

A Genetic Method using Hybrid Crossover for Solving Travelling Salesman Problem



Anubhav Kumar Prasad, Dharm Raj Singh, Pankaj

Abstract: This paper proposes a Genetic approach using Hybrid Crossover for Solving the Travelling Salesman Problem. Proposed hybrid method generates an initial population using Nearest Neighbor (NN) approach which is modified using “Sub-Path Mutation” (SPM) process. Modified population undergoes Distance Preserving Crossover (DPX) [2] and 2-opt Optimal mutation (2-opt) [1] to check for possible refinement. SPM searches position for the minimum distant city within a given path. This work is motivated by the algorithm developed by [3] who performed DPX and 2-opt mutation on the initial population generated using NN. For performance comparison, standard TSPLIB data is taken. The proposed hybrid method performances better in terms of % best error. It performs better than methods reported in [3 - 11].

Index Terms: Nearest Neighbor approach, Distance Preserving Crossover, Sub-Path Mutation, 2-opt mutation.

I. INTRODUCTION

A. Travelling Salesman Problem

For given cities with distance/cost between each pair of the city, the objective is to visit each city such that overall travelling cost is optimal. Travelling cost incurred for visiting a city may not be minimized, but the overall cost needs to be minimum known as optimal travelling cost. Travelling salesman problem holds if each city is visited exactly once or end the city and start a city are the same. The problem can be represented using graph theory, with vertices representing cities and edges representing the distance between joining cities. Travelling salesman problem can be better represented using tabular structure (cost matrix) as shown in Table 1.1 with diagonal elements as zero. In the graph-theoretic framework, finding Hamiltonian cycle is equivalent of Travelling Salesman Problem. This paper consists of five sections; Section II discusses problem formulation, followed by problem types in Section III. Section IV discusses related work describing NN, DPX and

2-opt. Section V presents the proposed hybrid method. Section VI discusses algorithm for NN, DPX, 2-opt and proposed hybrid method. Section VII presents the experimental setup used for the proposed hybrid method. Finally, Section VIII presents the conclusion of this paper.

Table 1.1: Travelling salesman problem with n cities with cost of travelling within same city as 0.

city	1	2	...	n
1	0	c_{12}	...	c_{1n}
2	c_{21}	0	...	c_{2n}
\vdots
n	c_{n1}	c_{n2}	...	0

B. Literature Review

Travelling salesman problem (TSP) belongs to network optimization problem and come under NP-complete problems in combinatorial optimization [12, 13]. The origin of TSP is unclear; however, it was mathematically formalized by Irish mathematician Hamilton and Thomas Kirkman. A detailed discussion of there works can be found in “Graph Theory 1736-1936” book by Biggs, Lloyd and Wilson [14]. Hamilton’s Icosian Game [15] was based on Hamiltonian Cycle [16]. It seems that it was Karl Menger [17] who first studied general form of TSP. Alexander Schrijver [18] explored the connection between works of Menger and Whitney along with the growth of TSP. The problem was considered mathematically by Merrill Flood [19] for the first time during the 1930s. It was Hassler Whitney [20] who coined the name Travelling Salesman Problem. TSP became increasingly popular during the 1950s and 1960s in scientific circles of Europe and the USA after prizes was offered by RAND Corporation for steps in solving the problem. Worthy contributions were made by Dantzig, Fulkerson and Johnson [21] of RAND Corporation, as they developed the cutting plane method [22] for its solution. The branch and bound algorithm [23] is believed to be used by them for getting the optimal solution for TSP. Karp [24] proved a Hamiltonian cycle to be NP-complete. Christofides [25] made a remarkable work as he presented a worst-case analysis of heuristic value to optimal value and proved the ratio to be less than 3/2. Lenstra and Kan [26] discussed some applications of TSP. In 1987, Padberg and Rinaldi [27] solved 532 city TSP using branch and bound algorithm. M.

Revised Manuscript Received on 30 July 2019.

* Correspondence Author

Anubhav Kumar Prasad*, Research Scholar, Department of Science and Technology- Centre for Interdisciplinary Mathematical Sciences, Institute of Science Banaras Hindu University, Varanasi, Uttar Pradesh, India, 221005.

(*Corresponding Author:*) **Dr. Dharm Raj Singh**, Assistant Professor and Head of Department, Computer Application, Jagatpur Degree College, Varanasi, Uttar Pradesh, India, 221005.

Dr. Pankaj, Assistant Professor, Mathematics Section, Mahila Maha Vidyalyaya, Banaras Hindu University, Varanasi, Uttar Pradesh, India ,221005.

