive crossover, and Enhanced Sequential Constructive Crossover operator for TSP, have been coded using the Java programming language in the NetBeans environment. The initial population was randomly generated. The following common parameters were selected for the algorithms: population size: 200 individuals; maximum number of generations is 10,000, crossover probability is 1.0, mutation probability is 0.01. The experiments were performed 10 runs for each instance. The solution quality was measured by the percentage of excess above the optimal solution found in TSBLIB website [24], as the given equation:

$$\text{Excess}(\%) = \frac{\text{Best Solution} - \text{OptimumTSBLIB}}{\text{OptimumTSBLIB}} \times 100$$

We report the percentage of the excess of best, average (Avg.) solution values and the average time of convergence (in seconds).

Instance	n	Opt. Sol.	ESCX			SCX		
			Best (%)	Avg. (%)	Avg. Time	Best (%)	Avg. (%)	Avg. Time
br17	17	39	0	0	4	0	0	4
ftv33	34	1286	2.799378	8.38258165	8.2	8.709176	14.48678	7
ftv35	36	1474	6.24152	7.21166893	8.7	2.035278	8.080054	7.9
ftv38	39	1530	1.895425	6.45098039	9.1	3.660131	6.084967	8.2
p43	43	5620	0.017794	0.16370107	10	0.071174	0.272242	10
ftv44	45	1613	6.075635	7.32176069	11.5	7.439554	11.54991	10.6
ftv47	48	1776	4.222973	7.50563063	13	2.027027	10.63063	11.4
ry48p	48	14422	2.399112	4.56386077	12.9	2.198031	7.063514	11.4
ftv55	56	1608	0	4.62064677	15.6	4.975124	12.88557	14.3
ftv64	65	1839	2.175095	7.50951604	19.4	1.848831	9.129962	17.3
ft70	70	38673	1.626458	3.27024022	22.5	3.583896	4.580715	19.8
kro124p	100	36230	13.59371	16.0441623	43.7	11.32763	13.16092	41.1
ftv170	171	2755	15.64428	17.876588	108.3	13.902	23.24864	95

TABLE 2: Summary of the results by the crossover operators for Asymmetric TSPLIB instances.

Table 2 shows the results obtained for thirteen asymmetric TSPLIB instances of size from 17 to 171. Only one instance, br17 of size 17 could be solved exactly by SCX, whereas two instances, br17 and ftv55 could be solved exactly at least once in ten runs by ESCX. On the basis of the quality of average solution value, for all the instances except one: kro124p, ESCX is found to be better than SCX; but for the kro124p instance, SCX is found slightly better. On the basis of the quality of best solution value, for six instances ftv33, ftv38, p43, ftv44, ftv55 and ftv70, ESCX is found to be better; but for the other six instances ftv35, ftv47, ry48p, ftv64, kro124p and ftv170, SCX is found to be better. In term of computation time, the difference between the two crossover operators is insignificant, the average gap between ESCX and SCX in the convergence time is ~ 1%. Figure 2 compares the average solution values of the SCX and ESCX for asymmetric TSPLIB instances. In conclusion, ESCX dominates SCX in terms of solution quality on asymmetric problems.

