Novel Crossover Operator for Genetic Algorithm for Permutation Problems

Rakesh Kumar, Girdhar Gopal, Rajesh Kumar

Abstract*— Simple Symmetric Traveling Salesman Problem (TSP) has a combinational nature. When there are 25 or more cities to visit, brute force search is not feasible. Instead, heuristic & probabilistic search methods are more reasonable for obtaining optimal solutions. In this paper, Genetic algorithm and crossover are researched and a novel crossover operator has been introduced by combining two existing crossover methods named PMX and OX crossover. The proposed operator is tested on 4 different inputs from TSPLIB provided by Heidelberg University and the result are compared with Partial Matched Crossover(PMX), Order Crossover(OX) and cyclic crossover(CX) and is found that proposed crossover has outperformed the rest in all the problems..

Index Terms* — Crossover, Genetic Algorithm, Traveling Salesman Problem (TSP).

I. INTRODUCTION

One of the famous computational problems is Traveling Salesman Problem (TSP). It belongs to NP-complete class of problems. A basic explanation of TSP is as follows: A salesman with a map, including N cities and the distances between each pair of cities, aims to visit each city exactly once starting from a given city. Meanwhile, he has to find the shortest cycling path between these cities to complete his tour within a minimum time period. The problem ends up with N! Different possible cycles; therefore, the brute force algorithms are not feasible. One possible solution that is proposed is to use intelligent algorithms. In past many evolutionary algorithms are used to solve TSP like Genetic Algorithm. GA is a population based search consisting of five operators: Initialization, Selection, Crossover, Mutation and Replacement [1]. Initialization used to seed the initial population randomly. Selection is used to select the fittest from the population. Crossover is used to explore the search space. Mutation is used to remove the problems like genetic drift. Replacement is used to progress generation wise population. In past, a number of crossover operators are used to solve TSP problem. These are discussed in Section II with their merits and demerits. A novel crossover operator is proposed and defined in Section III. Comparison of existing crossover operators with proposed crossover is carried out in Section IV.

Manuscript received on May, 2013.

Rakesh Kumar, DCSA, Kurukshetra University Kurukshetra, Haryana, India.

Girdhar Gopal, DCSA, Kurukshetra University Kurukshetra, Haryana, India.

Rajesh Kumar, DCSA, Kurukshetra University Kurukshetra, Haryana, India.

II. RELATED WORK

Crossover operators are the backbone of the genetic algorithm. Reproduction makes clones of good strings but does not create new ones. Crossover operators are applied to mating pool with hope that it creates a better offspring.

Partially Matched or Mapped Crossover (PMX) is the most widely used crossover operator for chromosomes having permutation encoding. It was proposed by Goldberg and Lingle for Traveling Salesman Problem, [2]. This crossover builds an offspring by choosing a subsequence of tour from one parent and preserving the order and position of as many cities as possible from the other parent. The subsequence is selected by choosing two random cut points, which serve as boundaries for the swapping operations, [1, 3 & 4].

Order Crossover (OX1) is also used for chromosomes with permutation encoding and was proposed by Davis [5]. This crossover builds an offspring by choosing a subsequence of tour from one parent and preserving the relative order of cities from the other parent. It copies the subsequence of permutation elements between the crossover points from the cut string directly to the offspring, placing them in the same absolute position [3].

Order Based Crossover (OX2) selects randomly several positions is a parent tour. And the orders of the selected cities in this parent are imposed to the other parent. So that the offspring is equal to the parent 1 except the empty cities, and then remaining cities are filled from parent 2 in the same order in which they appear, [6].

Cycle crossover is used for chromosomes with permutation encoding. Cycle crossover performs recombination under the constraint that each gene comes from the parent or the other [7]. The basic principle behind cycle crossover is that each allele comes from one parent together with its position. It divides the elements into cycles. A cycle is a subset of elements that has the property that each element always occurs paired with another element of the same cycle when the two parents are aligned. Cycle Crossover occurs by picking some cycles from one parent and the remaining cycles from the alternate parent. All the elements in the offspring occupy the same positions in one of the two parents. First a cycle of alleles from parent 1 is created. Then the alleles of the cycle are put in child 1. Other cycle is taken from parent 2 and the process is repeated [1 & 8].

Position Based Crossover (POS) also starts by selecting a random set of positions in the parent tours. However, it imposes the position of the selected cities on the corresponding cities of the other parent, [6].