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an [open access](https://creativecommons.org/licenses/by-nc-nd/4.0/) article under the CC-BY-NC-ND license <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

A Genetic Method using Hybrid Crossover for Solving Travelling Salesman Problem

Padberg and Rinaldi [28] proposed Branch and cut approach to solving TSP for 2392 cities. M. Grotschel and O. Holland [29] used a cutting plane algorithm to solve TSP for up to 1000 cities. Reinelt [30] provided TSPLIB - TSP Library which is the main source to collect problem instances. In last few decades, many heuristic algorithms merged aiming to provide near-by optimal solution at faster rate covering large number of cities.

Some notable algorithms include: ant colony optimization [31-36], neural network [37], self-organizing maps [38-40], particle swarm optimization techniques [41, 42], simulated annealing [43, 44] weed optimization [45, 46], genetic algorithm [47, 48].

II. PROBLEM FORMULATION

Let the travelling salesman problem consist of n cities with c_{uv} being the cost involved in travelling from city u to city v . The Travelling salesman problem can be written as the following integer lpp (linear programming problem):

$$Z = \sum_{u=1}^n \sum_{v=1, v \neq u}^n c_{uv} x_{uv};$$

$$\sum_{u=1, u \neq v}^n x_{uv} = 1, v = 1, \dots, n;$$

$$\sum_{v=1, v \neq u}^n x_{uv} = 1, u = 1, \dots, n;$$

$$\sum_{u \in Q} \sum_{v \in Q} x_{uv} \leq |Q| - 1, \forall Q \subseteq \{2, \dots, n\}.$$

III. PROBLEM TYPES

Travelling Salesman Problem is confined with visiting a given number of cities such that every intermediate city is visited exactly with start city being the end city. However, it's not the case that the cost of travelling city A to city B is the same. There may be cost differences due to toll tax differences, airfare differences, etc. Hence, the problem can be Symmetric or Asymmetric.

A. Symmetric TSP

If the cost of visiting city A to city B is equal to that of visiting city A from city B, the problem forms an undirected graph representing symmetric TSP. This symmetry halves the number of possible solutions.

B. Asymmetric TSP

There is a possibility that a direct path exists from city A to be but not from city B to city A or travelling costs are different. This is the same as a directed graph with both edges representing different costs. Examples include toll tax differences, one-way streets, and different airfares with different departure and arrival fees.

IV. FRAMEWORK OF PROPOSED ALGORITHM

This section will discuss standard methods which motivated to develop the proposed heuristic algorithm.

A. Nearest Neighbor Algorithm (NN)

NN is one of the simplest algorithms based on a greedy approach to solving the Travelling Salesman Problem. This is similar to spanning tree formation approaches of Prim and Kruskal. It starts with selection of a random city as a starting point and searches for the minimum distant city until all cities have been visited. The algorithm generally misses a good heuristic solution due to its greedy nature. This forces to generate multiple tours by changing the starting city. For example, consider Travelling Salesman Problem with 6 cities given in Table 1.2, tour cost by taking city 1 as starting city is 21 (1423561), whereas its 20 (2351462) if starting city is 2. As an informal rule, if the last few stages of the tour are not comparable in length to the first stages, then there might exist a better tour. Section VI.A discusses the algorithm of NN.

Table 1.2: [49] Travelling salesman problem with 6 cities with the cost of travelling within the same city as 0.

City	1	2	3	4	5	6
1	0	3	4	2	8	3
2	5	0	3	4	4	5
3	4	1	0	5	3	4
4	4	2	6	0	4	5
5	3	3	3	5	0	4
6	7	4	5	6	7	0

B. Distance Preserving Crossover (DPX)

Distance Preserving Crossover (DPX) was introduced by Merz and Freisleben for producing offspring for the Quadratic Assignment Problem. The distance d between two paths (s_1 and s_2) is defined as a number of cities that are not at the same position in both paths. In Figure 1.1, the distance is 8, because out of 10 cities only city 4 and 8 share a common position. This helps to check whether the distance is preserved in offspring or not with respect to both parents and within offspring. DPX selects two parents randomly to generate two offspring preserving positions that are common in both parents. Remaining positions are filled by the following condition:

$$\text{if}(s_1(v)) = s_2(u) \\ c_1(u) = s_2(v) \\ \text{if}(s_2(v)) = s_1(u) \\ c_2(u) = s_1(v)$$

where u and v vary from 1 to n (number of cities), and c_1, c_2, s_1 and s_2 represent two offspring and two parents respectively.

This property is the same as transitive property which makes the transition from A to C if $A \rightarrow B$ and $B \rightarrow C$. Figure 1.1 shows the process for DPX with arrows indicating position number for cities. Section VI.B discusses the algorithm of DPX.

