**IMPACT: International Journal of Research in Engineering & Technology (IMPACT: IJRET)**
**ISSN(E): 2321-8843; ISSN(P): 2347-4599**
**Vol. 2, Issue 2, Feb 2014, 27-34**
**© Impact Journals**

# SOLVING TRAVELLING SALESMAN PROBLEM USING GENETIC ALGORITHM BASED ON HEURISTIC CROSSOVER AND MUTATION OPERATOR

**KANCHAN RANI[1] & VIKAS KUMAR[2]**

[1]Research Scholar, Department of Computer Science, Banasthali University, Sarojani Marg, Jaipur Campus, Jaipur, Rajasthan, India

[2]Associate Professor, Department of Computer Science, Moradabad Institute of Technology, Ram Ganga Vihar Phase II, Moradabad, Uttar Pradesh, India

## ABSTRACT

Genetic Algorithm (GAs) is used to solve optimization problems. It is depended on the selection operator, crossover and mutation rates. In this paper Roulette Wheel Selection (RWS) operator with different crossover and mutation probabilities, is used to solve well known optimization problem Traveling Salesmen Problem (TSP). We have compared the results of RWS with another selection method Stochastic Universal Selection (SUS), which demonstrate that the SUS is better for small number of cities; but as the number of cities increases RWS is much better than SUS. We have also compared the results with a variation between mutation & crossover probability, which concludes that mutation, is more effective for decimal chromosome. We have proposed a new crossover operator which is variation of Order Crossover (OX) and found results are better than existing crossover operator.

**KEYWORDS:** TSP, GAs, SUS, RWS, OX

## INTRODUCTION

Traveling Salesman Problem (TSP) is one of the most significant optimization problems. TSP, as a general NP-complete problem can be developed to be an admissible solution for any other problems that belongs to NP- complete class. genetic algorithm is search algorithms based on the mechanics of natural selection and natural genetics [1], various operators to solve optimization problems using a survival of the fittest idea.

TSP is one of the well known combinatorial optimization problem in which we have to find the tour of all nodes that has the minimum total cost [3, 4]. When no. of cities gets large, it becomes exhaustive search and it is impracticable to find the cost of every tour in polynomial time. Many different methods of optimization have been used to solve the TSP such as Hill Climbing [4], Tabu Search [5], Simulated Annealing [6], Particle Swarm [7], Ant Colony [8] and Genetic Algorithm [9, 4, 10] etc. Here we have used GA to solve TSP, which is heuristically, good solution in reasonable time & establishing the degree of goodness. In this paper, we have considered a symmetric weighted graph and for a cost matrix Euclidean method is used.

In this paper, RWS is used, which is most democratic selection method, in which highest fitted individual is selected for the crossover and mutation. We have solved TSP using RWS and compared its results with Stochastic Universal Selection method. The crossover recombines two individuals to generate new ones which might have a better performance. Most commonly used crossover methods to solve TSP problems are Partially Matched Crossover (PMX),

**This article can be downloaded from www.impactjournals.us**

Order Crossover (OX) & Cycle Crossover (CX) [1]. In this paper we have proposed variation of order crossover and compared its results with existing ones. The results were compared according to different mutation and crossover probabilities.

## METHOD OF GA FOR TSP

Pure Genetic Algorithm is started with a set of solutions (chromosomes/individuals) called populations. It is chosen from collection of candidate solutions to a problem (search space) [2]. Solution for one population are taken and used to form a new population. This is motivated by a hope that new population will be better than the old one. Solutions, which are selected to form new population (offspring), are selected according to their fitness.

- **Permutation Encoding**

As Travelling Salesman Problem is ordering problem so we have used permutation encoding. In permutation encoding, every chromosome is a string of numbers, which represents number in a sequence, number represents each city. The idea of TSP is to find cheapest round-trip tour a salesman has to take by starting from any city and visiting all the cities once and ending at the starting city. Let us consider five cities. A Chromosomes (possible solution) which is represented by (1, 5, 3, 4, 2) e.g. (1, 5, 3, 4, 2) says order of cities, in which salesman will visit 1→5→3→4→2→1 them.