Heuristic Crossover is a crossover which emphasizes edges. These create offspring's by first select a random city to be the current city. Then edges incident to current city are choose and some probability distribution is

defined on new edges based on their costs. And then edges are selected on this distribution. If uniform probability distribution is chosen, the offspring inherits about 30% of the edges of every parent, and about 40% of the edges are randomly selected, [9].

Edge crossover starts from creating Edge list for each Vertex V. Then recursively choosing a vertex with minimum edge set members and adding it to current city, and then delete it from all other edge sets, until all vertices are traversed once, [10]. It tends to inherit parents' edges more and introduce new edges with less probability 1-5%. It has a Computation complexity: ***O* (*n***).

Sorted Match Crossover was proposed by Brady in 1985, [11]. It searches for sub tours in both the parents which have the same length, and starts and end to the same city, and also contain same set of cities. If such sub tours are there the costs of these are determined. The offspring is constructed from the parent who contains the sub tour with the highest cost by substituting the sub tour for the sub tour with the lowest cost.

Maximal Preservative Crossover (MPX) was introduced by Muhelbein in 1988. It works similar to PMX Crossover. It first selects a random sub string of the first parent whose length is greater than or equal to 10 (except for small problem instances) and smaller than or equal to the problem size divided by 2. These restrictions assure that enough information is there to exchange between the parent strings without loosing too much information from any of the both parents. All the elements of chosen sub string are removed from the second parent. The sub string chosen from parent1 is copied into the first part of the offspring. Finally the end of the offspring is filled up with cities in the same order as they appear in the second parent, [11].

Voting Recombination (VR) is a p-sexual crossover operator. Where p is a natural number greater than or equal to 2. It does not originate from biology, [12]. A threshold is defined, which is a natural number smaller than or equal to p. then for every j from 1 to n. the set of j^{th} elements of all the parents is considered. If in this set an element occurs at least the threshold number of times, it is copied into the offspring. The remaining positions are then filled with mutations.

Alternating position crossover (AP) simply creates an offspring by selecting alternately the next element of the first parent and the next element of the second parent, omitting the elements already present in the offspring, [13].

III. PROPOSED CROSSOVER

The proposed method tries to avoid the disadvantages of above crossover techniques. The main idea of proposed crossover is to combine two crossovers to form a new one. By doing so, it is expected to get better results than using them individually. Mixing PMX and OX together to get two new genomes and appending them to population set will result better than simple technique. PMX has a time complexity of O (n) where n is the number of cities, because of the repairing procedure. OX has a less time complexity O(m) where m is the difference length of swath segment plus a constant, as it does not need a repairing procedure, it just fill the blank elements in a sliding motion from left to right.

The reason why expected results are better can be summarized as: PMX does things point-by-point, whereas OX applies sliding motion to left holes and take less time to fill them. So using both together can overcome the individual discrepancies and will result in an operator which works with both functionalities.

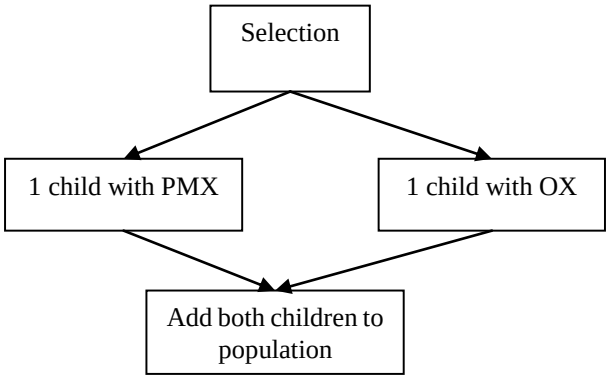

Figure 1: Summary of proposed crossover

For example, if two parents are selected as below for crossover:

Parent1: 4 3 6 2 5 1 9 7 8

Parent2: 6 4 7 1 5 2 9 8 3

And let crossover sites are chosen as 2 and 6. PMX produces following children:

Child1: 4 3 7 1 5 2 9 6 8

Child2: 7 4 6 2 5 1 9 8 3

While OX produces following children:

Child1: 3 6 7 1 5 2 9 8 4

Child2: 4 7 6 2 5 1 9 8 3

Whereas proposed crossover produces following children:

Child1: 4 3 7 1 5 2 9 6 8

Child2: 3 6 7 1 5 2 9 8 4

IV. COMPUTATIONAL EXPERIMENTS AND RESULTS

A. Experimental Setup

In this paper, 4 different inputs for benchmark TSP instances are used for testing. All experiments are coded in MATLAB R2011a and problem instances are taken from TSPLIB, which can be found at http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.