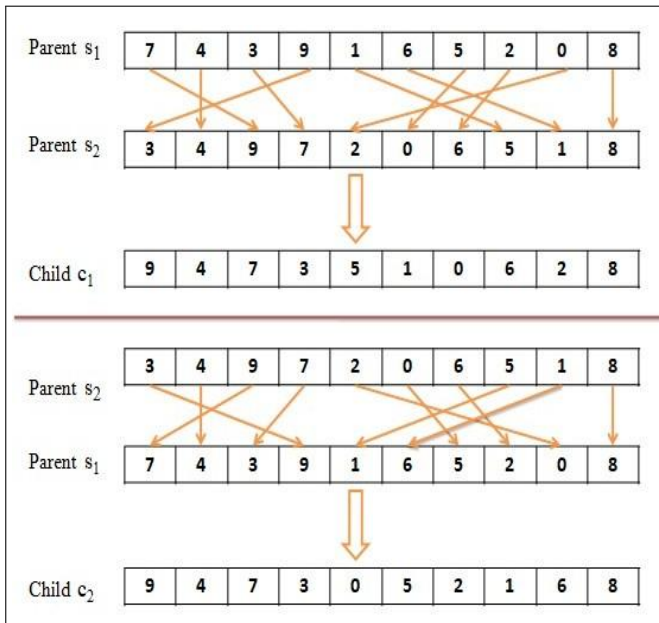


Figure 1.1: DPX crossover.

C. 2-opt mutation (2-opt)

Croes's introduced the 2-Opt method and works by interchanging connections/edges in a given path such that two edges from the path is replaced by two new edges that are not in solution to improve path cost. This is also known as inversion and can be seen from Figure 1.2, where Inversion leads to 2-opt mutation. Here inversion is involved for two connecting cities to check for possible improvement in the solution. Interchanging of connections continue until no more improvements are possible. The 2-opt method was generalized in the form of k-opt mutation that works by removing k-edges and adding k-new other edges that are not in solution. Section VI.C discusses the algorithm of 2-opt.

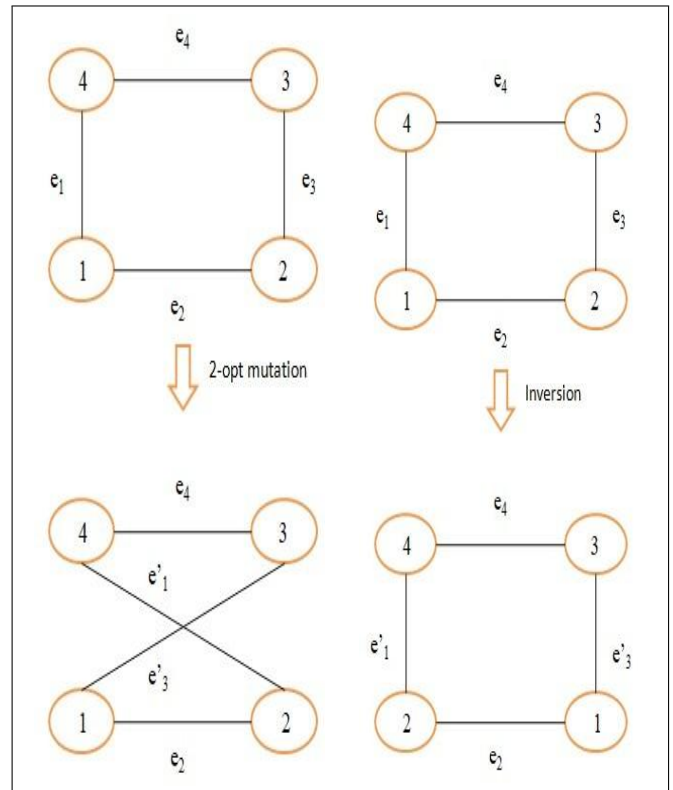


Figure 1.2: 2-opt mutation vs Inversion.

V. PROPOSED HYBRID METHOD

The efficiency of hybrid heuristic method depends on the quality of the initial solution and approach to refine it. In other words, solution quality depends on the use of search strategy: exploration and exploitation. The proposed hybrid algorithm is motivated from the work of Singh et al. who used a hybrid approach by generating the initial population using NN algorithm and refined the population using DPX crossover and 2-opt mutation. The proposed hybrid method makes two modifications: it refines initial population generated using NN by SPM process and changing crossover probability (pc) from 0.8 to 0.5 and mutation probability (pm) from 0.2 to 0.1. Figure 1.3 presents the working of the proposed algorithm. Section VI.D discusses the algorithm of the proposed hybrid method.

A. Sub-Path Mutation (SPM)

Sub-Path Mutation process searches the best position for sub-path containing the minimum distant city. This process is repeated for randomly generated paths using the NN algorithm. This approach helps in improving path cost which undergoes DPX and 2-opt mutation. Another approach could search for inserting sub-path between maximum distant city to improve path cost, but this process can become complex, a number of cities in sub-path may vary depending upon path construction. We used SPM which require only 2 cities in sub-path which reduces the complexity.

