ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

A Method of Solving Travelling Salesman Problem

US.Kirutikaa[1], Dr.SS.Dhenakaran[2]

M. Phil Research Scholar, Department of Computer Science, Alagappa University, Karaikudi, Tamil Nadu, India [1]

Professor, Department of Computer Science, Alagappa University, Karaikudi, Tamil Nadu, India[2]

ABSTRACT: Travelling Salesman Problem is one of the heuristic search based technique that leads to reduce time and resources in marketing products. Various methods are applied into travelling salesman problem for reducing the cost of time. Genetic Algorithms (GA) are biological representation approach that works similar to the genetic processing. GA's have been applied to optimization problems and have been shown to be effectively in searching large, Complex response surfaces even in the presence of difficulties such as high multidimensionality and multimodality. Branch and Bound (BB) method is another efficient method for finding solutions in optimization problem. This method solves a discrete optimization problem by breaking up its feasible set into successfully smaller subsets, calculating bounds of objective functions. In this paper, both Genetic algorithm and Branch and Bound techniques are applied to travelling sales man problem and performance is evaluated. The experimental result exposes the strength of branch and bound technique.

KEYWORDS: Genetic Algorithm, Travelling Salesman Problem, Branch and bound and Memory space .

I. INTRODUCTION

Travelling Salesman Problem (TSP) is widely studied combinatorial optimization problems and still it is challenged in operational research[4]. The objective of TSP is to find the shortest path, starting from the home city till the destination. The main difficulty of this problem is the immense number of possible tours (n-1)! /2 for n cities.

Definition:

Let G=(V,E) be a graph where V a set of n vertices. E is a set of edges, and let S= (Sij) be a distance matrix connected with E. The TSP consists of designing a minimum distance circuit passing through each vertex once and only once. Such a circuit is known as tour. The most common practical understanding of TSP is that of a salesman finding the shortest path through n cities.

Formulation

Combinatorial optimization problems can often formulated in several different ways. The standard formulation is the following due to Dsnty fulkorwe[14]. Let G be undirected graph with node set V and edge set E. Let x_e denote the cost of travelling edge e. for any node set $S \subseteq V$. Let δ(s) denote the set of edges with exactly one end-node in S. Next, for each edge $e \in E$, we define a binary variable xe taking value 1 if and only if edge e is in the tour. Finally, for any set of edges $F \subseteq E$, let x(F) denote

$$\sum_{e \subseteq F} x_e \qquad (1)$$

$$\text{Min} \sum_{e \in E} c_e x_e \qquad (2)$$

$$\text{s.t.} \qquad x(\delta(\{i\})) = 2 \ (\ \forall i \in V) \qquad (3)$$

$$x(\delta(\{S\})) >= 2 \ (\ \forall S \subset V: 2<=|S| <=\{V\}/2) \qquad (4)$$

$$x \in \{0,1\}^{|E|} \qquad (5)$$

The equations(1), called degree equations, simply express the fact that the salesman must arrive at and apart from each city. The inequalities (2) called subtour elimination constraints, ensure that the tour is connected.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

