Genetic Algorithm for the Traveling Salesman Problem

Omer Bolukbasi omerbolukbasi84@gmail.com

Department of Computer Science

Marmara University

Abstract

This paper contains the implementation of different crossover and mutation operators, on Travelling Salesman Problem. The proposed mutations are Inversion Mutation (IM), Insertion Mutation (ISM), Swap Mutation and Random Mutation (RM) which is a random selected mutation among ISM, SM and IM. As crossover operators Ordered Crossover (OX) and Sequential Constructive Crossover (SCX) are selected. Experimental results show that for small scale (having less than 100 cities) TSP topologies OX-RM combination has the highest performance while for large scale TSP topologies SCX crossover performs better. In case of applying 2-Opt operation to the population with the best combinations of mutation and crossover, the result is significantly improved.

Keywords: Traveling salesman problem, NP-complete, Genetic algorithm, Mutation and Crossover Combinations, 2-Opt Operator

---

1. INTRODUCTION

Travelling Salesman Problem (TSP) is an old NP Complete Problem that has been studied for decades in Computer Science researches. The problem consists of finding the best route over a certain number of cities. The route must include each and every city just one times and ends at the node where it starts.

In this research, various mutation and crossover operators, Steady State GA, Nearest Neighbor and Random population initialization and 2-Opt operator techniques are implemented for two sample TSP datasets during the experiments.

2. GENETIC ALGORITHMS

2.1 Initialization

2.1.1 Random Initialization

This initialization technique requires random selection of cities to compose individuals for population. The random selection distribution is uniform. That means each city has the probability of selection P(c) where N_c is the number of cities.

$$P(c) = \frac{1}{N_c}$$

2.1.2 Nearest Neighbor Initialization

Initialization is done based on the distance calculation for each selected city for the individual. If the Euclidean Distance is used as weights in EA then the following formula is used. X and y are the coordinates of cities in 2 dimensions.

$$distance_{city_{1,2}} = \sqrt{(x_{city2} - x_{city1})^2 + (y_{city2} - y_{city1})^2}$$

First a random city is selected. Then the distance between the selected city and all other cities are calculated with the Euclidean formula. The city that has the least distance with the previously selected city is selected as the second node for the individual. This process continues till no city remains unselected.

2.2 Representation and Performance Metric

For TSP, permutation representation is used. All individuals are represented as number arrays. Since the routes are cyclic, the node in the array is also connected to the first node of the array.

The total distance between all cities over a route (individual) is used as fitness indicator for that individual. Since the array is cyclic the distance between the last node and the first node is also added to the fitness calculation. So, for N_c cities fitness is calculated as follows:

$$Fitness = distance_{city_{N_c,0}} + \sum_{i=0}^{N_c-1} distance_{city_{i,i+1}}$$

2.3 Crossover Operator

Crossover is an operation that produces one or more children from selected parents. For permutation representations Ordered and Sequential Constructive Crossover are common for crossover operation.

2.3.1 Ordered Crossover

Ordered crossover produces two children from given two parents. Randomly choosen two nodes and the nodes between them are preserved in the offspring. [1]

P_1=(3 4 8 | 2 7 1 | 6 5),

P_2=(4 2 5 | 1 6 8 | 3 7).

The pipes are the randomly selected indices of the arrays. And the nodes between the cuts are copied to offsprings.

O_1=(X X X | 2 7 1 | X X),

O_2=(X X X | 1 6 8 | X X).

After insider nodes copied from parents to childs accordingly, then the rest nodes is copied from cross parents in the same order while preserving uniqueness of the nodes. For instance for O1 after 2,7,1 the nodes of P2 are added in the same order: 3->7->4->2->5->1->6->8. If any of the nodes are already in the offspring then this node will be skipped. And the same operation is done for O2.

O_1=(5 6 8 | 2 7 1 | 3 4),

O_2=(4 2 7 | 1 6 8 | 5 3).

2.3.2 Sequential Constructive Crossover

SCX constructs better individuals than its parents. The first node of the offspring is the first node of the first parent. Sequentially both of the parent nodes are searched and the first 'legitimate node' (the node that is not yet visited) appeared after previously selected node in each parent is considered. If no legitimate node after the previously selected node is present in any of the parent, the nodes are searched sequentially {2, 3, …, n} and the first legitimate node is taken as nominee. After the nominees from both parents are selected than the distances with the last node in child and the nominees are calculated and the node that has the least distance is selected as next node. This process continues till no available places remain in child.[4]

2.4 Mutation Operator

Four common types of mutation operators are investigated for TSP. These are the Insertion Mutation (ISM), Inversion Mutation (IVM), Swap Mutation (SM), and the Random Mutation (RM).

2.4.1. Inversion Mutation

In inversion mutation two cut points are selected randomly. All nodes in between these two points are placed as reversed in the individual with using individual's original indices. [3]

P = (2 3 | 4 5 6 | 1 7 9 8)

In the previous individual the cut points are selected randomly and placed in reverse order in mutated individual.

P = (2 3 | 6 5 4 | 1 7 9 8)

2.4.2 Swap Mutation

Swap Mutation simply swaps two randomly selected nodes of an individual.

2.4.3 Insertion Mutation

In the Insertion Mutation two indices selected randomly. The node in the first indice is placed after the second selected indice. The nodes after the second indice shifted right by 1 accordingly. Since the last node can not be shifted, it is placed in the first indice of the individual.

2.4.4 Random Mutation

Random mutation operator selects a random mutation operator among Inversion, Swap and Insertion Mutation. So, the probability of 3 mutations are equal after all crossover and mutations are completed.

2.5 Selection

2.5.1 Parent Selection

The parent selection process is done among a mating pool. The individuals in mating pool is selected randomly from population. After the mating pool constructed, the two parents that have minimum total distance will be elected as mating parents.

2.5.2 Survivor Selection

Survivor selection algorithm is Steady-State GA. In each iteration, only a part of the population is replaced by the offsprings. Child1 and child2 are replaced with random parents in the existing population.

2.6 2-Opt Local Search Operator

2-Opt Operator removes two randomly selected edges from the tour, and then it reconnects the two paths created. There is only one way to reconnect the two paths so that the result is also a valid tour. The structure of the operator is similar to inversion mutation. Both techniques select two cuts and connect newly created sub-route to existing sub-route in reverse order.