A Genetic Method using Hybrid Crossover for Solving Travelling Salesman Problem

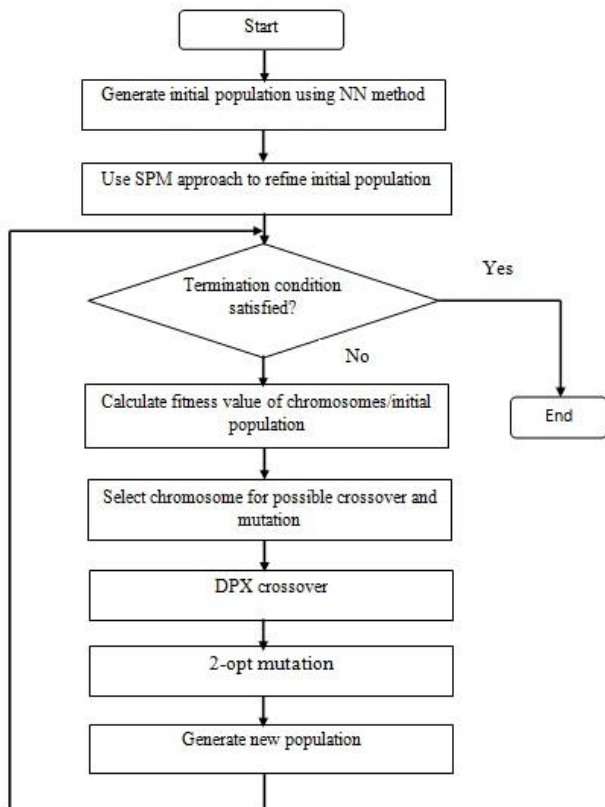


Figure 1.3: Proposed Hybrid Crossover flowchart.

VI. ALGORITHM

A. Nearest Neighbor (NN)

- Step 1:** Select start city sc randomly and marks it as current city cc .
Step 2: Select unvisited city uc which is at the minimum possible distance from cc .
Step 3: Mark city uc as current city cc .
Step 4: Repeat step 2 and 3 until all cities are visited.
Step 5: Connect last visited city with sc .
Step 6: Calculate the total path cost.

The algorithm of NN is simple, as it tries to connect different cities which are at a minimum possible distance. The process starts by randomly selecting start city sc , which is marked as the current city cc . After start the city is selected, the next step is to select the minimum distant city from current city cc that is not visited yet. Step 2 depicts this process. After finding such city, we mark it as current city cc which is the third step of the algorithm. After this, the process repeats step 2 and 3 until all cities are visited once. Finally, the last city is connected with start city sc and total path cost is calculated.

As discussed earlier, more paths need to be formed because of the greedy nature. Although it doesn't guarantee that constructing different paths will provide good heuristic solution always.

B. Distance Preserving Crossover (DPX)

- Step 1:** Generate initial population using any TSP method.
Step 2: Select two parent chromosome (s_1 and s_2) to create two offspring (c_1 and c_2).

Step 3: Copy city number to same position in both offspring satisfying following condition:

$$s_1(u) == s_2(u)$$

where $u = 1$ to n (number of city).

Step 4: Fill remaining positions of c_1 using following condition:

$$\text{if}(s_1(v) == s_2(u)) \\ c_1(u) = s_2(v)$$

where $u, v = 1$ to n (number of city).

Step 5: Fill remaining positions of c_2 using following condition:

$$\text{if}(s_1(u) == s_2(v)) \\ c_2(u) = s_1(v)$$

where $u, v = 1$ to n (number of city).

Step 6: Calculate path cost for both offspring.

Step 7: Replace s_1 and s_2 with offspring in case path cost of offspring is less than the parent.

Being a genetic operator, DPX needs two parent chromosome for a crossover which is the first step of the algorithm. After generating the initial population using any TSP method, the next step is to select two parent chromosomes to create two offspring. Step 2 reflects this process. DPX can be divided into two steps: the first step copies city that is at the same position in both parents to both offspring at the same position, which is step 3 of the algorithm. After this, remaining positions are filled using conditions shown in step 4 and 5 of the algorithm. Finally DPX checks for improvement in path cost and if found replaces parent with offspring having higher fitness value. DPX may be applied for each possible pair of cities or may work on the best chromosome with other chromosomes of population.

C. 2-opt mutation (2-opt)

- Step 1:** Generate initial population using any TSP method.
Step 2: Select single parent chromosome s_1 to create offspring c_1 .
Step 3: Select four cities A, B, C and D from s_1 such that there is a connection between A, B and C, D.
Step 4: Connect A to C and B to D by breaking the connection between A, B and C, D.
Step 5: Calculate the new path cost. If lesser, accept this change, else discard it.
Step 6: Repeat steps 3 to 5 for all possible combination of 4 cities.