- **Fitness Function**

As TSP is a minimization problem so to convert it into maximization problem we have considered fitness function f(x) =1/d, where d calculates cost (or distance/length) of the tour represented by a chromosome. The fitness function that characterizes each chromosome represents the total length of the route from the first to the last gene (city) moving according to the order of the genes in the chromosome. If the cities are represented with x and y coordinates in 2D coordinate system, then we calculate the distance between them according the equation [3]:

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

- **Selection and Elitism**

When creating new population by crossover and mutation, there is a big chance, that we will lose the best chromosome. Elitism is name of the method, which first copies the best chromosome (or a few best chromosomes) to new population. The rest is performed in classical way. Elitism can very rapidly increase performance of GA, because it prevents from loss of the best found solution. The elitism rate directly depends on the size of the population and the rate of elitism should be decreased when the population size increases [12]. In this paper, we have chosen two best fitted chromosomes according to their fitness and all the chromosomes were used in subsequent procedure. In the resultant population to worst chromosomes were replaced by to best selected chromosomes and repeat this for all generation.

- **Selection**

After deciding the methods of encoding and elitism, the decision for selection technique is to be made. Various selection methods are available such as Roulette-Wheel Selection [1], Tournament selection, Rank Selection, Steady-State Selection, Boltzmann Selection [13] etc. In this paper we have considered Roulette-Wheel & Stochastic Universal Selection technique for selection. Selection is to evaluate each individual and keeps only the fittest ones among

them. In addition to those fittest individuals, some less fit ones could be selected according to a small probability. The others are removed from the current population [2].

**Roulette-Wheel (Goldberg 1989a)**

In this method each individual is assigned a slice of a circular "roulette wheel", the size of the slice being proportional to the individual's fitness. The wheel is spun *N* times, where N is number of individuals in the population. On each spin, the individual under the wheel's marker is selected to be in the pool of parents for the next generation.

**Stochastic Universal Selection (James Baker 1987)**

To minimize the "spread" (the range of possible actual values, given an expected value) in RWS often SUS technique is used. Instead of spinning the roulette wheel *N* times to select *N* parents; SUS spins the wheel once, but with *N* equally spaced pointers, which are used to select the *N* parents. Under this method, each individual *i* is guaranteed to reproduce at least *ExpVal*(*i*,*t*) times but not more than *ExpVal*(*i*,*t*).

If fitness variance in the population is high and a small numbers of individuals are much fitter than the other.

- **Proposed Heuristic Crossover Operation**

So far, so many common crossover operators, such as PMX, CX and OX, have been introduced. But none of them consider the relation between edges in TSP. So they may not accelerate the speed of the algorithm. Here, a heuristic crossover operator, which can increase the algorithm speed, has been proposed. Although it must be mentioned that due to the goal function of TSP which is finding the shortest total distance in one closed cycle, parent chromosomes will be represented in a closed cycle, too [12].

**New Variation in Order Crossover**

We have proposed variant of order crossover (Ox6), which is similar as to cut point selection of the first variation (Ox2) [14], except the repairing procedure. Two cut points are selected and the elements between them are copied. The rest elements are copied from the beginning of the second parent respecting their relative order omitting those which already exist in the first parent.

Consider the above example.

O1 = (- - _ 3 4 5 _ - - - -) and

O2 = (- - _ 7 1 2 _ - - - -)

The sequence of cities in the second parent is 7 1 2 4 9 3 6 8 5

Removing 3, 4 and 5 we get 7 1 2 4 9 6 8

Placing this sequence in the first offspring from left to right according to the order we have

O1 = (7 1 3 4 5 2 4 9 8) Similarly O2 = (3 4 7 1 2 5 6 8 9).

- **Proposed Mutation Operator**

The mutation in GAs works on a single chromosome at a time and alters the genes of the chromosomes randomly.

To solve TSP using GA, the mutation plays the primary role in arriving at the solution. Its purpose is to maintain the population diversified enough during the optimization process. In this paper we have used Inversion Mutation (IVM). In which GAs chooses a particular chromosome is chosen at random for mutation and two indices within the chromosome are randomly selected. In Table 1 randomly select 2 and 5 indices and the order of genes within the indices is reversed.

**Table 1**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| A | | 4 | 5 | **6** | 7 | 1 | 2 | **3** |
| B | 1 | 2 | **6** | **5** | **4** | | **3** | 7 |

The probability of mutation generally is very low and it is of the order of one tenth of a percent for binary chromo some. But in the case of decimal chromosomes, the mutation rate goes up to of the order of 85%.

## EXPERIMENTAL RESULTS AND DISCUSSIONS

In this, experiments all computation performed on a Pentium processor with minimum 512 MB RAM under windows XP. The program was written in the C programming language. We tested RWS & SUS method for different crossover method. It is perform on six bench mark problem instances which are taken from the TSPLIB [15]. They are categorized in to two categories.

