AN ADAPTIVE CROSSOVER METHOD TO SOLVE MTSP USING REAL PARAMETER GENETIC ALGORITHM

Rayhaan Iqbal #50320850

University at Buffalo, New York, NY, 2020

ABSTRACT

In this paper, a new adaptive crossover method is proposed to solve the Multi Travelling Salesman Problem (mTSP) using real parameter Genetic Algorithm, to minimize the longest tour. The crossover operator used is dynamically changed with respect to the spread of the distances travelled by each salesman. A two-part chromosome is used to represent each candidate solution and crossover method is designed for the same. The current crossover methods have the problem of scarce diversity, computational expense and unnecessary breakdown of schema. Therefore, this paper attempts to propose a method which can help reduce all three of the above problems, by judicially varying the crossover operator in accordance to the deviation of the longest tour with the average tour length.

NOMENCLATURE

The major nomenclature used in this is as follows:

Table 1: Nomenclature

F	Longest Tour (objective function)
n	Number of cities
M	Number of salesmen
D_k	Distance travelled by salesman k
N_k	No of cities for salesman k to travel
c_i^k	i^{th} city travelled by salesman k
C^k	Set of cities travelled by salesman k
S	Starting depot
Overshoot	Overshoot of longest tour from mean of all tours
tol	*Specified tolerance* (allowed deviation)
$d(a,b)$	Distance between cities a & b
AC	Adaptive Crossover Method used in this paper.

1. INTRODUCTION

1.1 mTSP and Applications

The multi Travelling Salesman Problem (mTSP), is a modification on the well documented Travelling Salesman Problem (TSP). In the TSP, a single salesman has to cover a given set of cities, while travelling the minimum possible distance and visiting every city just once. In the mTSP, the same given set of cities have to be covered by multiple salesmen, with each salesman visiting at least one city. This problem can either be solved to minimize the cumulative distance travelled by the salesman or to minimize the longest tour, that is, the salesman to travel the longest distance. As a result of the sheer potential of variations on the basic TSP, its applications can be found in many other industries where the need to solve the same combinatorial problem with some modifications is imperative. This has resulted in massive amounts of research being done in this area.

Real life industrial applications of the mTSP are very well cited in [1]. *Print Press Scheduling*: multi-edition printing press scheduling for a periodical. *Crew scheduling*: Covering of given bus routes with optimum distance covered, while having constraints on time and loading of the bus. *Mission Planning*: finding the optimum path for autonomous robots to solve multiple tasks like construction, military reconnaissance, warehouse automation, post-office automation and planetary exploration. MTSP can also be used to solve certain Vehicle Routing Problems (VRP). Minimizing the vehicles required to fulfill a given number of customers in a with constraints on the distance is one of the types of VRPs where MTSP concept can be used.

1.2 Literature Review

The amount of research available on mTSP is quite substantial. Apart from GAs, it has been solved by a number of nature inspired methods. Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), are some of the good alternate ways of solving the mTSP problem. Apart from these, there have been numerous advances is existing GA variations, with a large amount of literature on the same. This paper, utilizes the concept of existing crossover methods to form a new adaptive crossover method which can provide better results.

Arthur E. Carter [5], in his paper solved the mTSP problem by proposing the Ordered Crossover (OX). This is very well regarded and commonly used operator for most TSP problems. This paper brought out the differences between chromosome types for solving the mTSP. Also, it highlights the importance and benefits of the two-part chromosome method which has also been incorporated for our algorithm. The benefit of reduced design search space, is a progress on the existing chromosome types. The OX suggested in this paper has its benefits in being easy to implement and computationally inexpensive. However, there is a major diversity problem that arises while using this crossover. This is mainly because the crossover method, does not operate so well on the second part of the chromosome. This problem is tackled in our algorithm.

Shuai Yuan [6], in his paper, has suggested a new two-part chromosome crossover (TCX) to solve the mTSP. The same two-part chromosome is used in this paper. In an attempt to solve the lack of diversity in existing GAs, this crossover method proposes a way to rearrange two parents to redistribute the cities of every salesman in with respect to the other parent. This helps in producing much more combinations and explores the design search space very efficiently. The drawbacks of this method is that it is computationally expensive and may see breaking up of schema more frequently.