As with DPX crossover, 2-opt needs an initial solution to perform mutation. Since, its a mutation operator, therefore it needs a single parent chromosome. 2-opt mutation works on the principle of inversion, it selects 4 cities with the connection between even and odd ones which is shown as step 3 of the algorithm. After selecting 4 cities, it creates two new connections between odd and even cities and breaks old connections discussed earlier. Step 4 depicts this process. Finally, it checks whether the improvement is made in path cost or not, if the improvement is not made, it rejects the mutation else accepts it. Steps 3 to 5 are repeated again and again with each possible combination so as to reduce path cost. As discussed earlier, newer connections may be reversed to maintain a predecessor-successor relationship.

D. Proposed Hybrid Method

Step 1: Generate initial population P using NN method.
 Step 2: Use SPM approach to improve the initial population.
 Step 3: Calculate the fitness value of improved population P’.
 Step 4: Select 20 chromosomes from P’ and select best 2 using tournament selection for possible crossover and mutation.
Step 5: Apply DPX crossover on the selected chromosome with crossover probability *pc*.
Step 6: Apply 2-opt mutation on selected chromosomes with mutation probability *pm*.
Step 7: Replace the selected chromosome with offspring generated using DPX and 2-opt if offspring has higher fitness value.
Step 8: Repeat steps 4 to 7 until the desired number of iterations have been performed.
 The proposed hybrid method is motivated from NN+2-opt mutation method, therefore most of the working is the same for the proposed hybrid method. Step 1 generates an initial population P using NN method which is refined using the proposed SPM approach to get population P j. SPM tries to find a better position for minimum distant sub-path with two cities for each chromosome. After improving initial solution P to P ‘ which is step 2 of the algorithm, pair of 20 chromosomes are selected using tournament selection, out of which 2 chromosomes (including fittest chromosome among 20) are selected for DPX crossover with probability rate *pc*. Offspring generation is dependent on *pc* and if generated, is compared with parent chromosome leaving only fittest chromosome.

This fitter chromosome undergoes 2-opt mutation to check for further improvement with a probability rate of *pm*. The process of crossover and mutation is repeated until the desired numbers of iterations are performed.

VII. EXPERIMENTAL RESULTS

A. Experimental Setup

For evaluating the time comparison of the proposed hybrid method, results were generated on 2.20 GHz Intel Core i5 machine with 4096 MB RAM.

B. Experimental Design

For performance comparison of the proposed hybrid method with some of the standard methods, following standard

Table 1.3: Performance comparison of different methods.

Methods	% Best Error for TSP data-set													
	kroA100	berlin52	ch150	pr144	pr152	kroB150	d198	rat195	ts225	kroA200	pr299	pr226	pcb442	lin318
EXC	0.1692	0.0000	1.9761	0.2426	2.0358	2.9277	1.9709	4.4770	3.0424	2.5061	10.2198	4.9186	7.3437	10.1026
DISP	4.2947	0.0663	3.4161	1.4845	4.7216	7.8263	3.8910	5.0336	3.3693	6.7284	12.2616	4.1322	11.3435	10.9996
INV	2.6125	0.0000	2.9412	1.0284	3.1202	7.1183	2.6743	3.7021	3.1640	6.2245	10.9543	8.1512	10.5124	9.5410
INS	0.4652	0.0000	0.6434	0.1879	1.3938	1.8178	2.0279	4.2617	3.1751	2.1554	3.8804	3.8423	8.9685	6.3099

benchmark data were taken from TSPLIB: pr144, ch150, berlin52, kroA100, kroB150, pr152, kroA200, ts225, rat195, d198, pr226, pr299, lin318 and pcb442. In this experiment, we set *pc* and *pm* values to 0.5 and 0.1, respectively. The population size of 40 was randomly generated which was divided into a pair of twenty individuals using tournament selection procedure. We used Percentage Best Error (% Best Err.) for performance comparison. The Percentage Best Error is calculated as follows:

$$PD_{best} = \frac{[(\text{Best past cost from } n \text{ trail}) - (\text{optimal path cost from TSPLIB})]}{(\text{Optimal path cost from TSPLIB})} * 100.$$

C. Experimental Result

For each standard TSP data taken, we performed *n* = 10 trails. Performance comparison is shown in Table 1.3 with best results shown in bold. It can be seen from Table 1.3 and Figure 1.4 that proposed hybrid method performs better than NN+2-opt mutation method. Out of 14 data-sets taken, the proposed hybrid method performed better for 8 data-sets excluding berlin52 and kroA100, for which our method gave almost the same %Best Error when taken to 4 decimal places. Proposed hybrid method differed from the optimal solution by 2 and 5 for berlin52 and kroA100. Out of 9 standard methods taken for comparison, GSTM and NN+2-opt mutation methods gave the best result for 6 and 7 data-sets. All other methods performed well for only berlin52 except DISP method and NN+2-opt.