3 EXPERIMENTS

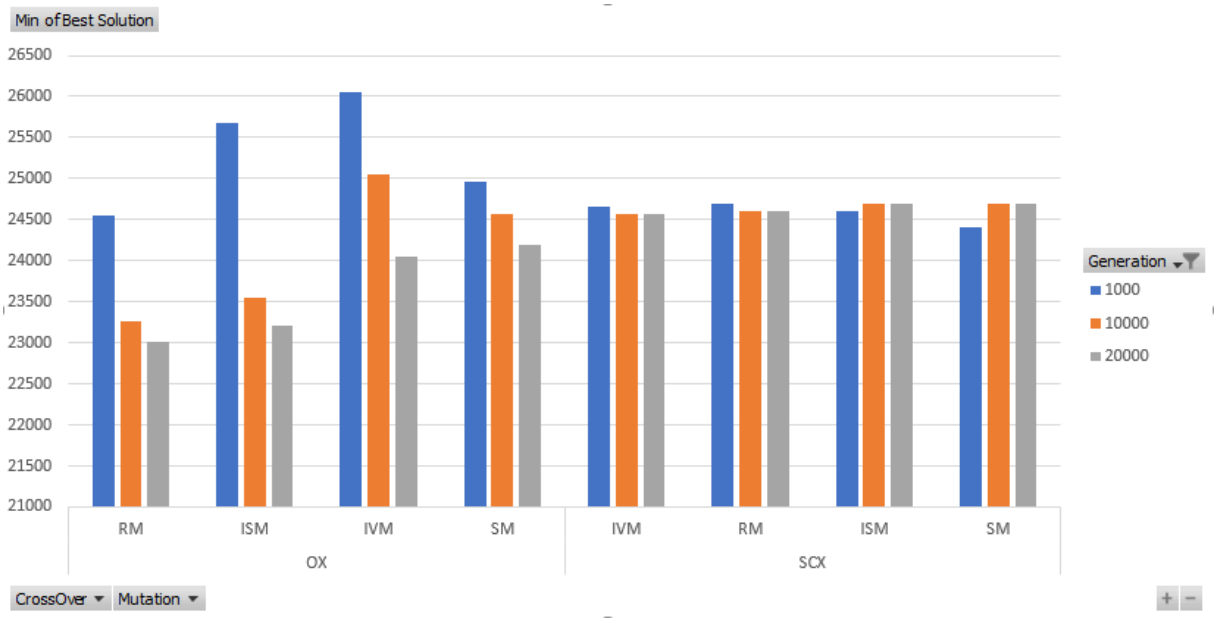
3.1 Varying the Crossover and Mutation Types

All combinations of the following Mutation and Crossover operators are applied on 2 different TSP topoloji over TSP EA simulation. The best result and average result (of the population) after 1000, 10000 and 20000 generations, for each case are presented.

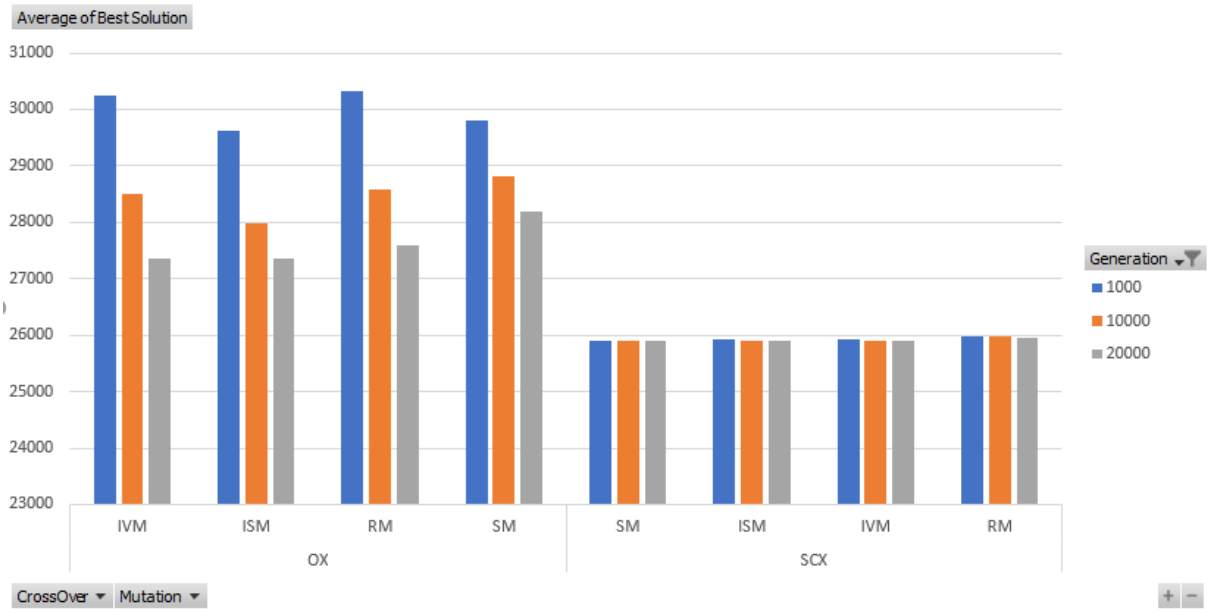
Parameter	Value	Explanation
Crossover Operators	OX, SCX	
Mutation Operators	ISM, IVM, SM, RM	
TSP Topoloji	kroA100.tsp, a280.tsp	2 TSP Topology with 100 and 280 cities.
Population Size	50	Number of individuals.
Mutation Probability	0.1	Probability to apply mutation after crossover.
Mating Pool Individuals	5	Number of selected individual for Mating Pool.
Iteration	100	Number of iterations for maximum generation.
Generation	1000,10000,20000	Number of generations for each iteration.

3.1.1 Kro100.tsp Results

The following graph shows that for 100 city case, the best performing CrossOver/Mutation combination is **OX/RM** with the best distance **23014** over 20000 generations where the optimum solution is **21282.** Other important insight here is that, SCX is not able to develop the first fitness value.