1.3 Goal/Problem Statement

This paper aims to solve the mTSP problem, with the constraint of each salesman visiting at least one city to minimize the longest tour of the problem using a real coded Genetic Algorithm. A crossover method that can judicially decide when to break up and redistribute cities belonging to all salesman is proposed. It is an adaptive method to get optimum results with minimum computation. For the crossover of any two parents, the spread of the travel distances of all the salesman are examined. The longest tour's overshoot with respect to the average distance travelled by all salesmen is calculated in terms of the percentage of the said average distance covered. If this exceeds a set tolerance percentage, the second crossover technique is used. If the tolerance percentage is not exceeded, the first crossover technique is used. In essence, if the longest tour is not close to the average distance, it is considered as an indication towards potential increase in objective function by redistributing the cities travelled by all salesman. Thus, the salesman tour distances are used as a measure of the productiveness of the city distribution. This reduces breaking of schema by not redistributing salesman when not necessary. This also contributes to relatively reduced computation. This adaptive crossover is compared with the OX and TCX based operators to solve the Berlin 52 problem with multiple salesmen.

2. METHODOLOGY

2.1 Problem Formulation

The problem statement considers n cities to be traversed by m salesmen so as to minimize the longest tour. The constraint here being that every city must be traversed just once and no salesman should visit zero cities. This problem can be translated into mathematical terms as follows. The objective function being the longest tour of all the salesmen, is posed as the following minimization problem

Objective Function,

$$\min \quad F = \max(D_k) \qquad k \in (1{:}m)$$

Subject to,

$$n = \sum_{k=1}^{m} N_k$$

$$\bigcup_{i=1}^{m} \{C^i\} = \{1 \ldots n\}$$

Where,

$$D_k = \sum_{n=1}^{N_k-1} d(c_n^k, c_n^k + 1) + d\left(c_{N_k}^k, 1\right) + d(s, c_1^k)$$

The first constraint equation makes sure that the total number of cities traversed by all salesman is equal to the number of cities. Second constraint equation ensures that no city is left out. Together they take care of the constraint of visiting every city only once. To guarantee that every salesman travels at least one city, initialization is done appropriately as will be explained in the following section. In this way the constraints of the optimization problem are handled.

2.2 Initialization

Representation of a candidate solution is done as a two-part chromosome. The first part of the chromosome represents the cities to be traversed. The second part of the chromosome mentions the number of cities assigned to each salesman. A sample case of 10 cities and 4 salesmen, is demonstrated in the Fig 1. Here, salesman 1 is allotted 4 cities, salesman two is allotted 4 cities, salesman 3 is allotted 3 cities and salesman 4 is allotted 4 cities. Now, the first 4 cities in the first part of the chromosome represent the cities visited by salesman 1, in the same order. The next 4 cities represent the cities visited by salesman 2 and finally the same for salesman 3. A two-part chromosome technique is used over the two chromosome and the single chromosome techniques as it reduces redundant solutions considerably. Thus, the design variable search space is reduced by a great amount and the duplicity of candidate solutions is much lesser.

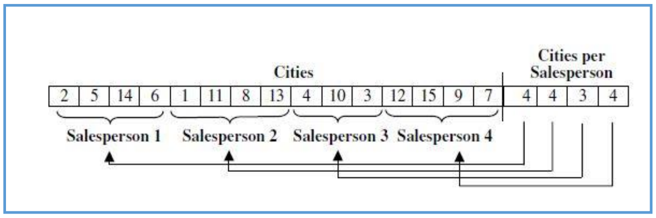


Fig1. Representation of the Two-Part Chromosome Technique

For initialization of the first part of the chromosome, numbers are randomly selected between 1 and *n* exhaustively for each member of the population. While initialization of the second part of the chromosome, care needs to be taken to make sure that the summation of the second part of the chromosome must equal to the total number of cities and also not have any salesman allotted zero cities. For this, we assign numbers to the salesman from 1 to *m.* Each salesman *j* is allotted a random number between 1 and U_j, where U_j is calculated as

$$Uj = \big(n - (m - j)\big) - \sum_{k=1}^{j-1} N_k$$

Where,

j goes from 1 to *m*

The above initialization would not only make sure that every city is visited only once, but also make every salesman visit at least one city.

2.3 Mating Pool

Size of the mating pool is taken the same as the population size. A stochastic remainder roulette wheel selection (SRWS) is used to create the mating pool. The fitness' of the population are evaluated and converted from their range to the range 0-1. At least N_i copies of each candidate are created as follows

$$P_i = \frac{f_i}{sum(f)}$$

$$N_i = floor\ (p_i * N)$$

Where,
N=population size
p_i = probability of selection of candidate *i*
f_i = fitness of candidate *i*

For the remaining positions in the mating pool, a normal Roulette Wheel Selection process is used by dividing a 360º space in accordance to the probability of each candidate and randomly picking solutions. The Stochastic Remainder Roulette Wheel Selection, avoids noisy selection by giving a fixed number of copies before the roulette wheel selection.