VIII. CONCLUSION

The proposed hybrid method is motivated from the work of Singh et al. where they used NN+2-opt mutation. We followed the same crossover and mutation but with different crossover and mutation probability. The proposed hybrid method also refines the initial population generated using the NN method using SPM. The proposed hybrid method gives better performance to the standard algorithms: EXC, DISP, INV, INS, SIM, SCM, GSM and GSTM. Out of 14 data-sets taken, the proposed hybrid method performed better for 8 data-sets excluding berlin52 and kroA100, for which our method gave almost same %Best Error when taken to 4 decimal places whereas GSTM and NN+2-opt mutation methods gave the best result for 6 and 7 data-sets with all methods except DISP and NN+2-opt methods giving the same result for berlin52.



A Genetic Method using Hybrid Crossover for Solving Travelling Salesman Problem

SIM	0.1692	0.0000	0.8425	0.2255	1.5282	2.8397	1.6984	2.3676	1.6930	1.5766	6.4390	2.5769	6.3728	6.3932
SCM	0.8176	0.0000	0.9344	0.1589	1.5146	3.2836	3.1179	4.6492	2.7945	3.9873	10.8776	3.7875	7.8597	9.1651
GSM	0.3007	0.0000	1.6544	0.2426	2.2977	3.4520	3.0482	4.3048	3.0424	4.4675	8.8191	5.0517	5.7643	9.4126
GSTM	0.0000	0.0000	0.4596	0.0000	0.7695	0.9644	0.3866	0.6027	0.2527	0.8683	1.2326	0.7242	2.0501	0.9827
NN+2-opt Mutation	0.0141	0.0318	0.7077	-0.0030	0.1886	0.0421	0.5830	1.2613	0.0055	0.5925	1.2139	0.0535	1.8177	1.0027
Proposed Hybrid Method	0.0000	0.0000	0.7077	0.0000	0.1886	0.0421	0.3866	1.2613	0.0055	0.5925	1.0375	0.0535	1.8177	1.0027

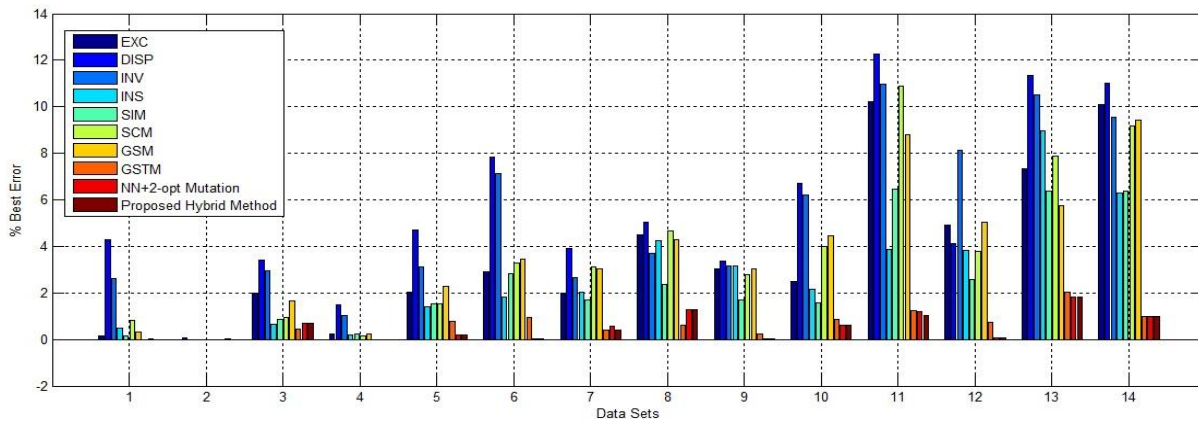


Figure 1.4: Performance comparison of different methods with numbers representing data- sets in the x-axis.