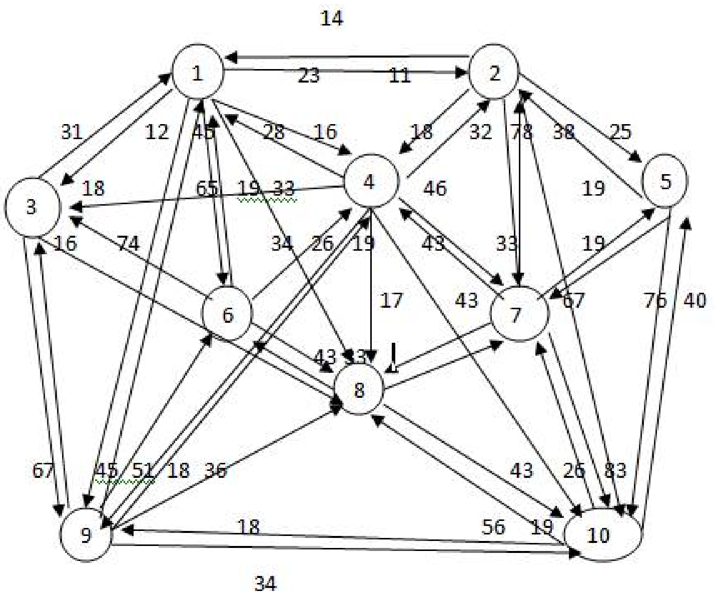

Fig1. The graph of TSP

II. Genetic Algorithm

Genetic algorithm is a heuristic search algorithm which has been given solutions for optimization problems [1][2]. The model of genetic algorithm is based on the natural evolution which is carried out in living beings. The search space contains individual chromosomes. Generally fitness chromosomes are selected for mutations in living beings. In genetic algorithms, there are three types of operations such as selection operation, cross over operation and mutant operation. There are various types of cross over operations applies into selected parents for achieving the optimized result.

Firstly, the mutation method helps to shuffling the route, The cities couldn't 't be add or remove while mutation process initiated. Otherwise it would risk creating an invalid solution. Swapping is the One type of mutation we could use is swap mutation.

Two location selects in the tour at random than their positions are quietly swapped. For example, if we apply swap mutation to the following list, [1,2,3,4,5] the designation might end up with, [1,2,5,4,3]. Here, positions 3 and 5 were swapped creating a new list with accurate the same values, just a different order. Because swap mutation is only swapping pre-existing values, the operation doesn't creates and it will never a list which contains missing or duplicate values when compared to the original, and that's exactly what we want for the traveling salesman problem.

1	2	3	4	5	6	7	8	9

1	2	8	4	5	6	7	3	9

Fig2. The Cost matrix of GA algorithm

In the GA, mutation method and crossover methods enforce can enforce the same constraints.

Each cross over method can able to product a route is ordered crossover. In this crossover method select the subset value from the parent, and then add that subset to the created next set. Any missing values are then adding to the next generated set from the second parent in order that they are found.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

To make this explanation a little clearer consider the following example:

Parents

1	2	3	4	5	6	7	8	9

9	8	7	6	5	4	3	2	1

Offspring

					6	7	8	

9	5	4	3	2	6	7	8	1

Fig3. The Cost matrix of GA algorithm

Here the first parent generates a subset of the route is (6,7,8) and added to the offspring's route. Next, the second parents missing route locations are adding in order from the second parent. The value 9 is the first location of second parent route. This isn't in the offspring's route so it's appended in the first available position. The value 8 is the next position in the parent's route which is in the offspring's route so it's skipped. This process continues until the offspring has no remaining empty values. End result contains one route which having all of the positions it's parents did with no positions missing or duplicated.

III. Branch and Bound Technique

The BB is used for solving large scale NP hard combinatorial optimization problem[10]. This algorithm searches the complete space of solutions for a given problem for the best solution. The use of bounds for the function to be optimized combined with the value of the current solution enables the algorithm to search parts of the solution space only implicitly.[11]. In a typical iteration the algorithm executes the following steps.

- It selects a leaf of the branching tree, i.e. a subproblem not divided yet into further subproblems.
- The subproblem is divided into further subproblems (branches) and their relaxations are defined.
- Each new relaxed subproblem is solved and checked if it belongs to one of the above-mentioned cases. If so then it is fathomed and no further investigation is needed. If not then it must be stored for further branching.
- If a new feasible solution is found which is better than the so far best one, then even stored branches having an upper bound less than the value of the new best feasible solution can be deleted without further investigation.

IV. Branch and Bound for TSP

In the tree, two cities are connecting the edges which have specified constraints in each node. Edges connect the two cities based on the constraints A sub-tree tress of nodes lead to generate the legal tour which containing leaf nodes[12][13] , if lower bound is higher than best solution that the node may be pruned. This pruning has the effect of spar the computational process the need to generate nodes below the given node. This could result in a significant saving if the pruned node were relatively near the top of the tree.

