# Enhanced Traveling Salesman Problem Solving using Genetic Algorithm Technique with modified Sequential Constructive Crossover Operator

**Dr.Sabry M. Abdel-Moetty**
Chief of Researches and Development Center, EAF, Egypt.
**Asmaa O. Heakil,**
Associate lecture, Faculty of Science, Al-Azhar University Cairo, Egypt.

**Abstract**
This paper proposes Genetic Algorithm (GA) with a new crossover operator, Modified Sequential constructive crossover (MSCX), to solve the Traveling Salesman Problem (TSP). The proposed algorithm constructs an offspring from a pair of parents using better edges on the basis of their values that may be present in the parents' structure maintaining the sequence of nodes in the parent chromosomes. The results of the proposed algorithm (MSCX) are compared with others GA algorithms which use different crossover operations, MSCX achieve a better solution for TSP.

***Keywords:***
*Traveling Salesman Problem (TSP), Sequential constructive crossover (SCX), Sequential constructive crossover (MSCX).*

## 1. Introduction

In 1975, Holland described how to apply the principles of natural evolution to optimization problems and built the first Genetic Algorithms in his book "Adaptation in natural and artificial systems". Holland's theory has been further developed and now Genetic Algorithms (GAs) stand up as a powerful tool for solving search and optimization problems. Genetic algorithms are based on the principle of genetics and evolution. The power of mathematics lies in technology transfer: there exist certain models and methods, which describe many different phenomena and solve wide variety of problems. GAs is an example of mathematical technology transfer: by simulating evolution one can solve optimization problems from a variety of sources. Today, GAs is used to resolve complicated optimization problems, like, timetabling, job-shop scheduling, and games playing [1]. Traveling Salesman Problem (TSP) optimization problem is one in which the aim is to find the best way to order elements according to the minimum total cost, (e.g., visit a set of cities according to the minimum total distance to travel.) There are many possible ways to order these elements, but some combinations will be better than others. Each possible answer, or combination, is considered to be a "solution". The best solution is called the "optimal solution". The answer the problem requires that a solution should be proved to be the optimal solution. Often, this means that a very large number of solutions need to be tested in order to determine which solution is optimal. Given a set of cities along with the cost of travel between each pair of them, TSP, is to find the cheapest way of visiting all the cities and returning to the starting point. The "way of visiting all the cities" is simply the order in which the cities are visited; the ordering is called a tour or circuit through the cities [2].

## 2. Background

### 2.1 Traveling Salesman Problem

The Traveling Salesman Problem (TSP) is one of the important subjects which have been widely addressed extensively by mathematicians and computer scientists. It is a permutation problem with the objective of finding the path of the shortest length (or the minimum cost) on an undirected graph that represents cities or node to be visited. The Traveling Salesman Problem (TSP) is one of the important subjects which have been widely addressed extensively by mathematicians and computer scientists [3]. It is a permutation problem with the objective of finding the path of the shortest length (or the minimum cost) on an undirected graph that represents cities or node to be visited. The traveling salesman starts at one node, visits all other nodes successively only one time each, and finally returns to the starting node. The objective is to find the path, which follows following constrains [4]:

1. Salesman has to visit each city. He should not leave any city unvisited.
2. Each city should be visited only one time.
3. The distance that has been traveled till returns back to the City that started with should be the shortest distance.

### 2.2 Genetic Algorithm

Genetic Algorithms (GA's) are relatively new paradigms in artificial intelligence which are based on the principles

Manuscript received June 5, 2012
Manuscript revised June 20, 2012

of natural selection. The formal theory was initially developed by John Holland and his students in the 1970's [5-6]. The continuing improvement in the price/performance value of GA's has made them attractive for many types of problem solving optimization methods. An initial population called genome (or chromosome) is randomly generated then successive populations, or generations, are derived by applying genetic operators such as selection, crossover and mutation to evolve the solutions in order to find the best one(s). The selection operator chooses two members of the present generation in order to participate in the next operations: crossover and mutation. The crossover operator recombines the alleles of the two parents to obtain an offspring. The mutation operator occurs a short period of time after crossover and as in nature it exchanges alleles randomly. The three most important aspects of using genetic algorithms are: (1) definition of the objective function, (2) definition and implementation of the genetic representation, and (3) definition and implementation of the genetic operators [3]. It is essential to modify the Genetic Algorithm when it is used to solve any specific problem .This modification is basically done in the Encoding Method and GA operator (Selection ,Crossover, Mutation).The way of encoding technique and selection ,crossover, mutation process vary problem to problem, still basic logic behind each operator is same. So to solve domain specific problem specialized GA operators are required [7].