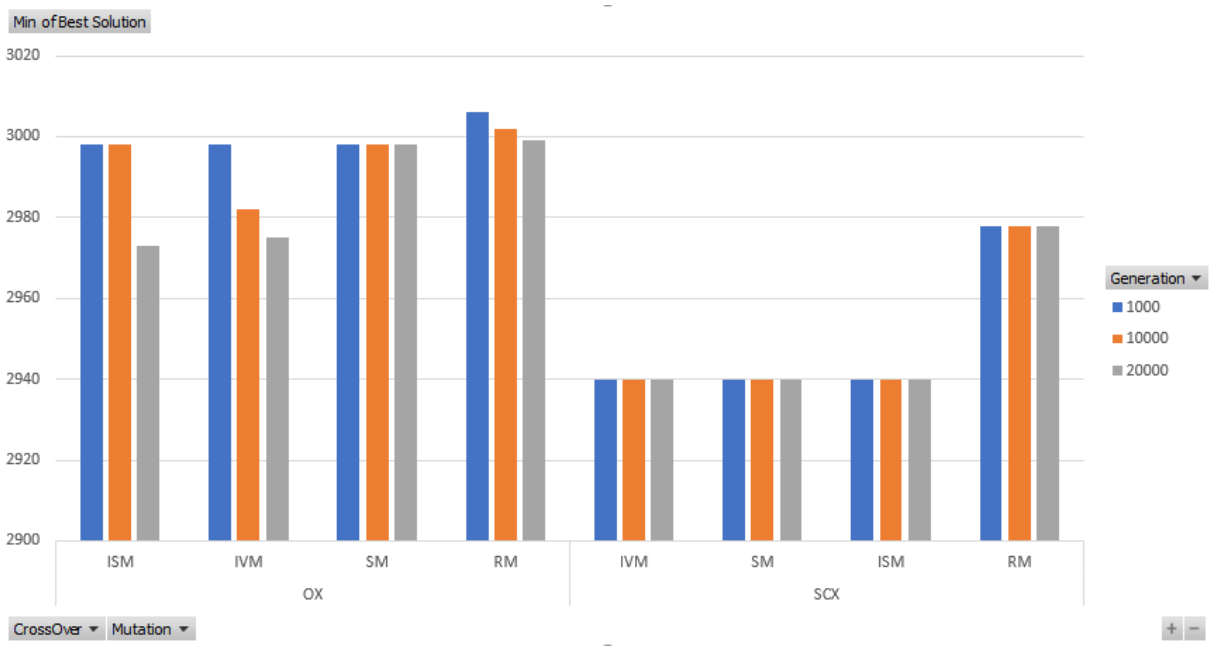
The average of 100 iteration of the best cases are as follows.



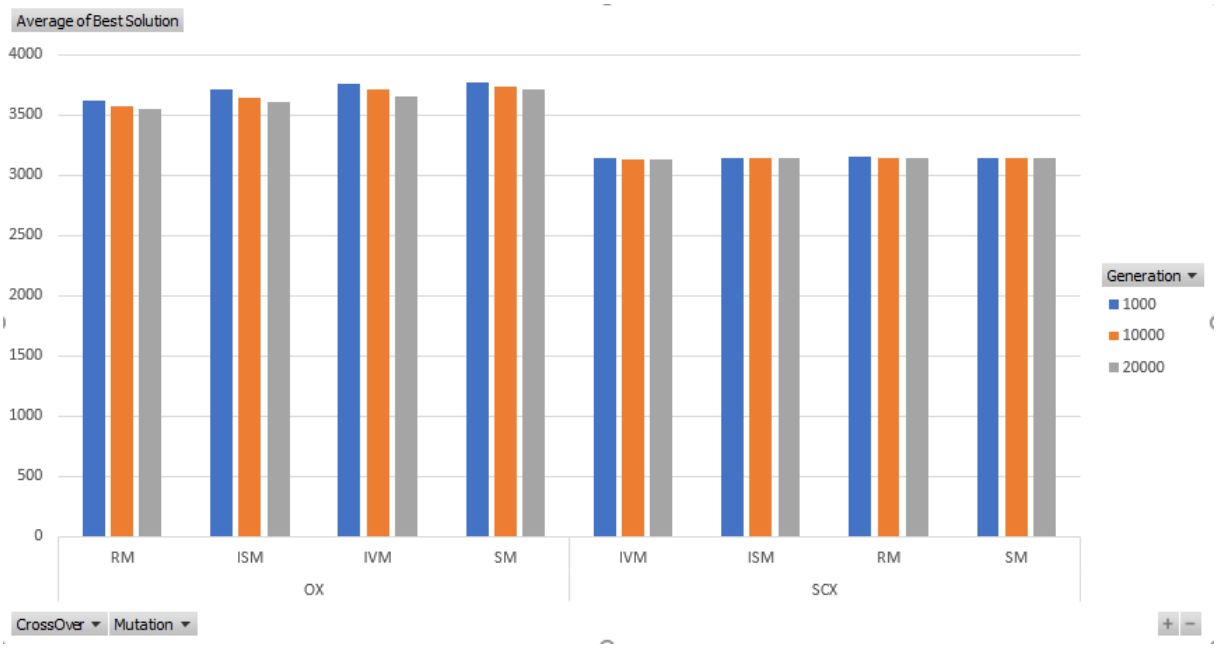
From the experiment results it is derived that OX's best solutions average of 100 iterations over generations develop significantly. However for SCX the average of best cases remain same over over generations even getting worse for RM mutation.

3.1.2 A280.tsp Results

The following graph shows that for 280 city case, the best performing CrossOver/Mutation combination is **SCX/IVM** with the best distance **2940** over 20000 generations where the optimum solution is **2579.** Same as in first part, SCX is not able to develop the first fitness value.



Average of best cases over 100 iterations.



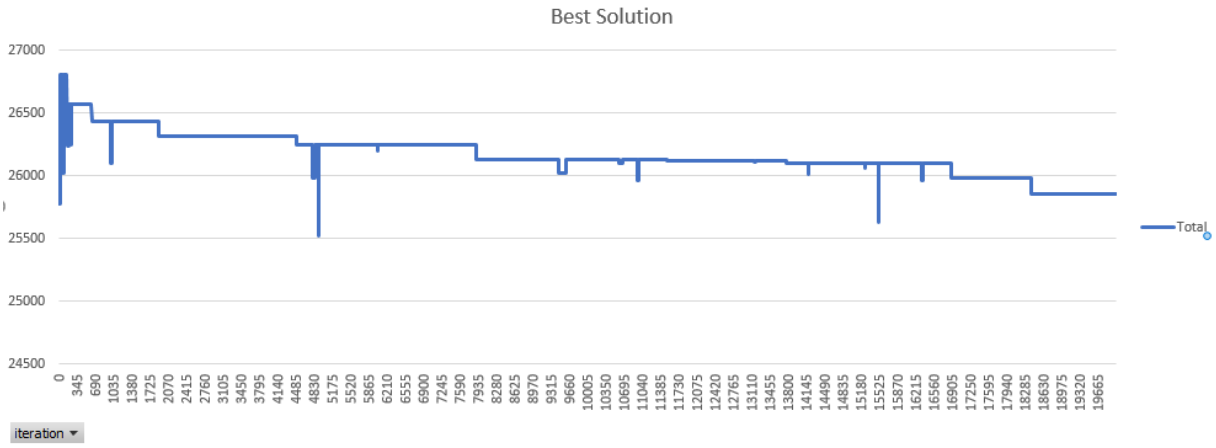
3.2 Performance of GA with Respect to Number of Generation

With the best and worst combinations of the previous experiment, the best and the average solutions of the population is measured in each generation and presented.