The following parameters are used in this implementation:

- Population size (N): 50 and 100
- Number of generations (ngen) : 200, 500 and 1000
- Selection method: Roulette Wheel Selection (RWS)
- Crossover: Partial Matched Crossover (PMX), Order Crossover (OX), Cycle Crossover (CX), Proposed Mixed Crossover (Prop MC) with pc=0.7.
- Mutation: Inversion with mutation probability 0.01
- Algorithm ending criteria: Execution stops on reaching ngen generations.
- Fitness Function: Objective value of function (Minimum tour length)

B. Experimental Results

Problems instances and results are recorded in following tables and figures:

TABLE - I: Comparison of Crossover operators (Population Size = 50)

Crossovers		PMX (COSTS)	OX (COSTS)	CX (COSTS)	Prop MC (COSTS)	Figure Number
Gen = 200	Eil51	1077	1091	1229	822	Figure 2
	Eil76	1024	1063	1161	875	Figure 3
	Eil101	2588	2540	2839	1712	Figure 4
	A280	2.858e+004	2.733e+004	3.075e+004	1.819e+004	Figure 5
Gen = 500	Eil51	1110	1020	1280	726	Figure 6
	Eil76	1864	1673	2116	1417	Figure 7
	Eil101	2639	2325	2871	1948	Figure 8
	A280	2.625e+004	2.659e+004	3.036e+004	2.125e+004	Figure 9
Gen = 1000	Eil51	1015	995	1190	880	Figure 10
	Eil76	1731	1566	2054	1149	Figure 11
	Eil101	2302	2220	2695	1737	Figure 12
	A280	2.612e+004	2.488e+004	2.969e+004	1.987e+004	Figure 13

TABLE-I is used to compare the results obtained by different crossover operators on 4 different TSP benchmark instances, Eil51 (which have 51 cities), Eil76 (which have 76 cities), Eil101(which have 101 cities) and a280(which have 280 cities) with population size=50 and different generation numbers like 200, 500 and 1000. It can be observed through results that proposed crossover has outperforms all other crossovers in majority cases.

TABLE - II: Comparison of Crossover operators (Population Size = 100)

Crossovers		PMX (COSTS)	OX (COSTS)	CX (COSTS)	Prop MC (COSTS)	Figure Number
Gen = 200	Eil51	1183	1044	1192	729	Figure 14
	Eil76	1789	1878	2074	1312	Figure 15
	Eil101	2569	2579	2781	1721	Figure 16
	A280	2.901e+004	2.823e+004	2.956e+004	1.868e+004	Figure 17
Gen = 500	Eil51	1006	980	1182	763	Figure 18
	Eil76	1568	1665	1967	1285	Figure 19
	Eil101	2418	2467	2819	1545	Figure 20
	A280	2.574e+004	2.709e+004	2.947e+004	1.553e+004	Figure 21
Gen = 1000	Eil51	928	1023	1290	702	Figure 22
	Eil76	1401	1610	1978	1107	Figure 23
	Eil101	2261	2367	2615	1548	Figure 24
	A280	2.539e+004	2.593e+004	2.93e+004	1.709e+004	Figure 25

TABLE-II is used to compare the results obtained by different crossover operators on 4 different TSP benchmark instances, Eil51 (which have 51 cities), Eil76 (which have 76 cities), Eil101(which have 101 cities) and a280(which have 280 cities) with population size=100 and different generation numbers like 200, 500 and 1000. It can be observed through results that proposed crossover has outperforms all other crossovers in majority cases

