Indian Journal of Science and Technology, *Vol 10(30), DOI: 10.17485/ijst/2017/v10i30/115512, August 2017*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Solving Travelling Salesman Problem Using Improved Genetic Algorithm

Shweta Rana and Saurabh Ranjan Srivastava

Department of Computer Science, Swami Keshvan Institute of Technology, Management and Gramothan, Ramnagaria, Jagatpura, Jaipur – 302017, Rajasthan, India; miss.shweta.rana@gmail.com, srs@skit.ac.in

Abstract

Objectives: Travelling Salesman Problem is a well known NP-Complete problem. It has many application areas in science and engineering. NP-Complete problems are most difficult problems in computer science. **Methods/Statistical Analysis:** These problems are not solvable using tradition algorithms till date. Soft computing techniques such as Genetic Algorithm (GA) can be used to solve such problems. In this paper Travelling Salesman problem has been solved using Genetic Algorithm. The proposed algorithm modifies the traditional genetic algorithm. Proposed algorithm generates some chromosome using greedy approach and adds these chromosomes in initial population. It also proposes a new greedy cross over operator for genetic algorithm. **Findings:** The implementation results of proposed algorithm prove that proposed algorithm performs better as it find out paths of less path length as compared to the optimal path known till date. **Application/Improvements:** This work is helpful in finding optimal or near optimal solutions of TSP. The algorithm can find application in all the relevant areas of science and engineering where TSP is used such as robot arm movement, drilling electronic boards etc.

Keywords: Travelling Salesman Problem, Genetic Algorithm, Crossover, NP-Complete Problems

1. Introduction

Travelling Salesperson Problem (TSP) is NP-Complete problem[1,2]. It is having numerous application zones, for example, drilling holes in PCB, order picking problems, UAV swarming etc. So this problem is impractical to tackle utilizing traditional algorithm. In Travelling Salesman problem, a sales representative needs to visit a given number of cities. Sales representative needs to begin the journey from one of the city and after that visit all different cities precisely once and after that arrived back to the starting city. Travelling salesman Problem is of two sorts, symmetric and awry. In symmetric TSP the distance between any two cities is same in both sides i.e. the distance will stay same while going to the cities in any request. In asymmetric Travelling salesman Problem the distance between two cities is distinctive while going by the cities in various requests. In Travelling Salesman Problem the limitation is that salesman needs to locate the shortest course to visit all the cities. As travelling salesman Problem is NP-Complete and this problem is settled utilizing some different strategies, for example, Genetic Algorithm (GA)[3], Ant Colony Optimization ACO, Particle Swarm Optimization PSO and so forth.

GA is an optimization technique that utilizes its exceptional operators to take care of problems[4]. Keeping in mind the end goal to tackle a problem utilizing GA, a problem is initially encoded into Genetic form so that GA can be applied on it. GA begins with

**Author for correspondence*

era of initial population followed by selection, reproduction and mutation. GA utilize taking after strides to tackle a problem.

Algorithm-1: **Traditional_Genetic_Algorithm**

```
Step 1: Encode a given problem into genetic form.
Step 2: Generate an initial population of chromosomes.
Step 3: Calculate the fitness value of every chromosome
        in the initial population.
Step 4: Perform selection to select parents to perform
        reproduction.
Step 5: Perform reproduction to generate children and
        include these children in population.
Step 6: Perform mutation and discover chromosomes for
        next generation.
Step 7: Repeat steps 4-6 until no of iterations executed or
        an optimal solution found.
```

The goal of this research is to utilize GA to find solution of Travelling Salesman Problem. This paper alters the general GA to search deeper in the solution space and discover solutions for TSP problem which are better than the solutions known by existing algorithm till date.

Multiple solutions have been proposed for the travelling salesman problem by utilizing GA approach. Inclusion of crossover provides quality solutions in less time when contrasted with other existing algorithms[3]. Enhanced differential evolution[5] is another approach solving TSP. In this method, Luihai crossover operator is utilized for settling TSP. This technique enhances the convergence pace of the GA and discovers better solutions in less time when contrasted with other existing calculations. TSP is represented as a framework or matrix that stores the Euclidean distance between the cities[6]. This technique proposes a random cross over operation and finds the solution for TSP problem. A Modified Genetic Algorithm (MGA)[7] is developed to solve Constrained Solid Travelling Salesman Problems (CSTSPs) in crisp, fuzzy, random, random-fuzzy, fuzzy-random and bi-random environments. In the proposed MGA, a new 'probabilistic selection' technique and a 'comparison crossover' are used along with conventional random mutation.