REFERENCES

1. A. G. Croes, "A method for solving traveling-salesman problems". *Operations research*, 1958, 6.6: 791-812.
2. B. Freisleben and P. Merz. "A genetic local search algorithm for solving symmetric and asymmetric traveling salesman problems." *Proceedings of IEEE international conference on evolutionary computation*. IEEE, 1996.
3. D. R. Singh, M. K. Singh, and T. Singh, "A hybrid heuristic algorithm for the Euclidean traveling salesman problem." *International Conference on Computing, Communication & Automation*. IEEE, 2015.
4. W. Banzhaf, "The "molecular" traveling salesman." *Biological Cybernetics* 64.1 1990: 7-14.
5. Z. Michalewicz, *Genetic algorithms+ data structures= evolution programs*. Springer Science & Business Media, 2013.
6. B. D. Fogel, "Applying evolutionary programming to selected traveling salesman problems." *Cybernetics and systems* 24.1 1993: 27-36.
7. D. B. Fogel, "An evolutionary approach to the traveling salesman problem". *Biological Cybernetics*, 60(2), 139-144, 1998.
8. J. Holland, "Adaptation in natural and artificial systems: an introductory analysis with application to biology." *Control and artificial intelligence* 1975.
9. G. Syswerda, "Scheduling optimization using genetic algorithms." *Handbook of genetic algorithms* 1991.
10. J. Louis, Sushil, and T. Rilun, "Interactive genetic algorithms for the traveling salesman problem." *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1*. Morgan Kaufmann Publishers Inc., 1999.
11. M. Albayrak and N. Allahverdi, "Development a new mutation operator to solve the traveling salesman problem by aid of genetic algorithms." *Expert Systems with Applications* 38.3 2011: 1313-132
12. H. C. Papadimitriou, "Combinatorial optimization." *Algorithms and Complexity* 1982.
13. G. Nemhauser, and L. Wolsey. "Integer programming and combinatorial optimization." *Wiley*, 1988. 1999.
14. L. N. Biggs, "E. Lloyd, RJ Wilson, *Graph Theory 1736-1936*." 1976.
15. A. S. Herschel, "Sir Wm. Hamilton's icosian game." *Quart. J. Pure Applied Math* 5, 1862: 305.
16. R. J. Wilson, "Introduction to graph theory". London: Longman, 107, 1972.
17. K. Menger, "Eine neue definition der bogenlänge." *Ergebnisse eines Mathematischen Kolloquiums* 2, 1932,11-12.
18. A. Schrijver, "On the history of combinatorial optimization (till 1960)." *Handbooks in operations research and management science* 12, 2005.: 1-68.
19. M. M. Flood, "The traveling-salesman problem." *Operations research* 4.1,1956.: 61-75.
20. H. Whitney, "The mathematics of physical quantities: Part I: Mathematical models for measurement." *The American Mathematical Monthly* 75.2, 1968.: 115-138.
21. G. Dantzig, R. Fulkerson, and S. Johnson. "Solution of a large-scale traveling-salesman problem." *Journal of the operations research society of America* 2.4, 1954.: 393-410.
22. G. Reinelt, "*The traveling salesman: computational solutions for TSP applications*". Springer-Verlag, 1994.
23. G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms." *European Journal of Operational Research* 59.2, 1992.: 231-247.
24. M. Karp, "Reducibility among combinatorial problems." *Complexity of computer computations*. Springer, Boston, MA, 1972. 85-103.
25. N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem (No. RR-388). Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.
26. J. K. Lenstra and A. R. Kan, "Some simple applications of the travelling salesman problem." *Journal of the Operational Research Society*, 26(4), 717-733, 1975.
27. M. Padberg, and G. Rinaldi, "Optimization of a 532-city symmetric traveling salesman problem by branch and cut." *Operations Research Letters*, 6(1), 1-7, 1987.
28. M. Padberg and G. Rinaldi, "Branch-and-cut approach to a variant of the traveling salesman problem". *Journal of Guidance, Control, and Dynamics*, 11(5), 436-440, 1988.
29. M. Grotschel and O. Holland, "Solution of large-scale symmetric travelling salesman problems. *Mathematical Programming*", 51(1-3), 141-202, 1991.
30. G. Reinelt, "TSPLIB-A traveling salesman problem library." *ORSA journal on computing*, 3(4), 376-384, 1991.
31. J. M. Cecilia, J. M. Garca, A. Nisbet, M. Amos and M. Ujaldon, "Enhancing data parallelism for ant colony optimization on GPUs". *Journal of Parallel and Distributed Computing*, 73(1), 42-51, 2013.
32. M. Mavrouniotis, and S. Yang, "Ant colony optimization with immigrants schemes for

the dynamic travelling salesman problem with traffic factors". Applied Soft Computing, 13(10), 4023-4037, 2013.