The result tree is built by adding edges in lexicographic order. In each time, A new node adds and employ the decision tree logic which nodes much be included or excluded from the tours which is represent by nodes.

The rules used are:

1. If eliminating an edge (x, y) would make it unfeasible for x or y to have as many as two adjacent edges in the tour, then (x, y) must be added.

2. If take a account of (x, y) would cause x or y to have more than two edges adjacent in the tour, or would finish a incomplete tour cycle with edges already included, then (x, y) must be eliminated. When the algorithm branches, and after magnifying the decision logic to add or eliminate edges, a lower bound is calculated for the node. If the lower bound for a selected node is as high as or higher than the lowest cost tour found so far, the node can be cut back. If neither child can be cut back, the algorithm descends to the node with least lower bound using a depth-first search in the tree. After taking one child, we must again consider whether the sibling can be cut back since a new best solution may have been achieved.

V. GENETIC ALGORITHM FOR TSP

Genetic algorithms have been applied to a variety of function optimization problem. Therefore it is also suitable for TSP. The TSP is often used to illustrate heuristic search methods [7][8]. It can be used to solve the travelling salesman problem. It not only gives accurate results for optimization problems as compared to its counterparts, but the genetic operators that it provides exceedingly helps in speeding up the search process[3][4].In the TSP context, each chromosomes encodes a solution to the problem(i.e., a tour). The fitness of the chromosome is related to the tour length, which in turn depends on the ordering of the cities. Since the TSP is minimizing the problem, the tour lengths must be transformed so that high fitness values are associated with short tours, and conversely. A well known approach is to subtract each tour length to the maximum tour length found in current population. Assuming a TSP size N each city would be coded using log2N bits, and the whole chromosome would encode a tour as a sequence of N * [log2N] bits.

```
Genetic Algorithm()
{
Initialize population of routes of cities randomly with a function Random()
Evaluate the fitness of  each individual  route using function fitness()
While the fitness criteria is not satisfied
Do
{
Selection of two routes for reproduction using select function
(select(parent_route1,parent_route2))
Perform cross over on the selected parent routes with crossover function
(child_route= rossover(parent_route1,parent_route2))
Perform mutation on the newly generated child routes  with mutation function
( mutation (child_route))
Evaluate the fitness of child_route and replace the parent population with child_route
}
}
```

VI. EXPERIMENTAL RESULTS AND ANALYSIS

The architecture of algorithm is being implemented for solving the TSP through genetic algorithm and BB technique. Consider a set of 20 cities and 6 different routes. The cities are represented in a graph coordinates. The distance of cities are representing difference between X,Y coordinates of two cities. The distance is calculated by following equation.

Xcoordinate= fromcity(x axis)- tocity(xaxis)

Ycoordinate = fromcity(xaxis) – tocity(yaxis)

Distance = Math.sqrt((Xcoordinate * Xcoordinate) + (ycoordinate*ycoordinate))

A model of TSP problem with 10 cites and cost of transportation is given below:

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

$$
\begin{array}{cc}
 & \begin{array}{cccccccccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{9} & \mathbf{10} \end{array} \\
\begin{array}{c} \mathbf{1}\\ \mathbf{2}\\ \mathbf{3}\\ \mathbf{4}\\ \mathbf{5}\\ \mathbf{6}\\ \mathbf{7}\\ \mathbf{8}\\ \mathbf{9}\\ \mathbf{10} \end{array} &
\begin{pmatrix}
0 & 23 & 12 & 16 & 34 & 65 & 28 & 33 & 45 & 8 \\
14 & 0 & 27 & 18 & 25 & 61 & 78 & 41 & 28 & 38 \\
31 & 23 & 0 & 38 & 17 & 18 & 71 & 23 & 67 & 42 \\
28 & 32 & 18 & 0 & 27 & 39 & 46 & 17 & 19 & 43 \\
23 & 14 & 15 & 43 & 0 & 75 & 67 & 45 & 23 & 76 \\
19 & 43 & 74 & 34 & 28 & 0 & 74 & 43 & 18 & 46 \\
54 & 33 & 46 & 26 & 19 & 6 & 0 & 30 & 41 & 26 \\
65 & 23 & 34 & 17 & 29 & 53 & 71 & 0 & 36 & 19 \\
28 & 34 & 16 & 26 & 42 & 51 & 61 & 12 & 0 & 18 \\
61 & 23 & 16 & 32 & 40 & 16 & 83 & 56 & 34 & 0
\end{pmatrix}
\end{array}
$$

Shortest Path: $1 \rightarrow 10 \rightarrow 3 \rightarrow 5 \rightarrow 2 \rightarrow 4 \rightarrow 9 \rightarrow 8 \rightarrow 6 \rightarrow 7 \rightarrow 1$

Minimum Cost: 285

Time Taken: 0.007

Memory Space: 2359256 bytes

Fig4. The outcome of GA algorithm

$$
\begin{array}{cc}
 & \begin{array}{cccccccccc} \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{9} & \mathbf{10} \end{array} \\
\begin{array}{c} \mathbf{1}\\ \mathbf{2}\\ \mathbf{3}\\ \mathbf{4}\\ \mathbf{5}\\ \mathbf{6}\\ \mathbf{7}\\ \mathbf{8}\\ \mathbf{9}\\ \mathbf{10} \end{array} &
\begin{pmatrix}
0 & 23 & 12 & 16 & 34 & 65 & 28 & 33 & 45 & 8 \\
14 & 0 & 27 & 18 & 25 & 61 & 78 & 41 & 28 & 38 \\
31 & 23 & 0 & 38 & 17 & 18 & 71 & 23 & 67 & 42 \\
28 & 32 & 18 & 0 & 27 & 39 & 46 & 17 & 19 & 43 \\
23 & 14 & 15 & 43 & 0 & 75 & 67 & 45 & 23 & 76 \\
19 & 43 & 74 & 34 & 28 & 0 & 74 & 43 & 18 & 46 \\
54 & 33 & 46 & 26 & 19 & 6 & 0 & 30 & 41 & 26 \\
65 & 23 & 34 & 17 & 29 & 53 & 71 & 0 & 36 & 19 \\
28 & 34 & 16 & 26 & 42 & 51 & 61 & 12 & 0 & 18 \\
61 & 23 & 16 & 32 & 40 & 16 & 83 & 56 & 34 & 0
\end{pmatrix}
\end{array}
$$

Shortest Path: $1 \rightarrow 10 \rightarrow 3 \rightarrow 5 \rightarrow 2 \rightarrow 4 \rightarrow 9 \rightarrow 8 \rightarrow 6 \rightarrow 7 \rightarrow 1$

Minimum Cost: 285

Time Taken: 0.003

Memory Space: 362128 bytes

Fig.5. The result of BB algorithm

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

Table I. Results of TSP through Genetic Algorithm

No of Cities	Cost of Distance	Time in secs
5	101	0.007
20	927	0.097
40	1348	0.232
60	2115	0.434

Table II. Results of TSP through Branch and Bound technique

No of Cities	Cost of Distance	Time in secs
5	101	0.003
20	910	0.0175
40	1118	0.0280
60	1892	0.0395

No.of Cites	Genetic Algorithm		Branch and Bound technique	
	Time	MS (bytes)	Time	MS (bytes)
10	0.007	2359256	0.003	362128
20	0.0175	4102728	0.097	362128
40	0.0280	3055104	0.232	362880
60	0.0395	1404304	0.434	387936

GA-Genetic algorithm
BB- Branch and bound
Time-Time taken to complete the travelling path
MS-Memory space(Program,Data,Temp Variables)