### 2.2.1 Encoding

In conventional approach a chromosome which is devised to represent a solution constitutes of N (counts the no of cities) genes. Each gene holds a number which is a label of a city. So the nth gene holds the label of the city which is visited nth in that particular tour. In other words the chromosome is a direct coding of a permutation of the sequences 1, 2…….N. To start with a population of valid chromosome gene repairing is needed where each invalid chromosome (with repetition of cities) are fed into an intermediate process which transform them into valid once.

### 2.2.2 The Fitness Function

A fitness function evaluation is incorporated to assigns a value to each organism, noted as fi. Organisms are chosen using the fitness value as a guide, where those with higher fitness values are chosen more often. Selecting organisms based on fitness value is a major factor in the strength of GAs as search algorithms. The method employed here was to calculate the total Euclidean distance Di for each organism first, then compute fi by using Eq. (1)

$$f_i = D_{max} - D_i \qquad (1)$$

Where Dmax is the Euclidean distance, should be longest organisms in the chosen population.

### 2.2.3 Selection Operations

The selection operator chooses two members of the present generation to participate in the next operations of crossover and mutation.

### 2.2.4 Crossover

Crossover is nothing but recombination of parents. It is the most important operation of a GA because in this operation, characteristics are exchanged between the individuals of the population. the main constraints on crossover operators for TSP is that there should not be any repetition of cities in a particular chromosome, that's why it is not possible to use 1 point,2 point crossover . Since the crossover operator plays a vital role in GA, so many crossover operators have been proposed for the TSP [7]. The used crossover operator for TSP is order crossover (OX) and modified order crossover (MOX), as described briefly below.

#### 2.2.4.1 Order Crossover

Two random cross points are selected. Alleles from parent1 that fall between the two cross points are copied into the same positions of the offspring. The remaining allele order is determined by parent2. Non duplicative alleles are copied from parent2 to the offspring beginning at the position following the second cross point. Both the parent2 and the offspring are traversed circularly from that point. A copy of the parent′s next non duplicative allele is placed in the next available child position [3].

#### 2.2.4.2 Modified Order Crossover (MOX):

Apply order crossover, then the allele of offspring from start to first cross point will be flipped, the allele from second cross point to end of offspring will be flipped. For example, consider parent tours:

**Parent1 (1 2 3 4 5 6 7 8 9)**

**Parent2 (7 4 1 9 2 5 3 6 8)**

Suppose that the two selected crossover points are (4 and 6), applying order crossover (OX) the offspring is

**Child (1 9 2 4 5 6 3 8 7)**

Applying modified order crossover (MOX) the offspring will be

**Child (2 9 1 4 5 6 7 8 3)**

### 2.2.5 Mutation Operator

The simple inversion mutation (SIM) is used for TSP, this operator selects randomly two cut points in the string, and it reverses the substring between these two cut points [3]. For example consider the tour:

**(2 5 8 7 1 3 4 6 9)**

And suppose that two cut points are (4 and 6) then the resulting strings will be

**(2 5 8 3 1 7 4 6 9)**

## 3. The proposed Genetic Algorithm to solve TSP with modified SCX

Sequential Constructive Crossover (SCX) is unconventional crossover operator which conserves the good sequential structure of the parent's chromosomes during creation of new child chromosomes. In this approach the child chromosome are constructed by sequentially choosing the nodes with minimum cost from parent chromosome [7].

### 3.1 The Modified SCX Crossover

MSCX different from SCX as shown in the proposed algorithm:
**Step 1:** Start from 'First Node' of the parent 1 (i.e., current node p = parent1 (1)).
**Step 2:** 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 from parent 1 and parent 2 (the first 'legitimate node' that is not yet visited from parent1 and parent2), and go to Step 3.
**Step 3:** 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.
**Step 4:** If $C_{p\alpha} < C_{p\beta}$, then select 'Node α', otherwise, 'Node β' as the next node 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.