The first in Table 2 are those instances whose number of cities is less than or equal to 51 for Roulette Wheel Selection & Stochastic Universal Selection. We have us pc 0.5 and pm 0.5. Table 3 compares results new variation order crossover and other available crossover with pc 0.5 and pm 0.8. Table 4 we have change crossover probability: pc 0.9 and pm 0.1.

- **Control Parameter**

GA performs well in cases where it is more important to find a good solution rather than the absolutely best solution [5]. Population Size is determines how many chromosomes and thereafter, how much genetic material is available for use during the search. If there is too little, the search has no chance to adequately cover the space. If there is too much, the GA wastes time evaluating chromosomes. For all instances probability of crossover (pc) is 0.5 or 0.9. Probability of mutation (pm) is 0.8 or 0.1after a continuous exhaustive fine tuning.

The mutation used here is Inversion Mutation after experiments are done to compare with reciprocal exchange mutation. Initial population generated by random function, which is inbuilt in C library and pseudo-random number [16] are used in selection method. Here we have used pure genetic algorithms. As to the selection we used roulette wheel or SUS with elitism. The maximum generation in all problems was 30000.

- **Termination Criteria**

The stopping criterion used here is just the numbers of generations evolved. All individual are equal & accepted range. Thus, the running of GA stops after completing certain number of iterations. The best chromosome in the last generation is the solution which may be either a local optimum or global optimum. It is Important to remember that GA in most cases provide only approximately correct solution, not optimal. The solution quality is measured by optimal value reported in TSPLIB [15].

```
begin
        GA-TSP
          Create initial population ( c library Random function)
          while (generation count  k)
                    (All individual cost are not equal) do

           /*k  max. Number of generations. */

          begin
                Elitism (2 best tours) Selection
               Heuristic Crossover m non linear adjusting
                crossover probability
                Mutation Crossover    non linear adjusting
                crossover probability
                Increment generation count
           end

          Output the best and worst individuals found

          end GA-TSP
```

**Figure 1: Pseudo Code of GA for TSP**

**Table 2**

| | | | Roulette Wheel Selection | | | Stochastic Universal Selection | | |
|---|---|---|---|---|---|---|---|---|
| **Problem and Optimal Value** | **Pop. Size** | **Optimal Cost** | **PMX** | **CX** | **OX** | **PMX** | **CX** | **OX** |
| cit5(5 cities) | 5 | 17 | 17.3 | 17.3 | 17.9 | 17 | 17 | 17.9 |
| cit10(10 cities) | 10 | 29 | 29 | 29 | 29 | 29 | 29 | 29 |
| cit20(20 cities) | 20 | 30 | 33.4 | 31.4 | 31.6 | 32.6 | 31.2 | 32.4 |
| eil51(51 cities) | 50 | 426 | 444.8 | 447.85 | 439.4 | 453 | 456.87 | 444.6 |

In Table 2 demonstrate comparison of RWS and SUS. It shows that SUS gives a better result for small number of cities and converges faster as compared to RWS. RWS takes maximum15 generation for 10 numbers of cities; While SUS takes less than 10 generation for the same cities. But as number of city get large. RWS gives better results than SUS.

**Table 3**

| Roulette Wheel Selection | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Problem and Optimal Value** | **Pop. Size** | **Optimal Cost** | **PMX** | **CX** | **OX** | **OX2** | **OX3** | **OX6** |
| eil51(51 cities) (426) | 50 | Avg. | 448.62 | 442.75 | 447.12 | 452.5 | 446 | **447.1** |
| | | Worst | 461 | 451 | 453 | 458 | 453 | **457** |
| | | Best | 436 | 437 | 439 | 445 | 442 | **441** |
| | | Time | 258 | 272 | 222 | 213 | 260 | **200** |
| st70(70 cities) (675) | 70 | Avg. | 744.5 | 722 | 715.3 | 712.4 | 703.2 | **708.5** |
| | | Worst | 849 | 734 | 743 | 734 | 712 | **728** |
| | | Best | 714 | 696 | 687 | 698 | 690 | **688** |
| | | Time | 244 | 336 | 751 | 323 | 345 | **337** |

In Table 3 demonstrate comparison of different crossover methods in which we have used Crossover probability is 0.5 & mutation probability 0.8. It shows that OX6 give a faster result comparatively to other crossover. OX6 takes 200 second for best result while other crossover takes more time for 51 cities.