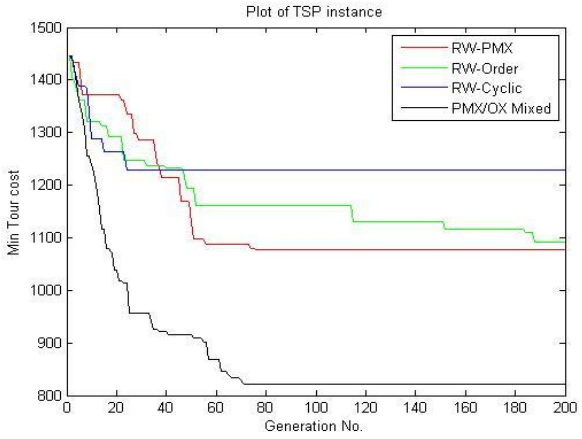

Figure 2

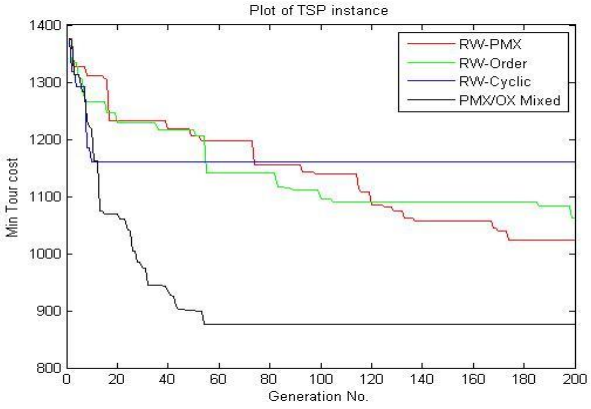

Figure 3

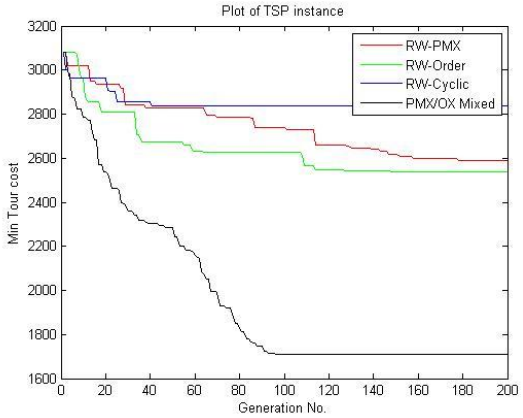

Figure 4

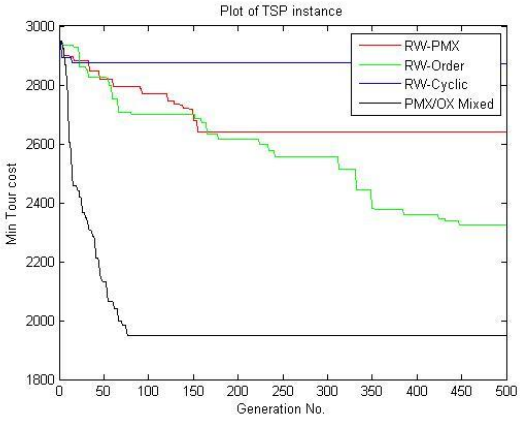

Figure 8

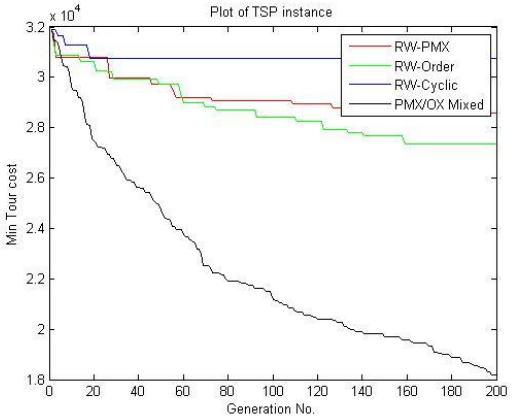

Figure 5

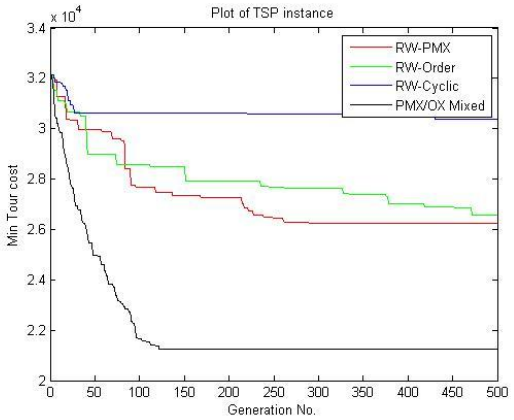

Figure 9

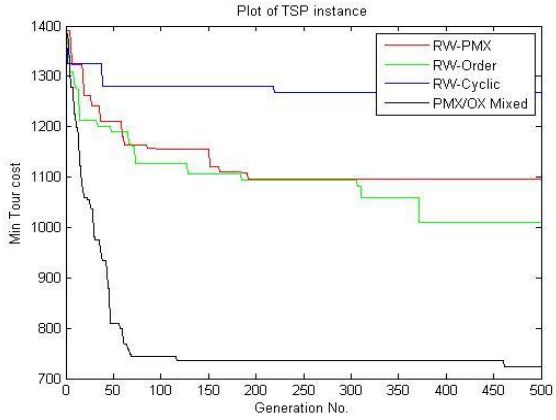

Figure 6

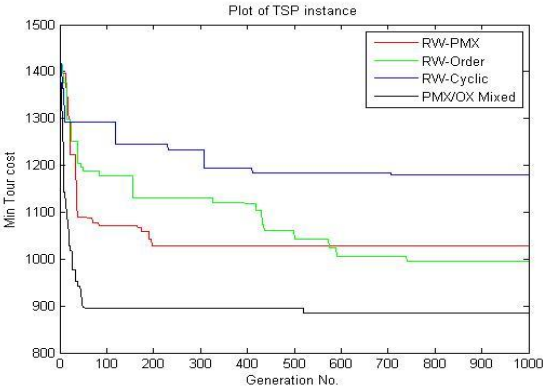

Figure 10

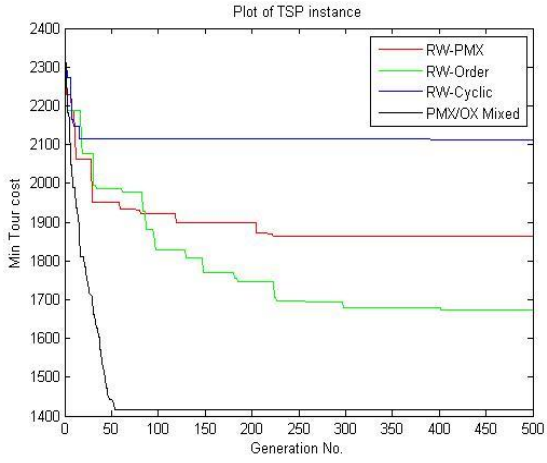

Figure 7

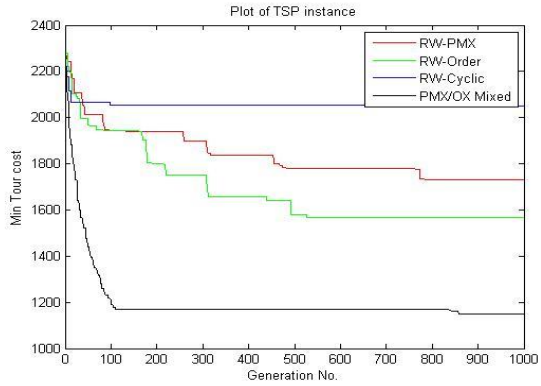