Initially, twenty cities have taken to solve the Travelling Salesman Problem through genetic algorithm in first iteration. There are 200 mutations are processed and the results have given the distance value and execution time in secs. It is shown in table I. In the next two iterations, it increases the cities in 40 and 60. If cities increase, the execution time is also increase in mutually exclusive. In table II shows the result for TSP which has executed through Branch and bound technique, the scenarios of city selection has following similarly in Branch and Bound technique. From the analysis of both algorithms, the overall performance of Branch and Bound technique is more efficient than Genetic Algorithm that is based on distance value and execution value. In every iteration, the metric values lead to deliver the performance which is more efficient than genetic algorithm.

VII. CONCLUSION

In this paper, travelling salesman problem has implemented in Genetic algorithm and Branch and Bound technique. More than 50 cities are taken for experimental test. The implementation result shows the performance of branch bound technique and Genetic algorithm. The Branch and Bound technique is efficient than genetic algorithm based on results as shown in the table I and table II. Time execution is reasonably decreased than time execution of genetic algorithm. In future, TSP will be implemented in parallel computing to find best solution from considering of large number of cities.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 4, Issue 7, July 2016

REFERENCES

1. D.B.Fogel, An Evolutionary Approach to the Travelling Salesman Problem,Biol.Cybem 1988,pp139-144
2. open loop Travelling Salesman Problem Using Genetic Algorithm, International Journal of Innovative Research in Computer and Communication Engineering Vol 1 issue1, March 2013.
3. Alba E. and Aldana J.F. "Genetic Algorithms as Heuristic in Optimization Problems. An Overview" (in spanish), Technical Report, Dept. of Lenguajes yCiencias de la Computación (Univ. of Málaga), 1992
4. Gutin, G. and A.P. Punnen, eds. (2002). The Traveling Salesman Problem and its Variations, Kluwer, The Netherlands.
5. Lin, S. & Kernighan, B. W. (1973). An Effective Heuristic Algorithm for the Traveling Salesman Problem. Operations Research 21: 498–516.
6. Zhao, F., Dong, J., Li, S., & Yang, X. (2008, July). An improved genetic algorithm for the multiple traveling salesman problem. In Control and Decision Conference, 2008. CCDC 2008. Chinese (pp. 1935-1939). IEEE.
7. Zhang Jingqang, Cao Yunfu A Genetic Algorithm Based on Common Path for TSP [J]Journal of Computer Engineering and Applications, 40 (20) (2004), p. 58~61
8. Oncan, T, Altinel, IK, Laporte, G. A comparative analysis of several asymmetric traveling salesman problem formulations. Comput. & Oper.Res. 2009; 36: 637{654}
9. Lawler, EL, Lenstra, JK, Rinnooy Kan AHG and Shmoys, DB, eds.The Traveling Salesman Problem. 1985. Wiley, Chichester.n JOURNAL OF OBJECT TECHNOLOGYOnline at . Published by ETH Zurich, Chair of Software Engineering ©JOT, 2003 Vol. 2, No. 2, February-March 2003
10. Richard Wiener: Branch and Bound Implementations for the Traveling Salesperson Problem - Part 1, in Journal of Object Technology, vol. 2, no. 2, March-April 2003, pp. 65-86. http://www.jot.fm/issues/issue_2003_03/column7
11. Branch and Bound Implementations for the Traveling Salesperson Problem, Richard Wiener, Editor-in-Chief, JOT, Associate Professor online at www.jot.fm. Published by ETH Zurich, Chair of Software Engineering
12. Archit Rastogi, Ankur kumar shrivastava,et.all, A Proposed Solution to Travelling Salesman Problem using Branch and BOund, International Journal of Computer Applications (0975 –8887) Volume 65–,pp. 44-49 2013
13. Egos Balas, Paolo Toth,Branch and Bound methods for the travelling salesman, problem,Management Science Research Report,1983