### 3.2 The structure of MSCX crossover is:

1. P=parent1 (1)
2. For i=1: n (number of cities) offspring (i) =p
/* search the first 'legitimate node' (the node that is not yet visited) appeared after 'node p' in each parent.*/
Node α, node β are the first 'legitimate node' from parent1 and parent2
If α, β ∈ offspring node α, node β are the first 'legitimate node' (the node that is not yet visited) from parent1 and parent2 respectively
/* selection the next node of offspring*/
If $C_{p\alpha} < C_{p\beta}$ then node α is the next node p = α

Else node β is the next node p= β
End;

## 4. Test and Results

Input Data (coordinates of each city (X, Y)) are shown in Table 1. After running programs by using the Genetic Algorithm with Order Crossover Operator (OX) and Modified Order Crossover (MOX), the results show that OX and MOX give the same results for TSP (the same solution with different starting city) but from No. of cities 15 cities achieve different results (short path) as shown in the Table 2 and Fig. 1 MOX achieve the best results from OX. MSCX achieve the best results from OX, MOX and SCX Table 3, Table 4, Fig.2 and Fig. 3. Show these results.
**Sample 1:**

**Input Data**

| City | X | Y |
|---|---|---|
| 1 | 0 | 3 |
| 2 | 1 | 5 |
| 3 | 4 | 5 |
| 4 | 5 | 2 |
| 5 | 4 | 0 |
| 6 | 1 | 0 |
| 7 | 3 | 7 |
| 8 | 6 | 2 |
| 9 | 8 | 4 |
| 10 | 7 | 6 |
| 11 | 10 | 0 |
| 12 | 9 | 2 |
| 13 | 5 | 3 |
| 14 | 4 | 6 |
| 15 | 6 | 1 |
| 16 | 11 | 2 |
| 17 | 7 | 4 |
| 18 | 8 | 3 |
| 19 | 2 | 9 |
| 20 | 3 | 10 |

Table 1 X, Y coordinate of cities

**Output Data**

| Number of cities | Short path GA(OX) | Short path GA(MOX) |
|---|---|---|
| 5 | 15.6344 | 15.6344 |
| 6 | 16.7967 | 16.7967 |
| 7 | 18.8612 | 18.8612 |
| 8 | 20.3045 | 20.3045 |
| 9 | 23.6504 | 23.6504 |
| 10 | 24.9257 | 24.9257 |
| 15 | 34.5500 | 33.9444 |
| 16 | 36.9362 | 35.944 |
| 17 | 37.825 | 37.1805 |
| 18 | 38.226 | 38.048 |
| 19 | 42.3098 | 42.1952 |
| 20 | 44.3428 | 42.4478 |

Table 2 Short paths by using GA with OX

**Note**

The results show that using GA with MOX crossover gives results better than using OX crossover as shown in fig.1, which show that the results at no. of cities 5 to no. of cities 10 is the same solution (with different starting city), but the results at no. of 15 to 20 show that using GA with MOX gives solutions (short paths) better than using OX crossover. This means that when number of cities increases the number of solutions, population size and number of iterations increase. Fig. 1 show that solution at 20 using MOX is the best solution(short path)from using OX crossover and at population size and number of iterations less than using OX crossover.

**Sample 2:**

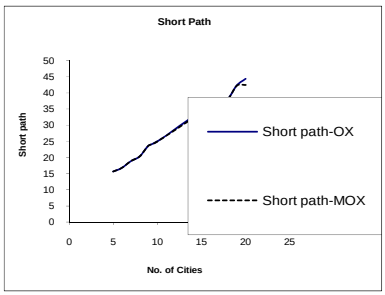


Fig. 1 Short Paths of MOX and MSCX

**Output Data**

| Number of cities | Short path GA(MOX) | Short path GA(MSCX) |
|---|---|---|
| 5 | 15.6344 | 15.6344 |
| 6 | 16.7967 | 16.7967 |
| 7 | 18.8612 | 18.8612 |
| 8 | 20.3045 | 20.3045 |
| 9 | 23.6504 | 23.6504 |
| 10 | 24.9257 | 24.9257 |
| 15 | 37.9843 | 33.2849 |
| 16 | 39.095 | 35.285 |
| 17 | 37.1805 | 36.0488 |
| 18 | 38.6543 | 36.2269 |
| 19 | 42.6673 | 41.6533 |
| 20 | 43.8575 | 41.9358 |

Table 3 Short paths by using GA

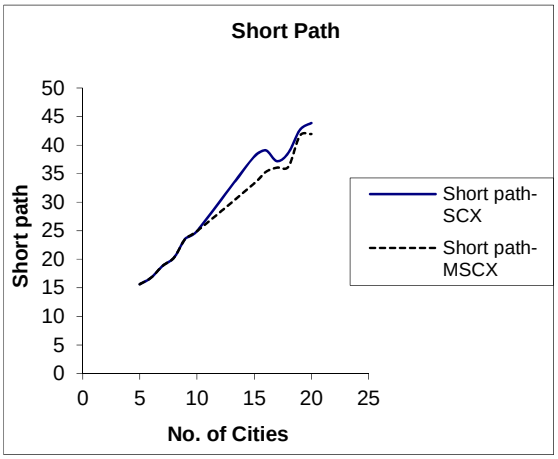


Fig. 2 Short Paths of SCX and MSCX

**Note**

Applying GA with MSCX crossover achieve best short paths (shorter) from using SCX crossover as shown in fig.2, which shows that GA with MSCX crossover gives good solution for TSP at no. of 15 to 20 , and at population size and number of iterations less than using SCX crossover. The solution at 20 using MSCX is the best solution (short path) from using SCX crossover.