2.4 Crossover

2.4.1 Overshoot

After formation of the mating pool, crossover is performed to generate the offspring. For this, two parents are randomly selected from the mating pool to create two children. Examination of the spread of individual salesman distances travelled is carried out for each parent. For this we compare the values of the salesman with the maximum distance travelled with the average of the distances travelled by all the salesman. The difference is expressed in terms of a percentage of the average distance covered by the salesmen. In other terms the longest tour overshoot value is calculated with respect to the average of salesmen distances travelled.

$$Overshoot = (\mathrm{Max}\ (D_k) - \mathrm{mean}\ (D_k))\ /\mathrm{mean}\ (D_k)$$

If the above calculated value exceeds a defined tolerance, it implies distribution of cities among salesman is not uniform and a redistribution of cities travelled by salesmen might help. In this case, the TCX operator is utilized, whose working is later explained. On the other hand, if the overshoot is within a specified tolerance, it is assumed that distribution of the cities among the salesmen is uniform and consequently the OX operator is used. The setting of the tolerance parameter is explained later. Working of the two crossover operators used is explained below.

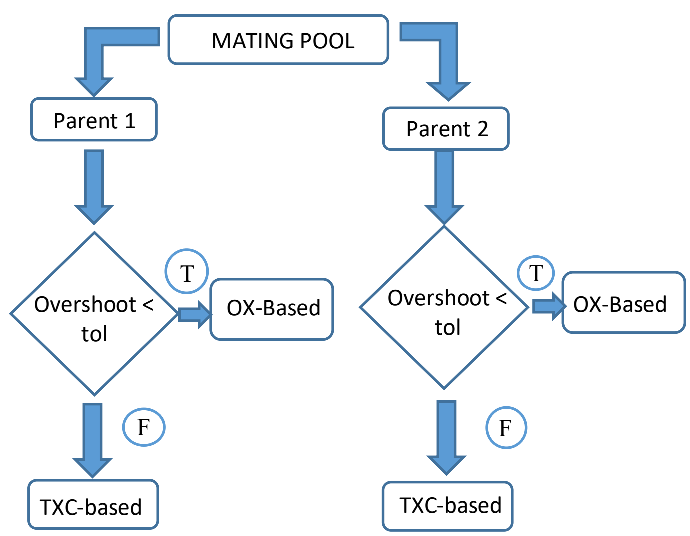


Fig 2. Adaptive Crossover operation

2.4.2 TCX Based Operation

In the TCX operator, every salesman is operated on individually, by changing their cities traversed. The diversity generated through this operator is substantial because of its reconstruction of the second part of the chromosome. Its working can be explained with the following example,