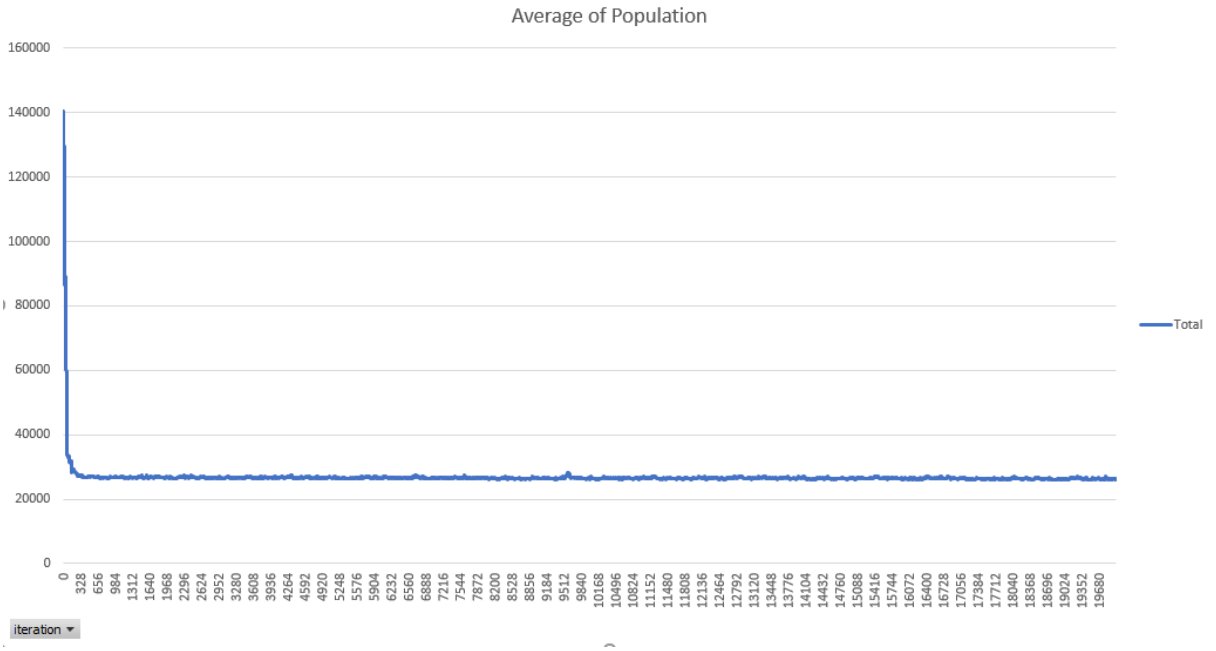
3.2.1 Kro100.tsp Results

3.2.1.1 Best Case

For the Best Case measurement OX and RM are selected as Cossover and Mutation operators. The best solution of kro100 topology getting better while passing generations.

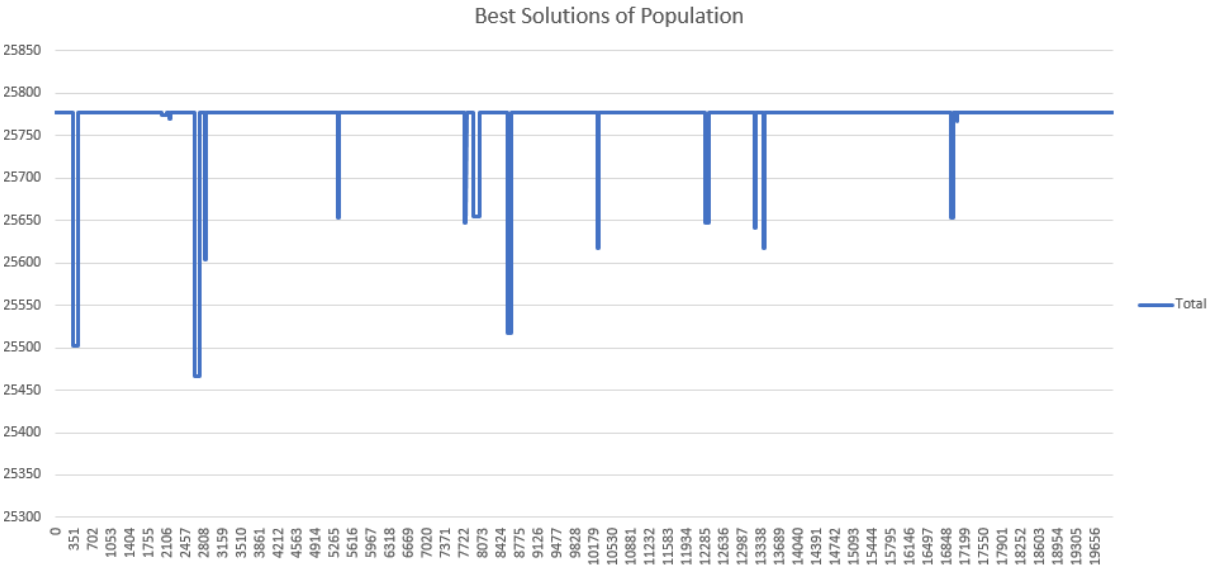


The average of the population significantly decreases at first 200 generations. However after 400 generations the average remains same till to 20000 generations.