**Table 4**

| Roulette Wheel Selection | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Problem and Optimal Value** | **Pop. Size** | **Optimal Cost** | **PMX** | **CX** | **OX** | **OX2** | **OX3** | **OX6** |
| eil51(51 cities) (426) | 50 | Avg. | 444.5 | 445.2 | 441.1 | 438.7 | 436.6 | 440.2 |
| | | Worst | 455 | 448 | 450 | 444 | 442 | 446 |
| | | Best | 433 | 442 | 437 | 435 | 429 | 432 |
| | | Time | 240 | 401 | 228 | 218 | 271 | **188** |
| st70(70 cities) (675) | 70 | Avg. | 715 | 733.2 | 707.6 | 701.4 | 717 | 730 |
| | | Worst | 744 | 752 | 732 | 715 | 734 | 750 |
| | | Best | 704 | 713 | 686 | 689 | 694 | 725 |
| | | Time | 1081 | 247 | 337 | 1452 | 340 | **200** |

In Table 4 we have used Crossover probability is 0.9 & mutation probability 0.1. If we compare Table 3 and Table 4 we have find result affected by mutation probability rate.

## CONCLUSIONS AND FUTURE WORK

The conclusion has to be that heuristic mutation gives better result when number of cities is large & new variation order crossover is faster as compare with already existing variation order crossover which is depended on rate of crossover, mutation and elitisms. Initial population is most important for GAs. So, Further we will use Tabu search & Nearest-Neighbor Heuristic algorithm which is over come initial population problem.

## REFERENCES


1. Goldberg, D.E. (1989) “Genetic Algorithms in search, optimization machine learning”, Pearson Education.
2. Mitchell M. “An Introduction to Genetic Algorithms”, Prentice-Hall of India.
3. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein, “Introduction to Algorithms Second Edition”, PHI Learning.
4. Ellis Horowitz, Sartaj Sahni, Rajasekarn S. “Fundamentals of Computer Algorithms”, Galgotia Publication pvt.ltd.
5. Stojanovska, I., Dika, A. “Impact of the Number of Generations on the Fitness value and the Time Required to Find the Optimal Solution in Standard GA Applications”, Proc of the 2010 IEEE, pp. 978-1-4244-5540-9(2010).
6. Thamilselvan, R., Balasubramanie, P., “A Genetic Algorithm with a Tabu Search (GTA) for Traveling Salesman Problem”, International Journal of Recent Trends in Engineering, Vol. 1, No. 1, May 2009 ACEEE.
7. Laarhoven, P.V. and Aarts, E.H.L.: Simulated Annealing: Theory and Applications, Kluwer Academic (1987).
8. Kennedy, J. and Eberhart, R.C.: Swarm Intelligence. Morgan Kauffman publishers, San Francisco (2001).
9. Dorigo, M. and Gambardella, L.M.: Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. IEEE Transactions on Evolutionary Computation, Vol. 1, No. 1 (1997) 53-66.
10. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution programs, Berlin: Springer-Verlag (1992).
11. Wang, Z. Duan, H. and Zhang, X. “An Improved Greedy Genetic Algorithm For Solving Travelling Salesman

Problem", Proc of the 2009 IEEE Fifth International Conference on Natural computation (ICNC), pp 978-0-7695-3736-8,2009.504.

12. KAUR, D., MURUGAPPAN, M. M., "Performance Enhancement in solving Traveling Salesman Problem using Hybrid Genetic Algorithm" pp. 978-1-4244-2352-1/08, 2008 IEEE.
13. Rajasekaran S., Vijayalakshmi Pai G.A., "Neural Networks, Fuzzy Logic, and Genetic Algorithms Snthesis and Applications", Prentice-Hall of India.
14. Deep, K. Mebrahtu, H. "New Variation Of Order Crossover for Travelling Salesman Problem", International Journal of Combinatorial Optimization Problem and Informatics, Vol.2, No.1, Jan-April 2011, pp. 2-13
15. http://www2.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB9 5/tsp/
16. NARASINGH DEO "System Simulation with Digital Computer", Prentice-Hall of India.
17. Takahashi, R. "Solving the Traveling Salesman Problem through Genetic Algorithms with Changing Crossover Operators", Proc of the 2009 IEEE Fourth International Conference on Machine Learning and Applications (ICMLA'05), pp. 0-7695-2495-8.

**This article can be downloaded from www.impactjournals.us**