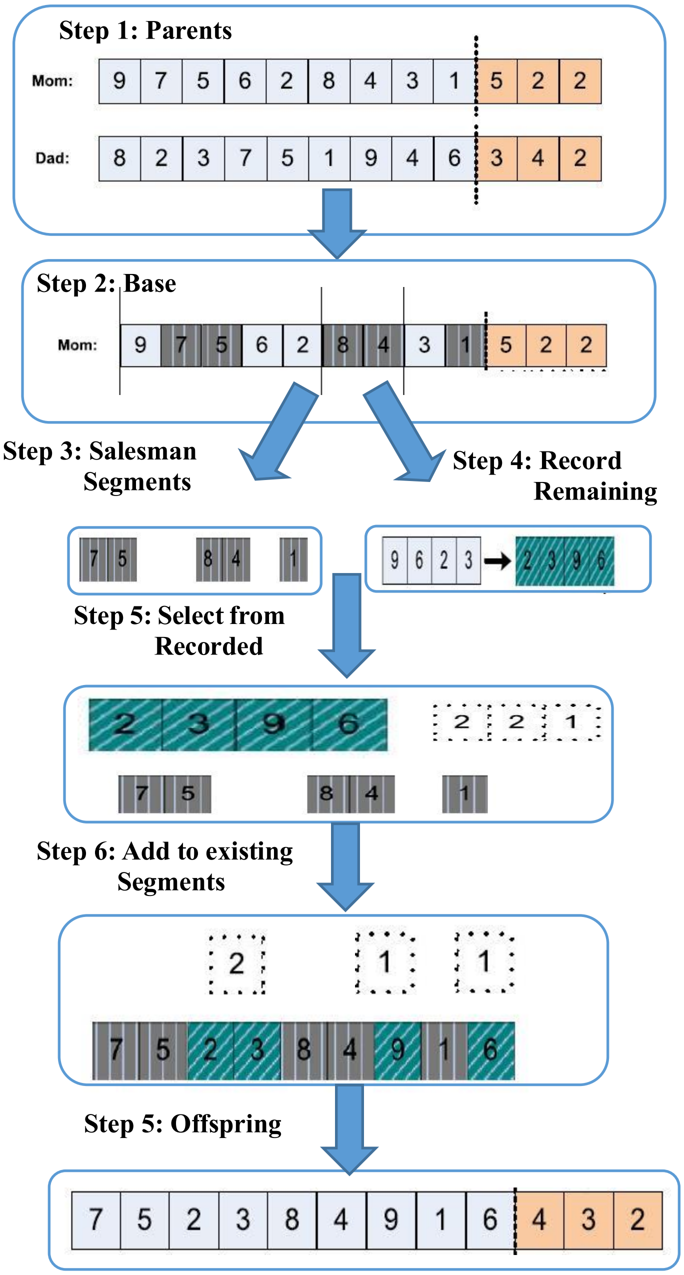


Fig 3. TCX Based Crossover

The First parent is used as the base for producing the child. A segment of the cities belonging to each salesman are randomly selected and assigned to the same salesmen of the offspring to be created. Now all the unselected cities from the base parent are rearranged as they appear in the secondary parent. Now, from these rearranged cities, random segments are selected exhaustively to add to each of the cities previously stored in the offspring. Depending on the cities allotted to each salesman, the second part of the chromosome is updated.

2.4.3 OX Based Operation

In the OX operator a segment of the first part of the chromosome parent taken as the base is randomly selected and passed on as it is to the offspring to be formed. The remaining part of the first chromosome is rearranged in the order that it appears in the second parent and then filled in the remaining slots of the first part of the chromosome in the offspring. For the second part of the chromosome, a random break point is selected and the order of values on each segment are simply reversed. The following Fig. 4 shows the working of the operator.

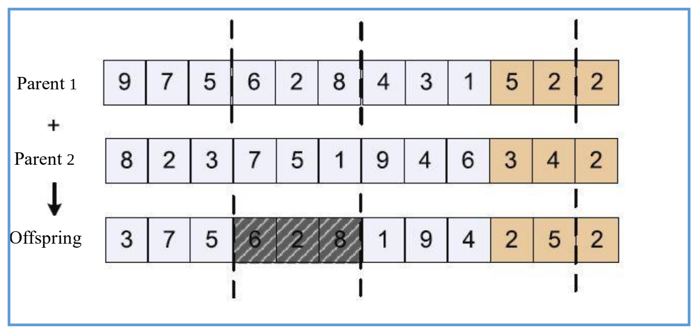


Fig 4. OX Based Crossover

It is clear that the diversity of this operator is a problem. This is owing to the fact that the second part of the chromosomes is not recombined with respect to both parents. Although this crossover operator is easy to implement and computationally efficient, it has a limited diversity.

2.4.4 Defining the tolerance

The crossover method applied to create an offspring depends on the overshoot calculated of the base parent and the specified tolerance, as mentioned previously. It is observed that a lower value of specified tolerance large fluctuations in initial iteration, faster average convergence per iteration and longer computation time. As we increase the value of specified tolerance, we see an inverse effect. A lower value of specified tolerance thus corresponds to a more exploitive approach. This paper utilizes a basic cooling schedule for the specified tolerance, which increases linearly with the number of iterations from a minimum specified to maximum specified. It can be demonstrated as below

$$tol = tol_{min} + itr * \frac{(tol_{max} - tol_{min})}{itr_{max}}$$

where,

itr = the current iteration number
itr_{max} = the maximum allowed iterations
tol_{min} = the starting value(minimum) of specified tolerance
tol_{max}= the starting value(maximum) of specified tolerance
tol = specified tolerance

The variation of tol with the iteration number can be shown by the following graph

2.5 Mutation

The mutation operator used in this paper is a straightforward single cell swap. This mutation is applied on both parts of the chromosome separately. Randomly, any two cells are selected from the first part of the chromosome and swapped with each other. The same process is carried out in the second part of the chromosome. This ensures that no constraints are violated while performing the mutation.

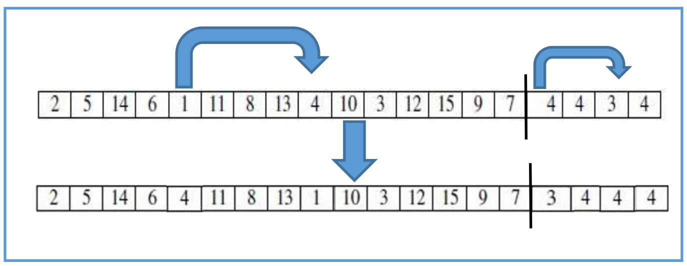


Fig 5. Mutation Working

2.6 Elitism

There is elitism applied in the algorithm. While taking the parents for the next generation, the offspring are compared with the parents of the current iteration. This is done by combining the offspring and parents of current iteration in one matrix and sorting them in descending order of their fitness'. Now the best values are selected to form the parents for the next generation.