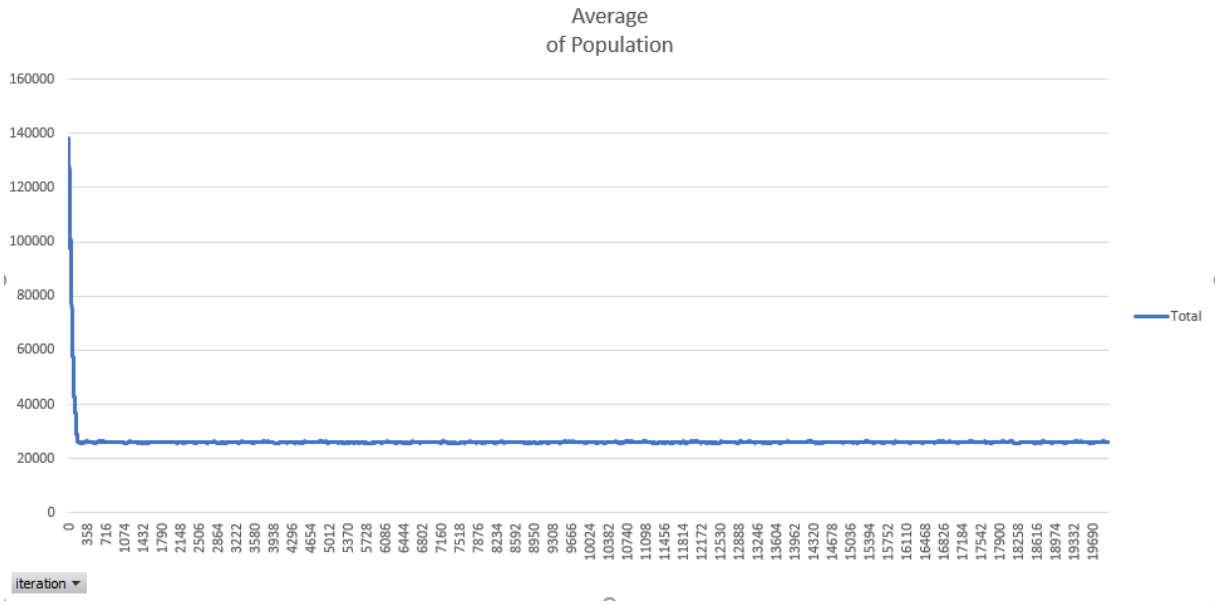


3.2.1.2 Worst Case

For the worst case, SCX and IVM are selected as crossover and mutation operator. The measurements show that although some good results obtained during generations but they disappear at the following generations. So the best solution remains same even after 20000 generations.



The average solution of the population significantly get better after 100 generations. And remains same till to the end of 20000 generations.

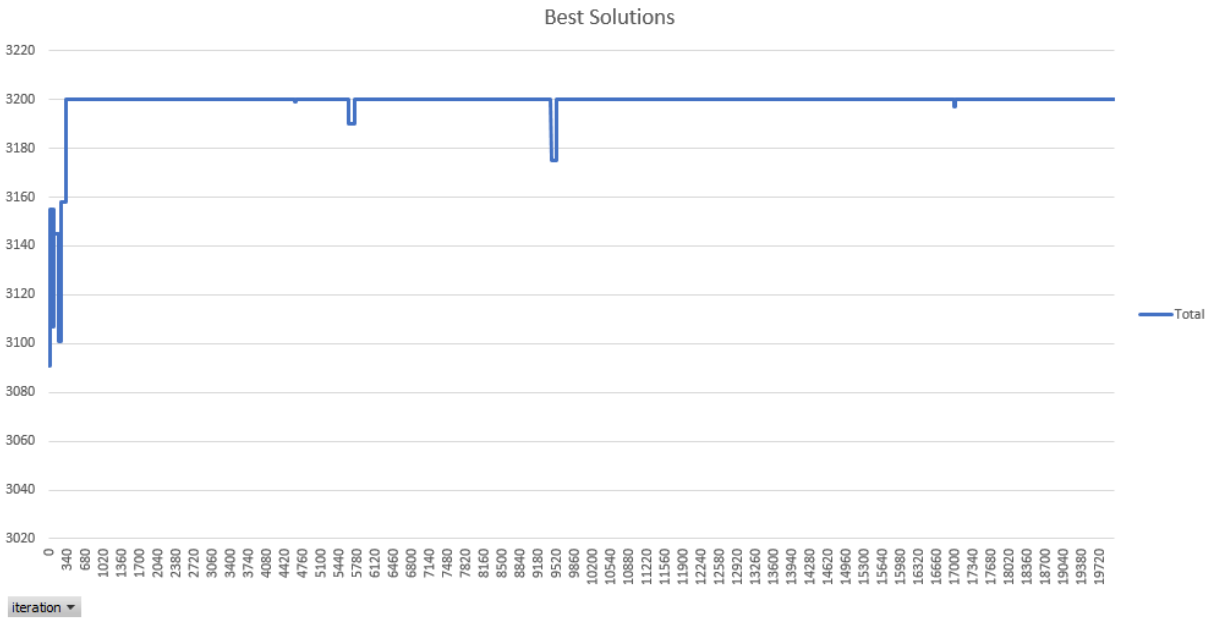


3.2.2 **A280.tsp Results**

For A280 SCX and IVM are selected as crossover and mutation operators.

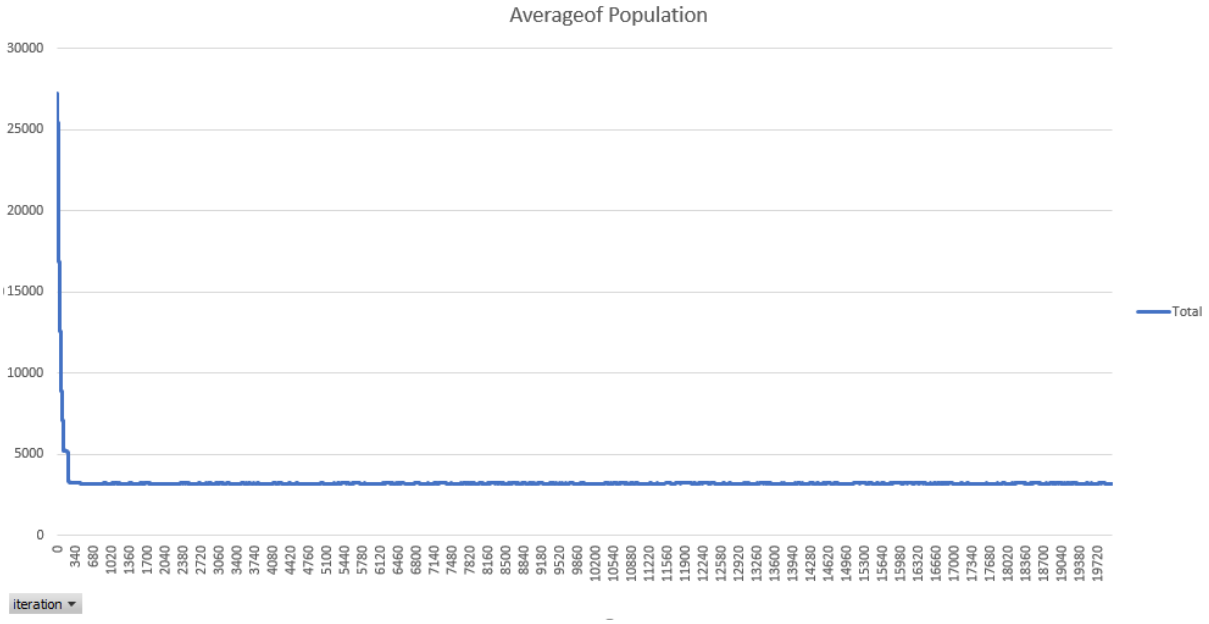
3.2.2.1 Best Case

For the first 200 generations best solutions are better however the at the remaining generations the best solutions stabilized on a constant value.