Travelling salesman problem can be solved using a sequential constructive cross over operator[8]. It uses two common subsequence of visited cities in two chromosomes and takes these sub tours as it is in the children. This technique provides a comparative study about different cross over operators such as SCX, ERX, and GNX. While comparing the results it is concluded that the proposed sequential constructive cross over SCX performs better than the other two. The performance of GA and PSO is compared for solving TSP[9]. It also applies the TSP problem in solving UAV swarming. This technique concludes that GA is better technique to solve TSP on the basis of many parameters such as convergence time, finding optimal solutions etc. Hopfield neural network is applied while solving TSP[10]. We try to find a solution of stucking the solution in local optimum which a major problem while solving TSP using GA. TSP is solved using a hybrid HPSACO of ant colony optimization ACO and particle swarm optimization PSO[11]. Hybrid optimization (HPSACO) algorithm is based on combining collaborative strategy of PSO and ACO. We concluded that HPSACO performs better as compared to PSO and ACO.

Next, in section II the existing works which have been done till date to solve Travelling Salesman Problem is explained. Then proposed algorithm has been explained in detail. Section III, contains the experimental results and analysis of the algorithm. Finally, section IV, conclusions and future scope of the paper has been discussed.

2. Discussion

In this work an improved GA has been proposed to find the solution of Travelling Salesman Problem. The proposed algorithm produced better results as compared to other existing algorithms. Greedy approach is used to define the initial population. The initial population is a mix of chromosomes generated using greedy approach and using traditional random method. Further in place of cross over operation a new greedy rearrange operator is used. It rearranges a selected chromosome using greedy approach. Two approaches used for initial population generation and cross over operation using greedy approach are as discussed as follows.

2.1 Greedy Initial Population Generation

To generate initial population a greedy approach that is nearest neighbour first has been used. In this approach a city has been selected as starting city and it will become the current city and then next cities are selected using nearest neighbour first greedy approach that is a next city will be the city which is at the minimum distance among all the cities from the current city. At that point that selected next city will turn into the present city and another next city will be the city at the minimum distance from the present city among all the rest of the cities. This method will remain proceed till all the cities has not been visited. This chromosome will be included in the initial population. Utilizing along these lines some different chromosomes are likewise created and included in the underlying population.

The algorithm to produce a greedy chromosome is as per the following:

Algorithm-2: Generate_Greedy_Chromosome

This algorithm will generate and return a chromosome utilizing greedy approach nearest neighbour first for a given TSP problem.

Step 1: Select a city randomly as the staring city of the chromosome.
Step 2: Add all the cities in a Set_of_Remaining_Cities. Add Starting_city in Set_of_cities_in_greedy_chromosome.
Step 3: Set starting_city as the current_city.
Step 4: Remove current city from the Set_of_remaining_cities.
Step 5: Repeat steps 6-8 till Set_of_remaining_cities is not empty.
Step 6: Select a city from the Set_of_remaining_cities as a next_city which is at the minimum distance among all the cities from current_city in the Set_of_remaining_cities.
Step 7: Set current_city = next_city. Add next_city in set_of_cities_in_greedy_chromosome.
Step 8: Remove current_city from the Set_of_remaining_cities.

2.2 Greedy Cross Over/ Rearrange Operation

This operation will work as a reproduction operator to solve for a given TSP problem using GA. This procedure will take a chromosome as input and rearrange all the cities using greedy approach. It will first select a cross over point. In this work the midpoint of the chromosome is chosen as a cross over point. While creating a child from a parent chromosome the procedure first add all the cities from parent chromosome to child chromosome from first city to city at the cross over point. It then rearranges all the remaining cities using nearest neighbour approach.

Algorithm-3: Greedy rearrange cross over operator

The algorithm for greedy cross over or greedy rearrange operator is as follows:

Step 1: Select a parent chromosome for rearrange.
Step 2: Select a cross over point.
Step 3: Copy all the cities from parent chromosome to child chromosome starting from the first city to city at the cross over point.
Step 4: Copy all the cities after cross over point in parent chromosome into set_of_remaining_cities.
Step 5: Select last city of the child chromosome as the current_city.
Step 6: Repeat steps 7-10 until set_of_remaining_cities is not empty.
Step 7: Select a city from the set_of_remaining_cities as next_city which is at the minimum distance from the current_city.
Step 8: Add next_city in the child chromosome.
Step 9: Remove next_city from the set_of_remaining_cities.
Step10: Set current_city = next_city.