2.7 Termination Criteria

For termination of the Genetic Algorithm, two conditions are provided. The first condition for termination is if the initially specified maximum number of function evaluations are reached. The number of function evaluations are the same as the number iteration executed as each iteration carries out 1 function evaluation. Thus, the algorithm is executed upon reaching the maximum specified iterations. The second condition for termination is if the there is no diversity in the entire mating pool. This implies that all the children are identical and no further improvement can be achieved. The algorithm terminates when any of the above two conditions are met.

4. RESULTS

4.1 Problem and parameter setting

The algorithm proposed in this paper is used to solve the Berlin 52 problem using multiple salesmen (3,5,7). The first city is considered as the starting depot for all the salesmen. To solve each of these mTSP problems, 10 runs of the genetic algorithm are carried out with 1000 iterations per run. The genetic algorithm parameters are shown in the table.

Table 2: Optimization Parameters

Run	10
itr_{max}	1000
Population Size	1000
tol_{min}	0.05
tol_{max}	0.2
Crossover Probability (C_p)	0.85
Mutation Probability (M_p)	0.2

4.2 Performance of Algorithm

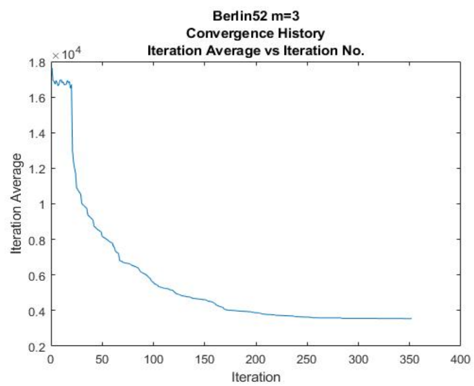
The berlin 52 problem for m=3,5,7 salesmen were solved for 10 runs and the results are shown below. The optimum path of all runs is compared with mTSP results from TSPLIB library [9]. Along with the longest tour, the total distance travelled is also compared. AC represents the adaptive crossover algorithm that has been used in this paper.

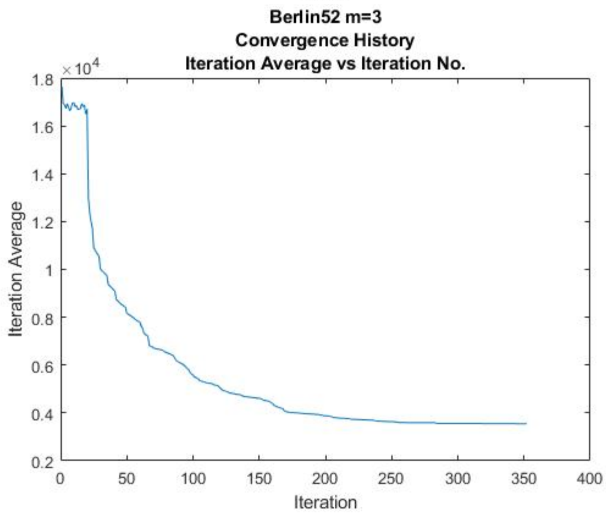
Table 3: Berlin 52 Results

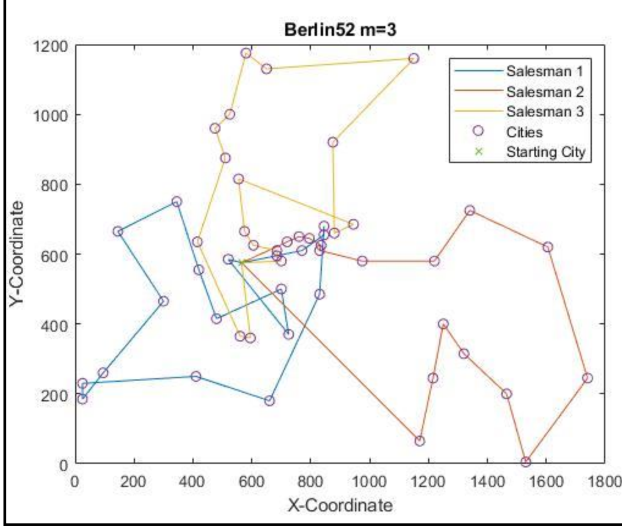
Method	No of salesman (m)	Longest Tour	Total travel distance
AC	3	3525.1	10316
	5	2663	16983
	7	2746.7	12310
TSPLIB	3	3244.37	9591.15
	5	2441.39	12084.9
	7	2440.92	16768.79