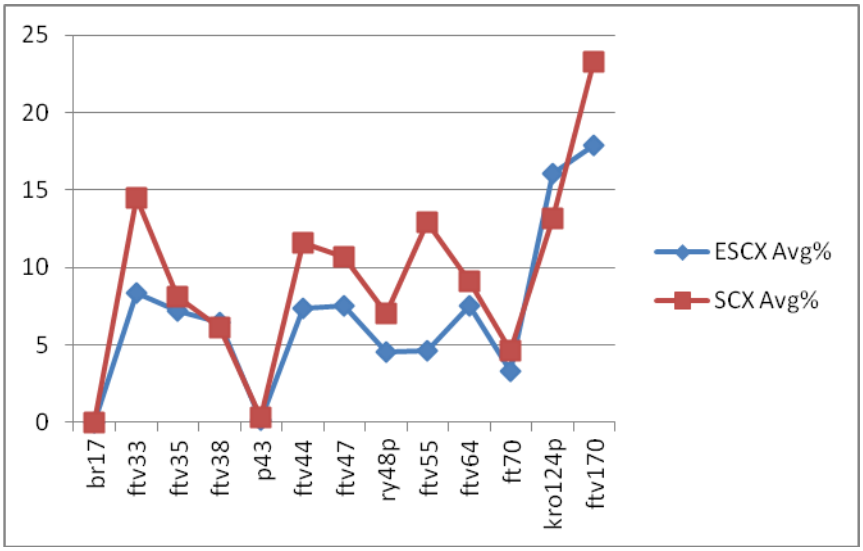


FIGURE 2: SCX and ESCX for Asymmetric TSPLIB instances.

Figure 3, shows the performance of the two crossover operators for the ftv170 instance, considering only 1000 generations in one run. Both of the ESCX and SCX have almost the same range of variations, but the SCX gets stuck in local minimum quickly and faster than ESCX. Hence, the figure shows that ESCX is better than SCX in term of solution quality.

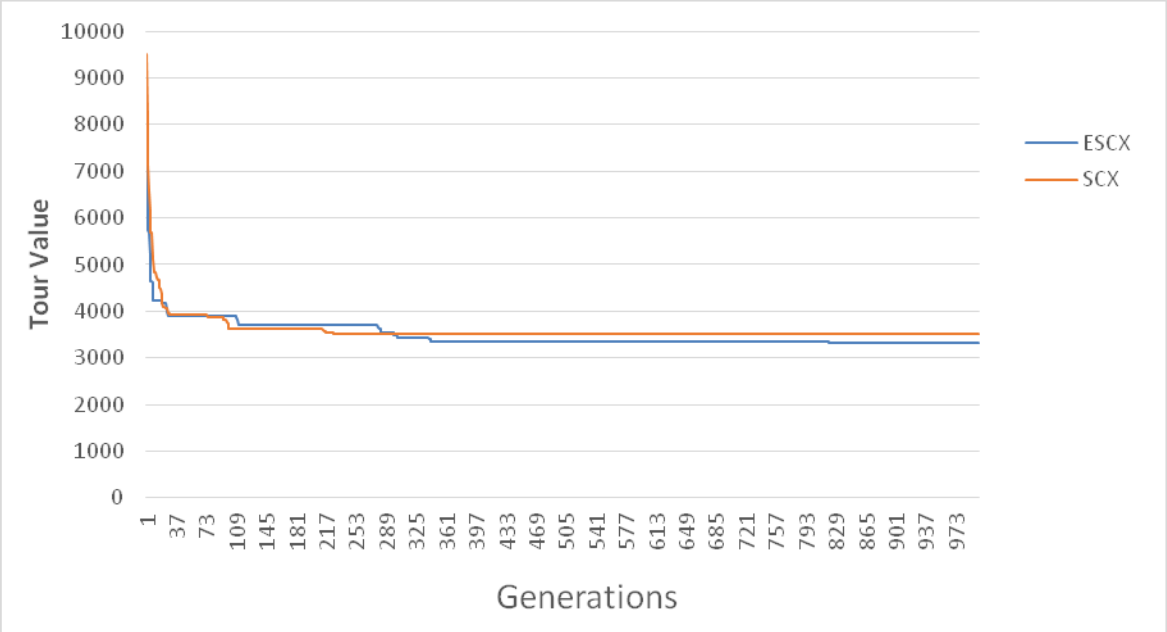


FIGURE 3: Performance of the two crossover operators on the instance ftv170.