**Sample 3:**

**Output Data**

| Number of cities | Short path GA(MOX) | Short path GA(MSCX) |
|---|---|---|
| 5 | 15.6344 | 15.6344 |
| 6 | 16.7967 | 16.7967 |
| 7 | 18.8612 | 18.8612 |
| 8 | 20.3045 | 20.3045 |
| 9 | 23.6504 | 23.6504 |
| 10 | 24.9257 | 24.9257 |
| 15 | 33.9444 | 33.2849 |
| 16 | 35.944 | 35.285 |
| 17 | 37.1805 | 36.0488 |
| 18 | 38.048 | 36.2269 |
| 19 | 42.1952 | 41.6533 |
| 20 | 42.4478 | 41.9358 |

Table 4 Short paths by using GA With MOX and MSCX at number

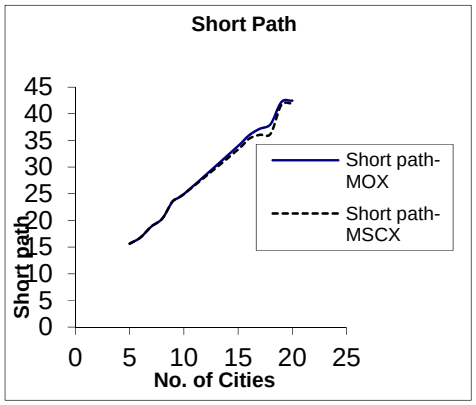


Fig. 3 Short Paths of MOX and MSCX

**Note**

The results show that using GA with MOX crossover gives results better than using OX crossover as shown in fig.1, which show that the results at no. of cities 5 to no. of cities 10 is the same solution (with different starting city), but the results at no. of 15 to 20 show that using GA with MOX gives solutions (short paths) better than using OX crossover. This means that when number of cities increases the number of solutions, population size and number of iterations increase. Fig. 1 show that solution at 20 using MOX is the best solution(short path)from using OX crossover and at population size and number of iterations less than using OX crossover.

## Conclusion

This paper proposed GA which can be effective to solve TSP with its new crossover MOX and MSCX. The results show at no. of cities 5 to 10 the short path is the same with using any crossover (with different starting city) and from no. of cities 15 to 20 using MSCX achieve good results better than the others compared GAs (PMX, OX, MOX, and SCX). The population size and number of iteration increase as number of cities increases. The GA with using MSCX crossover for solving TSP gives best solutions at small population size and iterations from using PMX, OX, MOX and SCX for the same problem.

## References

[1] S. N. Sivanandam, S. N. Deepa "Introduction to Genetic Algorithms", Springer, ISPN 10: 354073189X, 2007.

[2] David L. Applegate, RobertG.Bixby, Vasek Chvatal & Villiam J.Cook "The Traveling Salesman problem: a Computation study" 2007

[3] Buthainah Fahran Al-Dulaimi andHamza A. Ali, "Enhanced Traveling Salesman Problem Solving by Genetic Algorithm Technique (TSPGA)" World Academy of Science, Engineering and Technology 38 2008.

[4] D. Graupe, "Implementation of Traveling Salesman's Problem using Neural Network" Final Project Report (fall 2001) ECE 559 Neural Networks.

[5] B. P. Buckles, P. E. Petry and R. I. Kuester, "Schema Survival Rates and Heuristic Search in Genetic Algorithms", IEEE, 1990, PP 86-91.

[6] S. Ray, S. Bandyopadhyay and S. Pal, "New operators of genetic algorithms for traveling salesman problem" Cambridge: s.n., 2004, Vol. 2.

[7] Zakir H. Ahmed, "Genetic Algorithm for the Traveling Salesman Problem Using Sequential Constructive Crossover Operator", Journal of [8] Biometrics & Bioinformatics (IJBB) Volume (3): Issue (6) pp.96-105