Figure 11

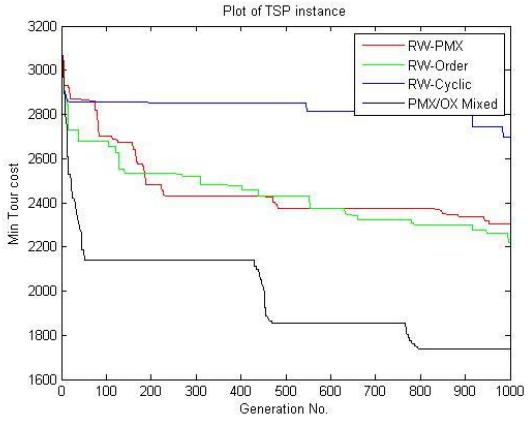

Figure 12

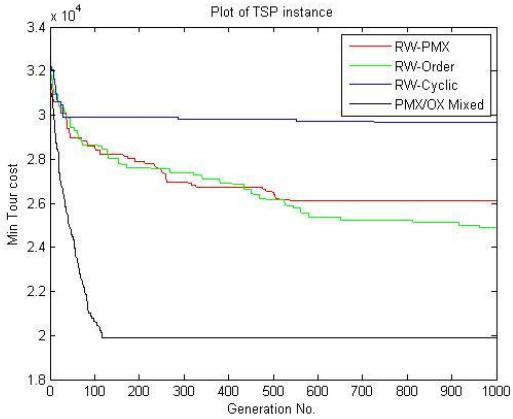

Figure 13

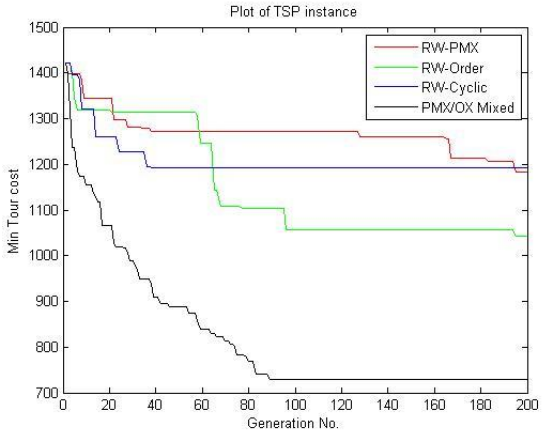

Figure 14

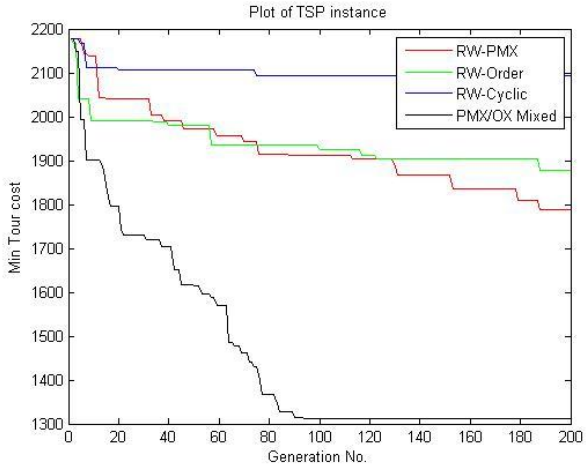

Figure 15

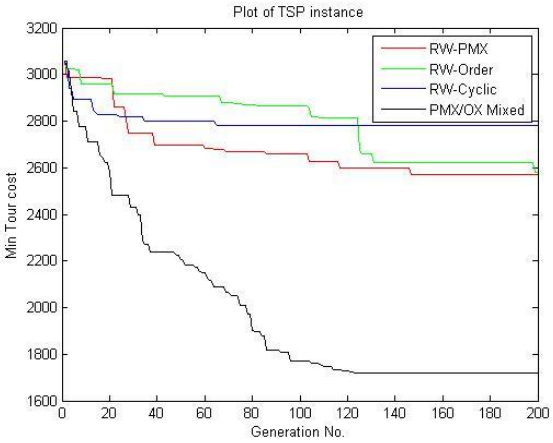

Figure 16

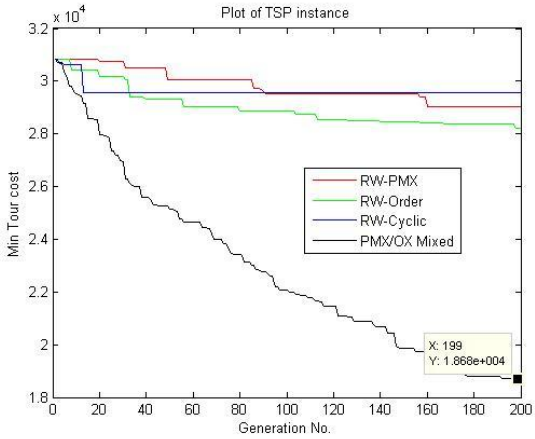

Figure 17

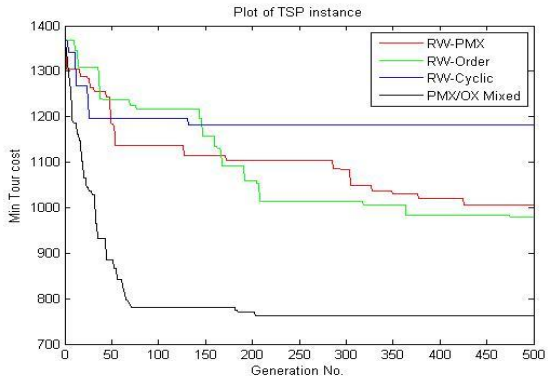

Figure 18

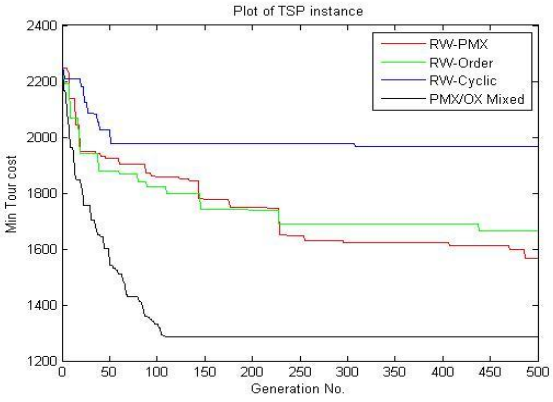

Figure 19

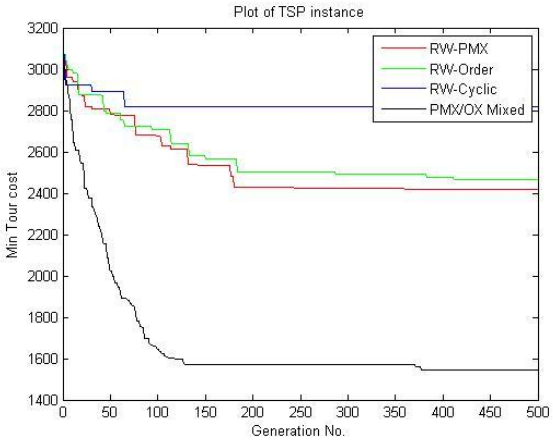