2.3 Improved GA for Solving TSP

In this work greedy approach is utilized to generate introductory population and to rearrange chromosomes in cross over operation.

The enhanced greedy algorithm is as follows.

Algorithm-4: Improved GA

Step 1: Encode a given TSP problem into genetic form.
Step 2: Generate an initial population of chromosomes.
Step 3: Add some chromosomes using Algorithm-2 i.e. Generate_Greedy_Chromosome.
Step 4: Calculate the fitness value of every chromosome in the initial population.
Step 5: Perform selection to select parents to perform reproduction.
Step 6: Generate children chromosome using Algorithm-3: Greedy rearrang cross over operator. Add the newly generated children chromosome in population.
Step 7: Perform mutation and find chromosomes for next generation.
Step 8: Repeat steps 4-7 until no of iterations executed or an optimal solution found

3. Experimental Results and Analysis

The proposed improved GA for understanding tsp has been implemented in JAVA utilizing jdk1.7 and NetBeans8.0.2 IDE. The proposed algorithm is implemented on a well known standard TSP problem that is Eil51. This TSP problem has total 51 cities that a salesman has to visit. The coordinates of all the cities has been downloaded from the library TSP-Lib[12]. This library also provides the optimal path for this problem which is known till date.

The coordinates of all the 51 cities in Table 1 is as follows:

NAME: eil51
COMMENT: 51-city problem (Christofides/Eilon)
TYPE: TSP
DIMENSION: 51
EDGE_WEIGHT_TYPE: EUC_2D

NODE_COORD_SECTION

The optimal path for Eil51 TSP problem given in TSP-Lib directory is represented in Table 2.

NAME: eil51.opt.tour
COMMENT: Optimal tour for eil51.tsp (426)
TYPE: TOUR
DIMENSION: 51
TOUR_SECTION

From library TSP-Lib the optimal path length known to TSP problem Eil51 is 426. The proposed algorithm finds a path of path length 417.91. The chromosomes with path length 417.91 in Table 3 is as follows –

Table 4 shows path length of optimal path found using proposed algorithm.

Table 1. Coordinates of 51 cities

1 - 37 – 52	2 - 49 - 49	3 - 52 - 64	4 - 20 – 26	5 - 40 - 30	6 - 21 - 47
7 - 17 - 63	8 - 31 - 62	9 - 52 - 33	10 - 51 – 21	11 - 42 - 41	12 - 31 - 32
13 - 5 - 25	14 - 12 - 42	15 - 36 - 16	16 - 52 – 41	17 - 27 - 23	18 - 17 - 33
19 - 13 - 13	20 - 57 - 58	21 - 62 - 42	22 - 42 - 57	23 - 16 - 57	24 - 8 - 52
25 - 7 - 38	26 - 27 - 68	27 - 30 - 48	28 - 43 - 67	29 - 58 - 48	30 - 58 - 27
31 - 37 - 69	32 - 38 - 46	33 - 46 - 10	34 - 61 - 33	35 - 62 - 63	36 - 63 - 69
37 - 32 - 22	38 - 45 - 35	39 - 59 - 15	40 - 5 - 6	41 - 10 - 17	42 - 21 - 10
43 - 5 – 64	44 - 30 - 15	45 - 39 - 10	46 - 32 - 39	47 - 25 - 32	48 - 25 - 55
49 - 48 - 28	50 - 56 - 37	51 - 30 - 40	**EOF**		

Table 2. Optimal path for Eil51 TSP problem

1	22	8	26	31	28	3	36	35	20	2	29
	21	16	50	34	30	9	49	10	39	33	45
	15	44	42								
40	19	41	13	25	14	24	43	7	23	48	6
	27										
51	46	12	47	18	4	17	37	5	38	11	32
	-1										
EOF											

Table 3. Chromosomes with path length 417.91

40	19	41	13	25	14	6	48	27	51	46	12
47	18	4	42	44	17	37	15	45	33	10	39
30	34	21	29	2	16	50	9	49	5	38	11
32	1	22	20	35	36	3	28	31	8	26	7
23	24	43	-1								

Table 4. Results comparison

PROBLEM		SWAP_ GATSP	OX_SIM	MOC_ SIM	[10]	PROPOSED ALGORITHM
Eil51 N=51, Optimal = 426	Best	439	493	444	429	417.91
	Average	442	540	453	434	422

Table 4 also demonstrates that the proposed algorithm performs better when contrasted with existing algorithm in terms of path length of optimal path.