Instance	n	Opt. Sol.	ESCX			SCX		
			Best (%)	Avg. (%)	Avg. Time	Best (%)	Avg. (%)	Avg. Time
bayg29	29	1610	0	2.018634	7	0	2.142857	6.2
berlin52	52	7542	5.436224	9.22567	14.9	5.595333	9.263277	13.4
eil51	51	426	0.234742	2.58216	14.4	0.234742	4.097311	13.4
eil76	76	538	0.185874	3.866171	25.7	0.929368	4.241298	23.4
pr76	76	108159	0.550116	3.743748	25.8	0.908847	6.00404	25.5
kroA100	100	21282	15.87727	19.98449	40.1	7.607368	11.2527	38.1
kroC100	100	20749	12.87291	15.91306	38.7	9.079956	13.44402	37.1
eil101	101	629	5.72337	7.917329	40.7	3.81558	6.3593	40.5
lin105	105	14379	10.92566	13.36463	48.2	5.132485	10.30461	39.8
d198	198	15780	10.32319	13.06717	210.7	5.519645	9.808619	339.1

TABLE 3: Summary of the results by the crossover operators for symmetric TSPLIB instances.

Table 3, reports the results for ten symmetric TSPLIB instances of size from 29 to 198. Only one instances bayg29 of size 29, could be solved exactly by SCX and ESCX. On the basis of the quality of best solution value and average solution value, for the instances from bayg29 to pr76, ESCX is found to be better; but for the instances from kroA100 to d198, SCX is found to be better. For these symmetric instances, as the size of the problem decreases from 100 cities, ESCX is found to be better than SCX.

5. DISCUSSION

While the SCX has been one of the best-known algorithms for solving the TSP problems, our comparison was focused on the SCX approach to show the impact of our proposed improvement (ESCX) on some benchmark TSPLIB instances. The experimental results clearly demonstrate the effectiveness of the ESCX for asymmetric TSP instances in term of solution quality and computation time, that's because the ESCX applies more elaborated criteria in the building of the offspring. It constructs offspring based on an estimation cost of the remaining tour - in addition to the real cost of the traversed cities. Then, it selects the next node based on that evaluation. This evaluation would be significant and leads to constructing offspring with better quality. Further comparisons will be conducted in the future to evaluate the enhanced crossover operator against recent genetic algorithms for solving TSPs.

6. CONCLUSION

In this paper, we have proposed a new enhancement to the SCX approach, called ESCX. The main idea of ESCX is that it introduces an additional step from the future node by estimating the minimum remaining cost for each node of the parents' nodes. So it will select the next node to the offspring based on the distance from the current node to the next node and the estimation of the remaining cost. We presented a comparative study among ESCX and SCX for some benchmark TSPLIB instances to compare the efficiency of them.

The experimental results show that our proposed crossover operator (ESCX) obtained effective results for GA for TSP in terms of solution quality and computation time. Our future investigation aims to combine GA with adequate local search technique in order to improve the solution quality of the problem.

7. REFERENCES

[1] A. Saiyed, "The Traveling Salesman problem", Indiana State University, vol. 2, pp. 1-15, 2012.

[2] R. Matai, S. Singh and M. L. Mittal. "Traveling Salesman Problem: An Overview of Applications, Formulations, and Solution Approaches," in TRAVELING SALESMAN PROBLEM, THEORY AND APPLICATIONS, InTech, Dec. 2010.

[3] D. Nandi, S. Samanta and A. De. "An Approach to Analyze the Performance of Inversion Sequence Crossover and Sequential Constructive Crossover Along With Greedy Mutation Operator for Solving TSP Using Genetic Algorithm," ICCCM 2014, International Conference on Computing, Communication & Manufacturing, 2014.

[4] D. Satyananda. "Modification of Crossover Operator on GA Application for TSP," Proceeding of International Conference On Research, Implementation And Education Of Mathematics And Sciences, 2015.

[5] H. Gress and E. Selene. "The job shop scheduling problem solved with the travel salesman problem and genetic algorithms," https://repository.uaeh.edu.mx/bitstream/handle/123456789/15273, Apr. 14, 2017.

[6] F. Imeson and S. Smith."A Language For Robot Path Planning in Discrete Environments: The TSP with Boolean Satisfiability Constraints," Proceedings of the IEEE Conf. on Robotics and Automation, May 2014.

[7] G. Gopal, R. Kumar, I. Jawa and N. Kumar. "Enhanced Order Crossover for Permutation Problems". International Journal of Innovative Research in Science, Engineering, and Technology, vol. 4, no. 2, 2015.