The optimum path over all runs and the convergence history for a random run is displayed for all problems:

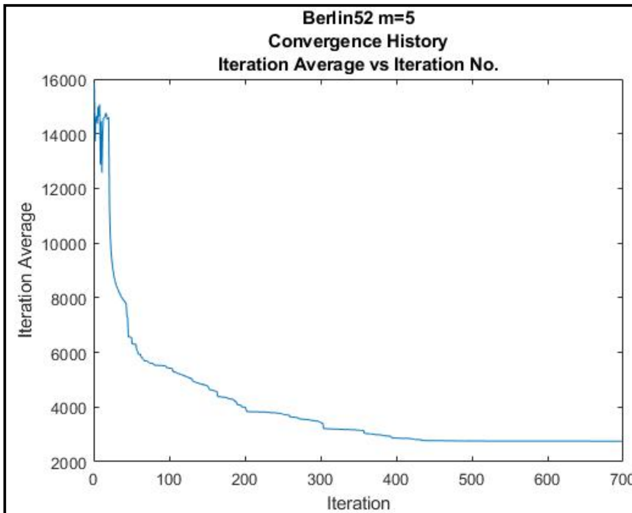
At m=3,

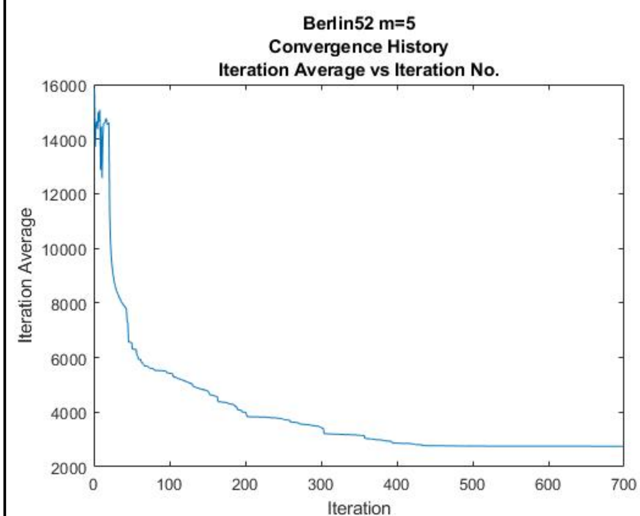

Berlin52 m=3
Convergence History
Iteration Average vs Iteration No.
Iteration Average
Iteration

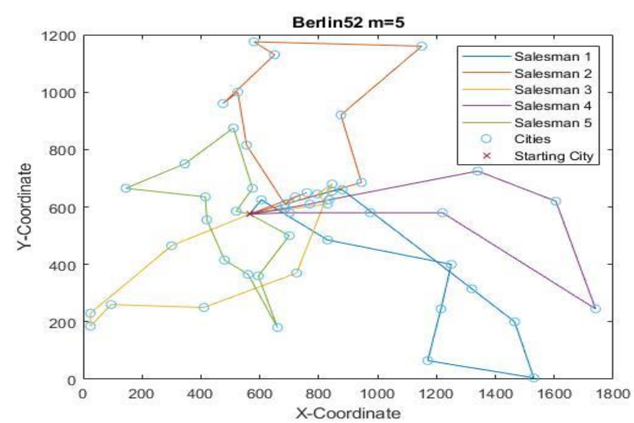

Berlin52 m=3
Convergence History
Iteration Average vs Iteration No.
Iteration Average
Iteration


Berlin52 m=3
Salesman 1
Salesman 2
Salesman 3
Cities
Starting City
Y-Coordinate
X-Coordinate