33. M. Tuba and R. Jovanovic, "Improved ACO algorithm with pheromone correction strategy for the traveling salesman problem". International Journal of Computers & Control, 8(3), 477-485, 2013.
34. H. Y. Yun, S. J. Jeong and K. S. Kim, "Advanced harmony search with ant colony optimization for solving the traveling salesman problem". Journal of Applied Mathematics, 2013.
35. T. Saenphon, S. Phimoltares and C. Lursinsap, "Combining new fast opposite gradient search with ant colony optimization for solving travelling salesman problem". Engineering Applications of Artificial Intelligence, 35, 324-334, 2014.
36. M. Mahi, O. K. Baykan, and H. Kodaz, "A new hybrid method based on particle swarm optimization Kodaz", ant colony optimization and 3-opt algorithms for traveling salesman problem. Applied Soft Computing, 30, 484-490, 2015.
37. J. C. Creput and A. Koukam, "A memetic neural network for the Euclidean traveling salesman problem". Neurocomputing, 72(4-6), 1250-1264, 2009.
38. T. A. Masutti and L. N. de Castro, "A self-organizing neural network using ideas from the immune system to solve the traveling salesman problem. Information Sciences", 179(10), 1454-1468, 2009.
39. S. Somhom, A. Modares and T. Enkawa, "A self-organising model for the travelling salesman problem". Journal of the Operational Research Society, 48(9), 919-928, 1997.
40. Y. Bai, W. Zhang and Z. Jin, "A new self-organizing maps strategy for solving the travelling salesman problem". Chaos, Solitons & Fractals, 28(4), 1082-1089, 2014
41. X. H. Shi, Y. C. Liang, H. P. Lee, C. Lu and Q. X. Wang, "Particle swarm optimization based algorithms for TSP and generalized TSP". Information processing letters, 103(5), 169-176, 2007.
42. Y. Marinakis and M. Marinaki, "A hybrid multi-swarm particle swarm optimization algorithm for the probabilistic traveling salesman problem". Computers & Operations Research, 37(3), 432-442, 2010.
43. S. M. Chen and C. Y. Chien, "Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques". Expert Systems with Applications, 38(12), 14439-14450, 2011.
44. Y. Chen and P. Zhang, "Optimized annealing of traveling salesman problem from the nth-nearest-neighbor distribution". Physica A: Statistical Mechanics and its Applications, 371(2), 627-632, 2006.
45. G. G. Roy, S. Das, P. Chakraborty, and P. N. Suganthan, "Design of non-uniform circular antenna arrays using a modified invasive weed optimization algorithm". IEEE Transactions on antennas and propagation, 59(1), 110-118, 2011.
46. A. Sengupta, T. Chakraborti, A. Konar and A. Nagar, "Energy efficient trajectory planning by a robot arm using invasive weed optimization technique". In Nature and Biologically Inspired Computing (NaBIC), Third World Congress, 311-316, 2011.
47. G. Syswerda, "Scheduling optimization using genetic algorithms. Handbook of genetic algorithms", 1991.
48. B. Freisleben and P. Merz, "New genetic local search operators for the traveling salesman problem". In International Conference on Parallel Problem Solving from Nature. Springer, Berlin, Heidelberg, 890-899, 1996.
49. H. Tucker, "Applied Combinatorics (sixth edition)". John Wiley and Sons, 1943.

AUTHORS PROFILE



Mr. Anubhav Kumar Prasad pursued Master of Computer Applications from School of Management Sciences, Varanasi, Uttar Pradesh India in 2012. He has experience of teaching at National Institute of Technology, Jamshedpur, Jharkhand, India as Adhoc faculty in Department of Computer Applications. He is currently pursuing Ph.D. from Department of Science and Technology-Centre for Interdisciplinary Mathematical Sciences, Institute of Science, BHU, India since 2014. During his research period he is engaged in taking classes of Post Graduate students in Mahila Maha Vidyalaya and DST-CIMS, Institute of Science, BHU. His main research work focuses on Optimization Techniques in Travelling and Transportation Problems, Network Security, Graph Theory and Data Structure. He is NET and GATE qualified.



Dr. Dharm Raj Singh is PhD from Department of Science and Technology-Centre for Interdisciplinary Mathematical Sciences, Institute of Science, Banaras Hindu University, Varanasi, India in 2017 and Master of Computer Applications from Uttar Pradesh Technical University Lucknow, India in 2004, which is currently known as Dr. A.P.J Abdul Kalam Technical University. He is Assistant Professor and Head of Department of Computer Application Department at Jagatpur Degree College, Jagatpur, Varanasi, Uttar Pradesh, India. His research interest work focuses on Optimization Techniques in Graph based Problems, Artificial Intelligence, Data Structure and Algorithms and Network Security. He is UGC-NET qualified in July 2016.



Dr. Pankaj pursued Master of Science (Mathematics) from Indian Institute of Technology Roorkee, India. He pursued Ph.D. from Department of Mathematics, Institute of Science, Banaras Hindu University, India and currently working as Assistant Professor in Mathematics Section, Mahila Maha Vidyalaya, BHU, India. He has vast experience of teaching at undergraduate and postgraduate levels. He is involved in supervision of Research Scholars and has many notable contributions in different areas of optimization including Convex and Inconvex optimization. His main research work focuses on Generalised Convex Optimization, Vector Variational Inequalities, Duality, Optimization in Transportation and Travelling Salesman Problems, Graph Theory. He is NET qualified.