[8] H. Raşit and N. Erdoğan, "Parallel Genetic Algorithm to Solve Traveling Salesman Problem on MapReduce Framework using Hadoop Cluster". The International Journal of Soft Computing and Software Engineering [JSCSE], vol. 3, no. 3, 2013.

[9] J. Singh and A. Solanki. "An Improved Genetic Algorithm on MapReduce Framework Using Hadoop Cluster for DNA Sequencing". The International Journal of Advanced Research in Computer Science and Software Engineering, vol. 5, 2015.

[10] K. Miclaus, R. Pratt and M. Galati. "The Traveling Salesman Traverses the Genome: Using SAS® Optimization in JMP® Genomics to build Genetic Maps", SAS global forum,2012.

[11] K. Puljic and R. Manger. "Comparison of eight evolutionary crossover operators for the vehicle routing problem". Journal of mathematical communications, 2013.

[12] L. S´anchez, J. Armenta and V. Ram´ırez. "Parallel Genetic Algorithms on a GPU to Solve the Travelling Salesman Problem," Difu100ci@ Revista en Ingeniería y Tecnología, UAZ, vol. 8, 2014.

[13] N. Bansal, A. Blum, S. Chawla and A. Meyerson. "Approximation Algorithms for Deadline-TSP and Vehicle Routing with Time-Windows," Proceedings of the thirty-sixth annual ACM symposium on Theory of Computing (STOC), 2004, pp. 166-174.

[14] O. Sallabi and Y. El-Haddad. "An Improved Genetic Algorithm to Solve the Traveling Salesman Problem". International Journal of Computer, Electrical, Automation, Control and Information Engineering, vol. 3, 2009.

[15] P. Dinh, H. Binh and B. Lam. "New Mechanism of Combination Crossover Operators in Genetic Algorithm for Solving the Traveling Salesman Problem". Springer International Publishing Switzerland, Knowledge and Systems Engineering, vol. 326, 2015.

[16] P. Larranaga, C.M.H. Kuijpers, R.H. Murga, I. Innza and S. Dizdarevic."Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators", Artificial Intelligence Review 13: 129–170, 1999.

[17] S. Abdel-Moetty and A. Heakil. "Enhanced Traveling Salesman Problem Solving using Genetic Algorithm Technique with modified Sequential Constructive Crossover Operator". The International Journal of Computer Science and Network Security (IJCSNS), vol. 12, 2012.

[18] S. Ray, S. Bandyopadhyay and S. Pal. "New genetic operators for solving TSP: Application to microarray gene ordering.". Springer-Verlag Berlin Heidelberg, pp. 617–622, 2005.

[19] S. Ray, S. Bandyopadhyay and S. Pal, "New operators of genetic algorithms for traveling salesman problem",Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.

[20] Z. Ahmed. "Genetic Algorithm for the Traveling Salesman Problem using Sequential Constructive Crossover Operator". International Journal of Biometrics & Bioinformatics (IJBB), vol. 3, 2010.

[21] A. Rao and S. Hegde. "Literature Survey On Travelling Salesman Problem Using Genetic Algorithms". International Journal of Advanced Research in Education Technology (IJARET), vol. 2, 2015.

[22] B.F. Al-Dulaimi and H. A. Ali. "Enhanced Traveling Salesman Problem Solving by Genetic Algorithm Technique (TSPGA)". International Journal of Mathematical, Computational, Physical, Electrical and Computer Engineering, vol. 2, 2008.

[23] I. Gupta and A. Par. (2011, Aug.). “Study of Crossover operators in Genetic Algorithm for Travelling Salesman Problem.” [On-line]. 2(4), pp. 194-198. Available: www.ijarcs.info [May 1, 2017].

[24] G. Reinelt. “TSPLIB.” Internet: https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/ , Jan, 1, 2013 [Jun. 3, 2017].

[25] W. Zeng and R. L. Church. “Finding shortest paths on real road networks: the case for A*”. International Journal of Geographical Information Science, vol. 23, pp. 531-543, Jun. 2009.