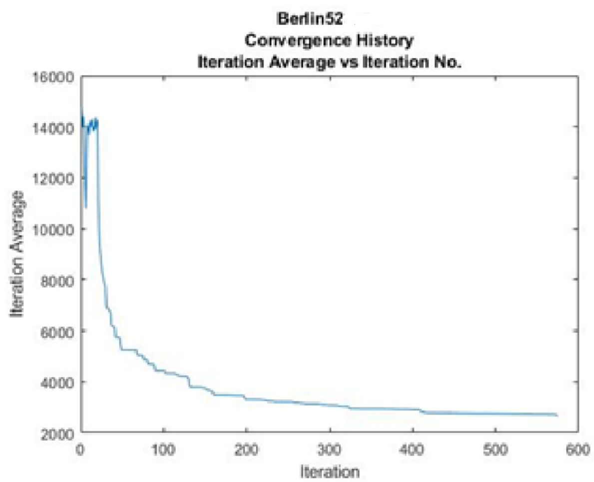
At m=5,

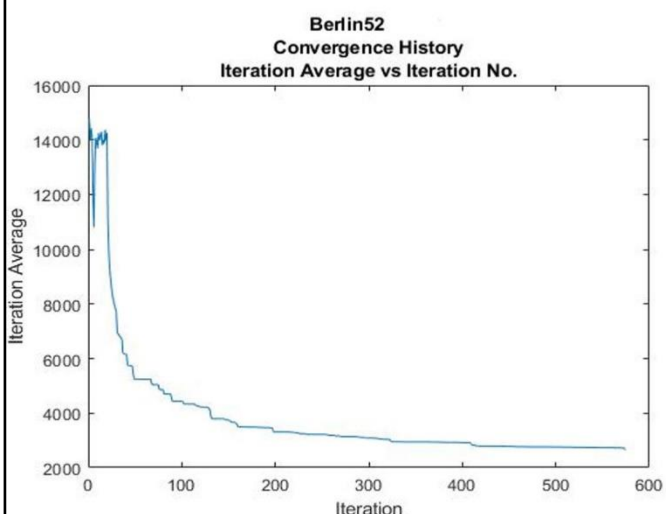

Berlin52 m=5
Convergence History
Iteration Average vs Iteration No.
Iteration Average
Iteration

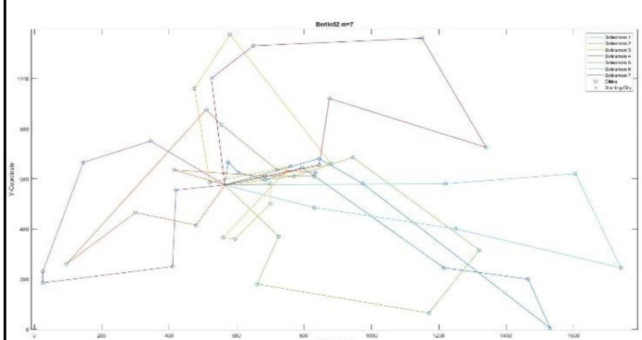

Berlin52 m=5
Convergence History
Iteration Average vs Iteration No.
Iteration Average
Iteration


Berlin52 m=5
Salesman 1
Salesman 2
Salesman 3
Salesman 4
Salesman 5
Cities
Starting City
Y-Coordinate
X-Coordinate

At m=7,







4.3 Comparison with other crossover methods

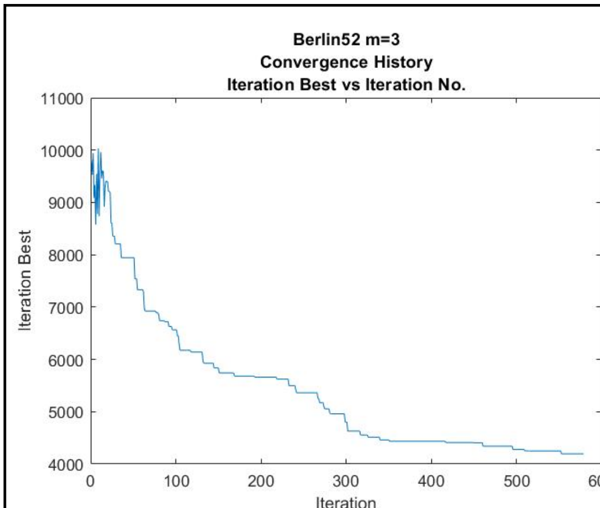
All results are compared with the OX and TCX operator for the Berlin 52 problem for multiple salesmen. All the optimization parameters were kept as the same in table 4**.** The best, mean and standard deviation of the longest tour over 10 runs is displayed.