Figure 20

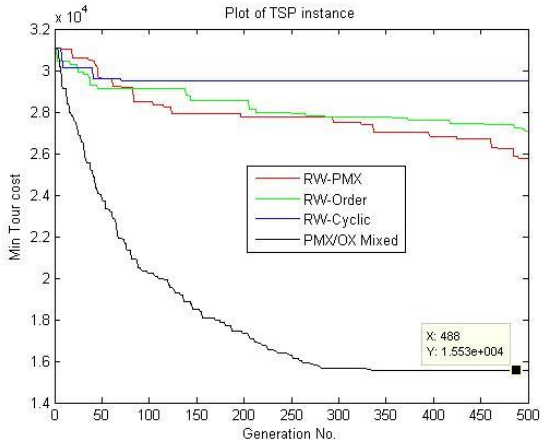

Figure 21

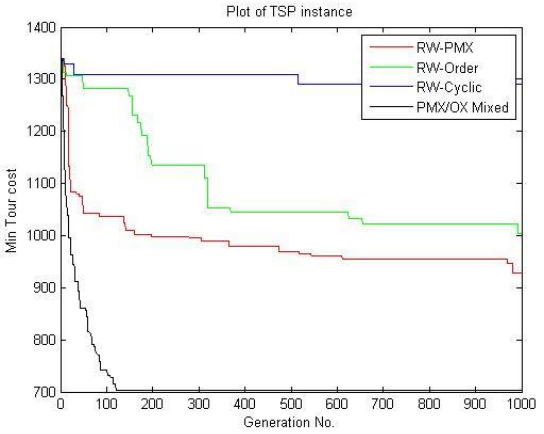

Figure 22

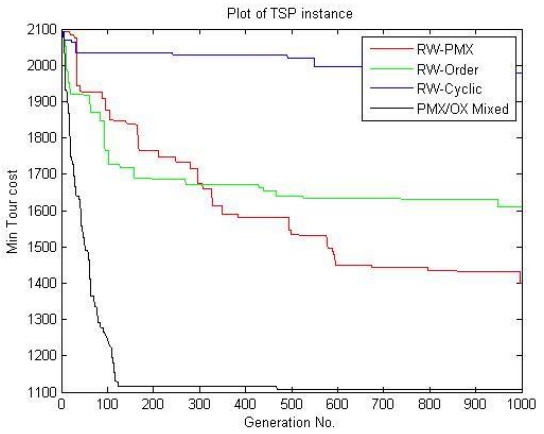

Figure 23

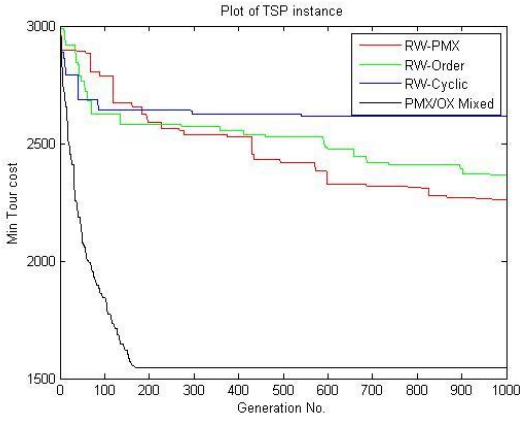

Figure 24

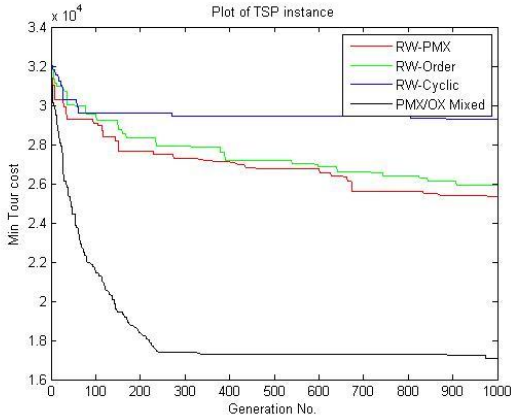

Figure 25

V. CONCLUSION

This paper compares proposed crossover operator with existing ones on benchmark TSP problems like eil51, eil76, eil101 and a280. It was found that proposed crossover yields better results than existing crossovers. Proposed crossover has features of PMX and OX both, so helpful in improving the solution quality. Also improve the performance of genetic algorithm in terms of convergence and number of iterations. Proposed crossover can be tested and implemented in different combination of selection and mutation in future to substantiate its performance. Hybridization of crossover has increased the existing technique of genetic algorithms and amplified the search performance of the algorithm.