3.2.2.2 Worst Case

In the worst case scenario, the average of population significantly decreases in 200 iterations. After that it remains constant till to the end of 20000 generations.



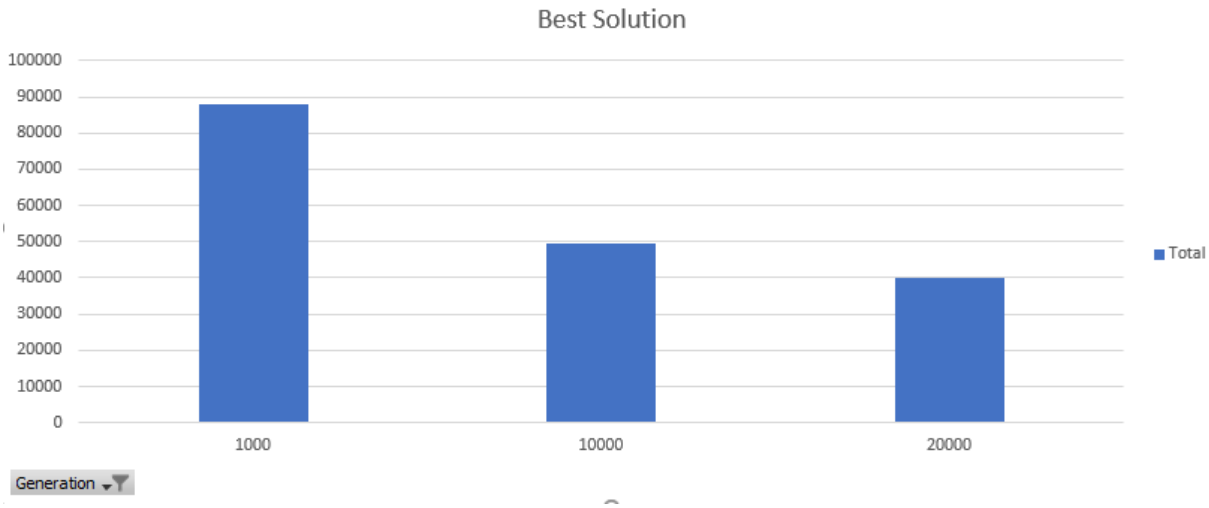
3.3 Effect of Initial Population on Performance

In this experiment, different initial conditions are proposed. In the first case %100 of the population initiated randomly. In the second case the initialization rates remains same as in the previous experiments.

3.3.1 Kro100.tsp Results

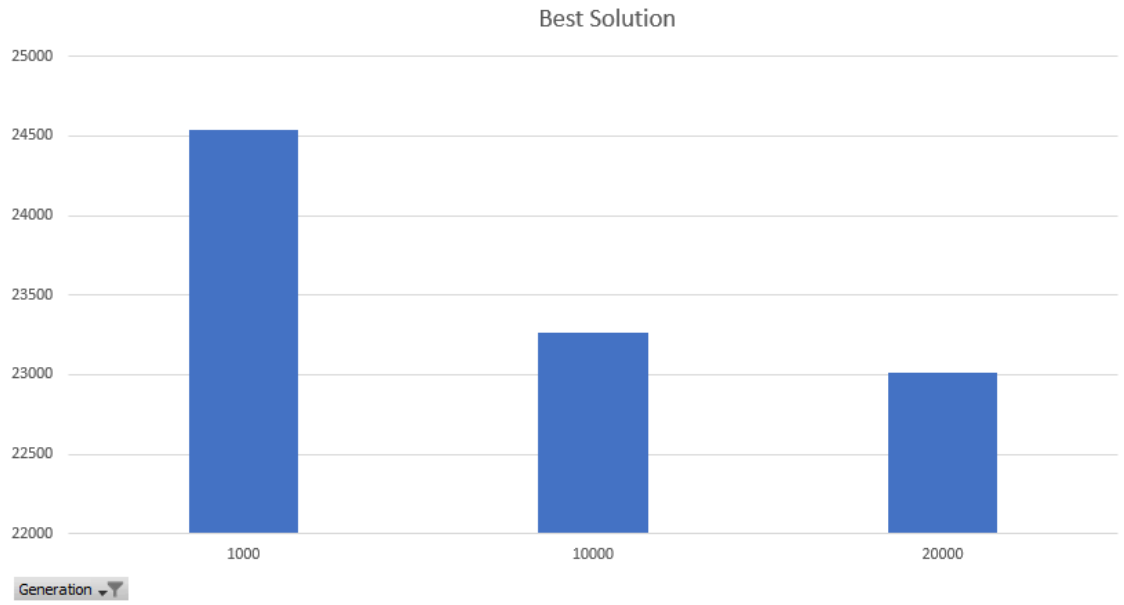
3.3.1.1 CASE1

For this case, the initial individuals composed randomly and OX and RM are used as operators. And the best of 100 iterations is impoved over generations. However the final solution (39864) is not good at the best result of previous experiment.



3.3.1.2 CASE2

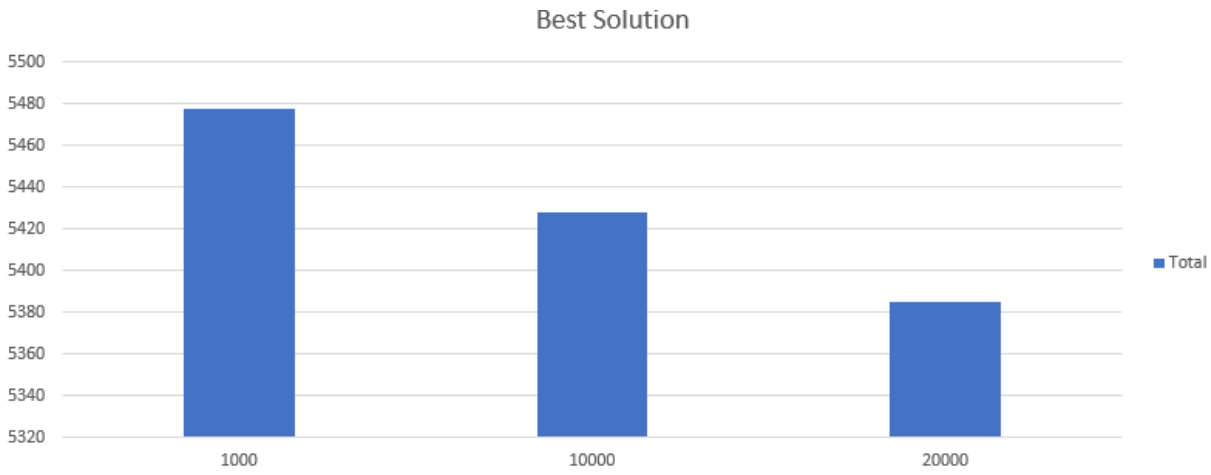
In case2 the same comfiguration as in Experiment 1 is used. So the best Solution is 23014 again.