Table 4: Berlin 52 Results & Comparison

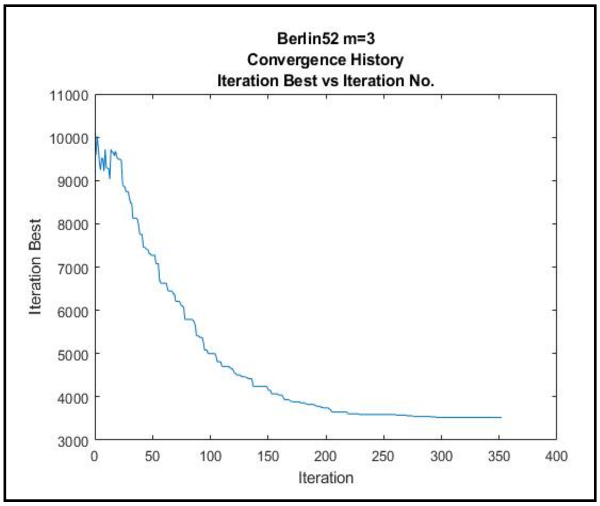
Salesmen	Crossover	Longest Tour		
		Minimum	Mean	Standard deviation
m=3	AC	3525.1	3935.3	291.2
	TCX	3780.9	3925.4	112.68
	OX	3451.8	3724.1	251.7
m=5	AC	2663	2819.3	101.4
	TCX	2787.6	3123.4	227.58
	OX	2746.7	2956.6	126.9
m=7	AC	2665	2711.2	33.4
	TCX	2706	2731	51.12
	OX	2906.8	3055.6	117.56

Also, for the Berlin 52 problem with 3 salesmen (m=3) the convergence history of all algorithms for a sample run are displayed for the 3 crossover operators.

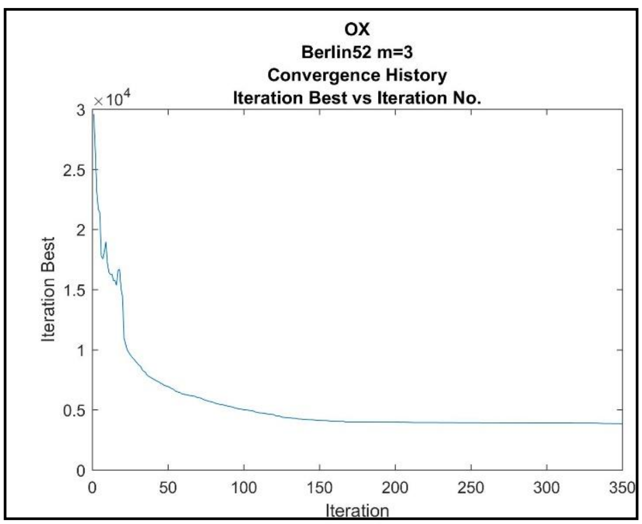
AC Convergence



TCX Convergence



OX Convergence



5. DISCUSSION

The proposed crossover operator has shown faster convergence than the compared operators. As a result of its adative nature, its computationally slightly cheaper than TCX but more expensive than OX. Its performance stays stable with the increase in number of salesmen. It however does pose its disadvantages as follows:

5.1 Advantages

- Difficult to implement
- As a result of the salesmen being identical, there are some redundant candidate formation linked with the two-part chromosome
- This minimizes just the longest tour and not the cumulative distance travelled by the salesmen

5.2 Future Scope

There is still quite some future scope linked with this algorithm

- Implement a local search like 2-opt to improve final results
- Solve the problem as a multi-objective problem to minimize both the longest tour as well as the cumulative distance travelled by the salesmen

REFERENCES

[1] Marco Dorigo, Senior Member, IEEE, and Luca Maria Gambardella, Member, IEEE ‘Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem’, IEEE Transactions of Evolutionary Computations, VOL. 1, April 1997

[2] Tolga Bektas, ‘The multiple traveling salesman problem: an overview of formulations and solution procedures’, Department of Industrial Engineering, Baskent University, October 2005.

[3] Payam Ghassemi, and Souma Chowdhury, ‘Decentralized Task Allocation in Multi-Robot Systems via Bipartite Graph Matching Augmented with Fuzzy Clustering’, Department of Mechanical and Aerospace Engineering University at Buffalo, July 2018

[4] Pan Junjie and Wang Dingwei, ‘An Ant Colony Optimization Algorithm for Multiple Travelling Salesman Problem’, Northeastern University Shenyang, China, 2006.

[5] Arthur E. Carter, Cliff T. Ragsdale, ‘A new approach to solving the multiple traveling salesperson problem using genetic algorithms’, College of Information Science and Technology, Radford University, Radford, United States, 2006.

[6] Shuai Yuan et al., ‘A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms’ , University of Technoogy Sydney, Australia, 2013.

[7] Ali A. R. Hosseinabadi et al., ‘GELS-GA: Hybrid Metaheuristic Algorithm for Solving Multiple Travelling Salesman Problem’, Iran, 2014.

[8] Yang Shuai et al., ‘An effective method for solving multiple travelling salesman problem based on NSGA-II’, 2019.

[9] https://profs.info.uaic.ro/~mtsplib/