Figure 1 shows a bar chart representation of the best solution found using existing algorithm such as SWAP_ GATSP, OX_SIM, MOC_SIM and solution found in the hybrid Search Algorithm with Hopfield Neural Network[10].

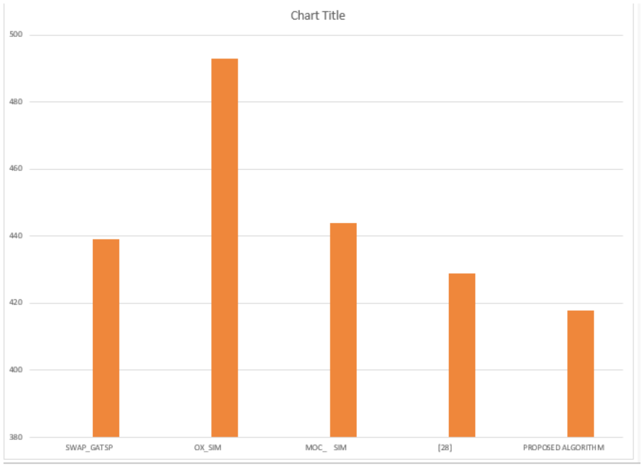

Figure 1. Bar chart representation of the best solution found using existing algorithms and proposed algorithm.

It can be concluded from Figure 1 that the path length of the best path using proposed algorithm is minimum as compared to other existing algorithms.

4. Conclusion

From this work it is concluded that Travelling Salesman Problem can be solved using GA. The performance of GA can be improved by improving its genetic operators such as selection, reproduction and cross over. Further greedy approach is helpful in improving the performance of GA. Proposed algorithm finds a solution which is almost 2% better than the best solution known till date. Future extent of this work is to apply the proposed algorithm on other TSP problems and to check the execution of it. Likewise, some work should likewise be possible to apply greedy approach in some other genetic operators, for example, selection and mutation and this algorithm can likewise be applied on some other NP-Complete problems.

5. References

1. Wikipedia. Travelling salesman problem [Internet]. 2015 [updated 2017 Jun 2; cited 2015 Jul 1]. Available from: Crossref.
2. Wikipedia. NP-completeness [Internet]. 2015 [updated 2017 May 10; cited 2015 Jul 15]. Available from: Crossref.
3. Dwivedi V, Chauhan T, Saxena S, Agrawal P. Travelling salesman problem using genetic algorithm. In the Proceedings of the National Conference on Development of Reliable Information Systems, Techniques and Related Issues; 2012. p. 25–30.
4. Wikipedia. Genetic algorithm [Internet]. 2015 [updated 2017 Apr 10; cited 2015 Jul 15]. Available from: Crossref.
5. Mi M, Huifeng X, Ming Z, Yu G. An improved differential evolution algorithm for TSP problem. In the Proceedings of the Institute of Electrical and Electronics Engineers (IEEE) International Conference on Intelligent Computation Technology and Automation. 2010 May 11–12; 1:544–7. Crossref.
6. Gupta S, Panwar P. Solving travelling salesman problem using genetic algorithm. International Journal of Advanced Research in Computer Science and Software Engineering. 2013 Jun; 3(6):376–80.
7. Maity S, Roy A, Maity M. A modified genetic algorithm for solving uncertain constrained solid travelling salesman problems. Computers and Industrial Engineering, Elsevier, ScienceDirect. 2015 May; 83:273–96. Crossref.
8. Ahmed ZH. Genetic Algorithm for the travelling salesman problem using sequential constructive crossover operator. International Journal of Biometrics and Bioinformatics (IJBB). 2010 Jan; 3(6):96–105.
9. Sathyan A, Boone N, Cohen K. Comparison of approximate approaches to solving the travelling salesman problem and its application to UAV swarming. International Journal of Unmanned Systems Engineering. 2015 Jan; 3(1):1–16. Crossref.
10. Vahdati G, Ghouchani SY, Yaghoobi M. A hybrid Search Algorithm with Hopfield Neural Network and Genetic

Algorithm for Solving Traveling Salesman Problem. In the Proceedings of the Institute of Electrical and Electronics Engineers (IEEE) 2nd International Conference on Computer and Automation Engineering (ICCAE). 2010 Feb 26–28; 1:435–9.

11. Jia H. A novel hybrid optimization algorithm and its application in solving complex problem. International Journal of Hybrid Information Technology. 2015; 8(2):1–10. Crossref.
12. Reinelt G.TSPLIB—A Travelling Salesman Problem Library. ORSA Journal on Computing. 1991 Nov 1; 3(4):376–84.