3.3.2 A280.tsp Results

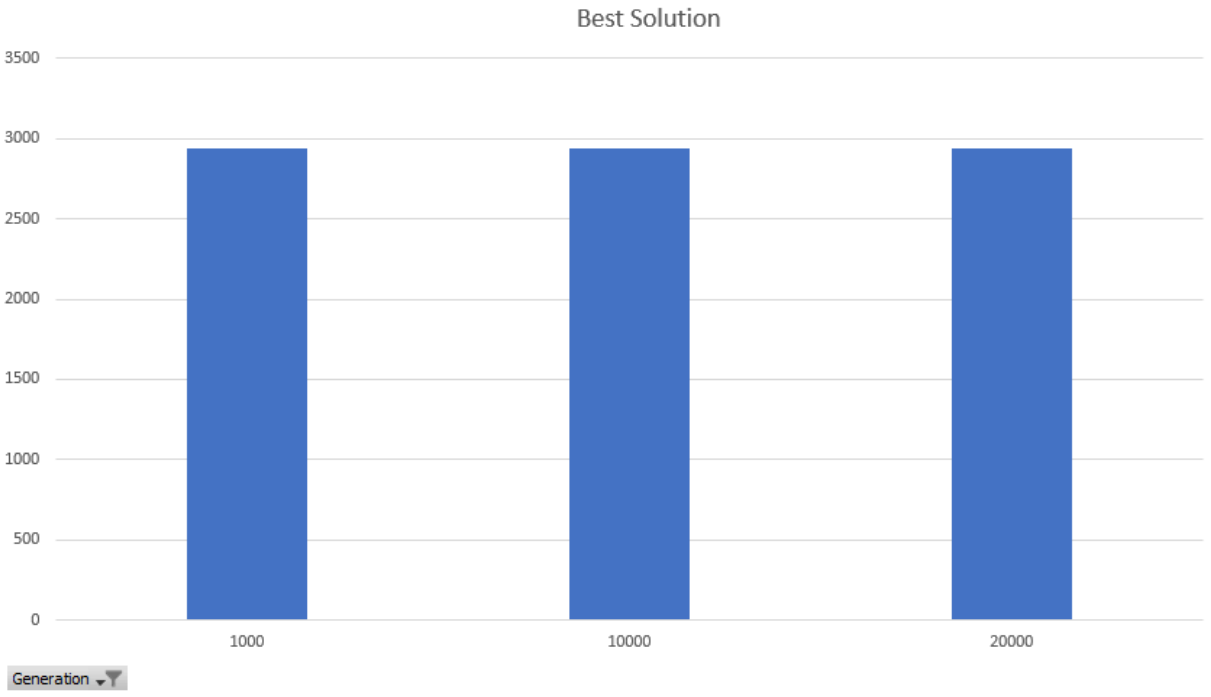
3.3.2.1 CASE1

For this case, the initial individuals composed randomly and SCX and ISM are used as operators. And the best of 100 iterations is improved over generations. This result is not same as in first experiment. In the first experiment the best results are remains same even in 20000 generations. But in this case the best results are getting better. Note that in this combination all initialization of the population is done randomly. The final solution is 5385.



3.3.2.2 CASE2

Best solution 2940 is same as previous experiment. The best solutions does not change during generations.