REFERENCES

[1] Goldberg D.E. "*Genetic algorithms in search, optimization, and machine learning*", Addison Wesley Longman, Inc., ISBN 0-201-15767-5, 1989.

[2] Goldberg D.E. and Lingle R. "*Alleles, loci and the travelling salesman problem,*" Proceedings of an International Conference on Genetic Algorithms, Morgan Kauffman, 1985, pp 10-19.

[3] Sivanandam S.N. and Deepa S. N. "*Introduction to Genetic Algorithms*", Springer, ISBN 9783540731894, 2007.

[4] Eiben A.E. and Smith J.E. "*Introduction to Evolutionary Computing*", Springer, Heilderberg, Germany, 2003.

[5] Davis L, "*Handbook of Genetic Algorithms*", New York, Van Nostrand Reinhold, 1991.

[6] Syswerda G., "*Schedule optimization using genetic Algorithms",* in Davis L. Ed. Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 2008, pp.-332-349.

[7] Oliver I.M., Smith D.J. and Holland J.H. "*A study of permutation crossover operators on the travelling salesman problem*", In Proceedings of the Third International Conference on Genetic Algorithms, London, Lawrence Eribaum Associates, 1987.
[8] Kumar Rakesh, Jyotishree, 2012, "*Novel Knowledge Based Tabu Crossover In Genetic Algorithms*", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 8, August 2012 ISSN: 2277 128X.
[9] Grefenstette, J. J., "*Incorporating Problem Specific Knowledge into Genetic Algorithms*". In L. Davis, ed., *Genetic Algorithms and Simulated Annealing*, 42-60, Los Altos, CA,Morgan Kaufmann, 1987.
[10] Whitley Darrell, "*Functions as Permutations: Regarding No Free Lunch*", Walsh Analysis and Summary Statistics. PPSN, 2000, 169-178
[11] Muhlenbein H. Gorges-Schleuter M. Kramer O. "*Evolution algorithms in Combinatorial Optimization",* Parallel Computing 7, 1988, pp. 65-85
[12] Muhlenbein H. and Kinderman J. "*the Dynamics of Evolution and learning – Towards Genetic Neural Networks",* in J. Pfeiffer (Ed.), Connectionism in Perspectives, 1989.
[13] Larranaga P., Kuijpers C.M.H., Poza M., y Murga R.H., "*Decomposing Baysian Networks: Triangulation of the Moral Graph with genetic Algorithms"* Statistics and Computing, 1996.

Dr. Rakesh Kumar obtained his B.Sc. Degree, Master's degree – Gold Medalist (Master of Computer Applications) and PhD (Computer Science & Applications) from Kurukshetra University, Kurukshetra. Currently, He is Professor in the Department of Computer Science and Applications, Kurukshetra, University, Kurukshetra, Haryana, India. His research interests are in Genetic Algorithm, Software Testing, Artificial Intelligence, and Networking. He is a senior member of International Association of Computer Science and Information Technology (IACSIT).

Mr. Girdhar Gopal obtained his B.Com Degree, Master's degree – (Master of Computer Science (S/W)) from Kurukshetra University, Kurukshetra. He had qualified GATE exam two times and also UGC-NET Exam two times. Currently, He is Research Scholar in the Department of Computer Science and Applications, Kurukshetra, University, Kurukshetra, Haryana, India. His research interests are in Genetic Algorithm, Software Testing and Design of Algorithms.

Mr. Rajesh Kumar is working as an Assistant Professor in the Department of Computer Science and Applications, Kurukshetra, University, Kurukshetra, Haryana, India. He has approximately four years teaching experience. He obtained his B.Sc. Degree, Master's degree (Master of Computer Applications) from Kurukshetra University, Kurukshetra and M.Tech (IT) from Karnataka State Open University, Mysore. He had qualified UGC-NET exam. His research interests are Theory of computation, Linux Administration, Analysis of Design and Algorithm, Networking.