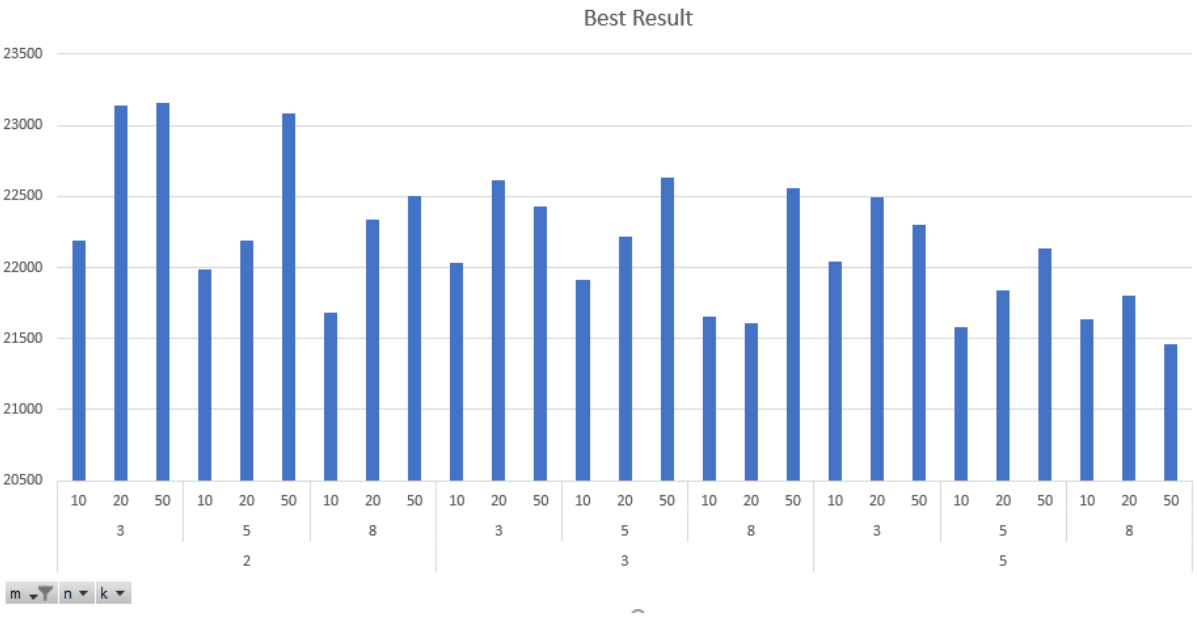


3.4 Improving the Performance of Tours

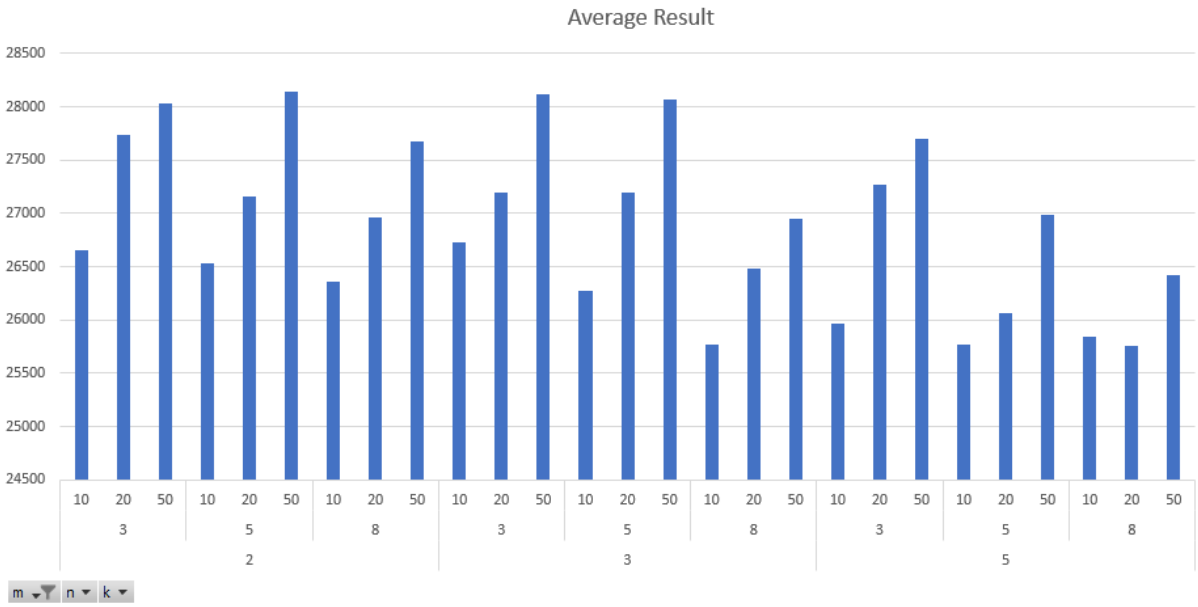
In this experiment 2-Opt operator is applied to the population with the presented parameters. OX and RM are used as crossover and mutation operators over 100 iterations.

Parameter	Value	Explanation
k	10,20,50	Procedure is applied in every k Generations/iterations
m	2,3,5	Number of randomly selected individuals in each iteration.
n	3,5,8	Number of times that the operator is applied to the individual.

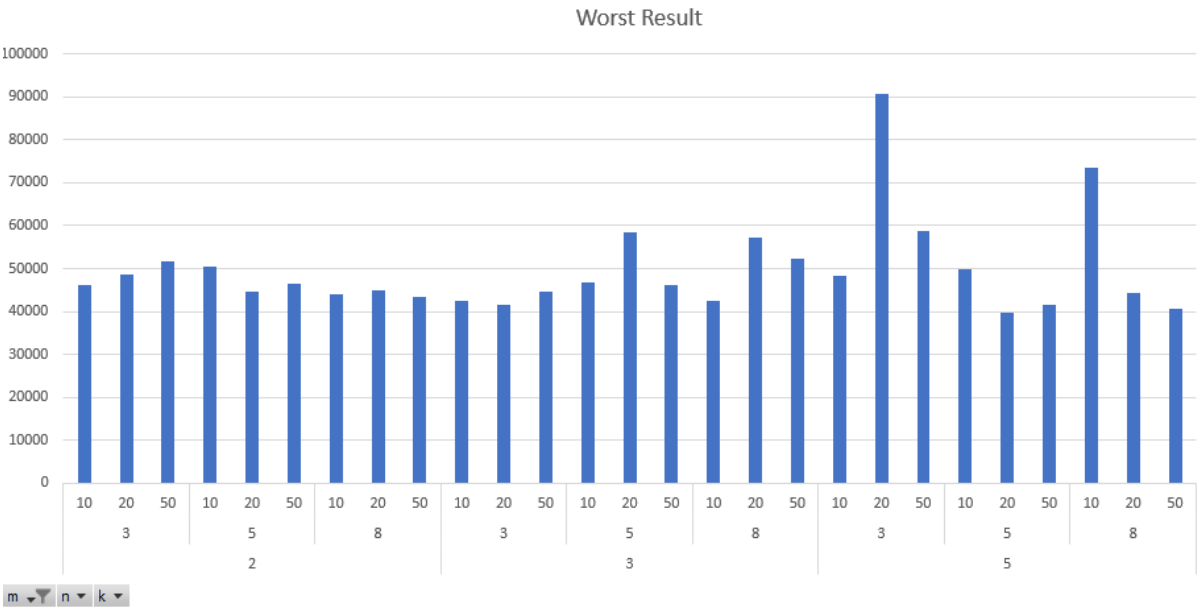
After 100 iterations the best solution is **21459** with the combination m=5, n=8, k=50 which is significantly higher than the first experiment's best result.



The best of average result of the population is m:3 n:8 k:10.



The worst case is m:5 n:3 k:20 which is significantly high in comparison with other combinations.



4. CONCLUSION

The initialization rates of the population have high effect on the performance of crossover and mutation operators. It is observed from the first and third experiment that if the population initialized fully random, then the SCX crossover has high optimization rates on population for the further generations. If Nearest Neighbor process implemented to the population in the initialization phase, then SCX can't optimize to the best route. Because the best routes already inserted in the initialization with nearest neighbor approach. The structure of SCX is similar to the nearest neighbor approach because the best distance is calculated from the parents. In SCX, always the nearest neighbors within the parent nodes can be selected, not the nearest neighbors within de whole nodes like in the initialization. So, this is a disadvantage of SCX in comparison to nearest neighbor initialization. That is why SCX can't overcome the best solutions that is already initiated in the population during the initialization phase.

OX is not effected from the initialization rates. For the ordered crossover the fitness rate is not considered which increases the randomness. So new best solutions can occur on new generations.

2-Opt operator has a significant optimization on the population. It is observed from the fourth experiment that, higher the n and m, better the performance. Such that, the best result 21459 is close to the optimum solution 21282 for kro100.tsp.

5. REFERENCES

[1] Abid Hussain, Yousaf Shad Muhammad, M. Nauman Sajid, Ijaz Hussain, Alaa Mohamd Shoukry, Showkat Gani, "Genetic Algorithm for Traveling Salesman Problem with Modified Cycle Crossover Operator", Computational Intelligence and Neuroscience, vol. 2017, Article ID 7430125, 7 pages, 2017.

[2] Zakir H. Ahmed, “Genetic Algorithm for the Traveling Salesman Problem using Sequential Constructive Crossover “, International Journal of Engineering and Sciences & Research Technology,2018.

[3] Kusum Deep, Hadush Mebrahtu, “Combined Mutation Operators of Genetic Algorithm for the Travelling Salesman Problem”, International Journal of Combinatorial Optimization Problems and Informatics, Vol. 2, No.3, Sep-Dec 2011.

[4] Zakir H. Ahmed, “Genetic Algorithm for the Traveling Salesman Problem using Sequential Constructive Crossover Operator”