**“Implementing Genetic Algorithm for solving TSP”**

***A***

***Project Report***

*submitted in partial fulfillment of the*

*requirements for the award of the degree of*

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE & ENGINEERING**

**With Specialization in**

**Telecom Informatics**

**by**

|  |  |
| --- | --- |
| **Name** | **Roll No.** |
| **Amardeep Kumar** | **R106215053** |
| **Rohnit Bhagat** | **R106215026** |
| **Shashank S. Karchuli** | **R106215028** |
| **Udit Chowdhry** | **R106215036** |  |

***Under the guidance of***

**Mr. Amar Shukla**

Assistant Professor

Department of Informatics

****

**School of Computer Science and Engineering**

**University of Petroleum & Energy Studies**

**Bidholi, Via Prem Nagar, Dehradun, UK**

**December – 2017**

****

**CANDIDATE’S DECLARATION**

I/We hereby certify that the project work entitled **“Implementing Genetic Algorithm for solving TSP”** in partial fulfilment of the requirements for the award of the Degree of BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING with specialization in Telecom Informatics) and submitted to the Department of Informatics at School of Computer Science and Engineering, University of Petroleum & Energy Studies, Dehradun, is an authentic record of my/ our work carried out during a period from **August, 2017** to **December, 2017** under the supervision of  **Mr. Amar Shukla , Assistant Professor, SOCS, UPES.**

The matter presented in this project has not been submitted by me/ us for the award of any other degree of this or any other University.

**(Rohnit Bhagat)**

**Roll No. - R106215026**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: 12th December, 2017

**Mr. Amar Shukla**

Assistant Professor

SOCS, UPES

**Mr. M Venkatadari**

Program Head - Telecom Informatics

Department Of Informatics

University of Petroleum & Energy Studies

Dehradun – 248 001 (Uttarakhand)

**ACKNOWLEDGEMENT**

We wish to express our deep gratitude to our guide **Mr. Amar Shukla, Assistant Professor-SOCS** for all advice, encouragement and constant support he has given us through out our project work. This work would not have been possible without his support and valuable suggestions.

We sincerely thank to our respected Program Head of the Department, **Dr. M Venkatadari**, for his great support in doing our project in **Area (like Network, Big data etc.)** at **CIT**.

We are also grateful to **Dr. Manish Prateek, Director SoCS** for giving us the necessary facilities to carry out our project work successfully.

We would like to thank all our **friends** for their help and constructive criticism during our project work. Finally we have no words to express our sincere gratitude to our **parents** who have shown us this world and for every support they have given us.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Amardeep Kumar** | **Rohnit Bhagat** | **Shashank S.Karchuli** | **Udit Chowdhry** |
| **Roll No.** | **R106215053** | **R106215026** | **R106215028** | **R106215036** |

**ABSTRACT**

Travelling Salesman problem is a problem in combinatorial optimization (a topic that consists of finding an optimal object from a finite set of objects) studied in operational research and theoretical computer science. In our project, we will use two algorithms which are Branch and Bound algorithm and heuristic technique Genetic Algorithm for solving travelling salesman problem. The comparison between these techniques is accomplished to state which one is better for solving travelling salesman problem. Both the algorithms will be implemented to find out the optimized path and hence compare their time complexities.

In **Genetic Algorithm** parent selection for next generation is very important because solution of problem depends on how much optimized solution derives.Genetic algorithms (GAs) are based essentially on mimicking the survival of the fittest among the species generated by random changes in the gene-structure of the chromosomes in the evolutionary biology.

In **Branch and Bound method**, for the current node in a tree we compute a bound on best possible solution that we can get if we traverse down this node. If the bound on best possible solution itself is worse than current best (best computed so far), then we ignore the sub-tree rooted with the node.

Thus with implementing these algorithms with their key features travelling salesman problem has an efficient and optimized solution.

**TABLE OF CONTENTS**

**S.No. Contents Page No**

1. **Introduction 1**
   1. History 1
   2. Requirement Analysis 2
   3. Main Objective 3
   4. Pert Chart Legend 5
2. **System Analysis 6**
   1. Existing System 6
   2. Motivations 7
   3. Proposed System 7

1. **Design 10**
   1. Modelling 10

3.1.1. Behaviour-based Robotics 10

3.1.2. Use Case model for Requirement Analysis 11

3.1.3. The design model 12

3.1.4. Object and Class Design 13

3.1.5. State Transition 14

3.1.6. Activity Diagram 17

1. **TSP 18**
   1. TSP Overview 18
   2. Basic Stamp –Boe-Bot’s Brain 19

4.2.1.Technical Specifications 20

4.2.2 Features 21

4.2.3 Architecture-memory organization 23

1. **Implementation 34**
   1. Path tracing using a Cartesian Co-ordinate System 34
   2. Scenarios 36
   3. Algorithms 39

5.3.1.Scenario -1 Genetic Algorithm 39

5.3.2.Scenario -2 Branch And Bound 40

1. **Output screens 44**
2. **Limitations and Future Enhancements** 68
3. **Conclusion 69**

**Appendix A: BASIC Stamp and Carrier Board Components 70**

**and Features**

**Appendix B: Resistor Color Codes 72**

**Appendix C: Breadboarding Rules 73**

**References** 76

**LIST OF FIGURES**

**S.No. Figure Page No**

1. **Chapter 1**

Fig. 1.1 Pert Chart 5

1. **Chapter 2**

Fig. 2.1 Servo Motors Overview 8

1. **Chapter 3**

Fig. 3.1 Use Cases Model for the System 11

Fig. 3.2 Sequence Diagram 12

Fig. 3.3 Class Diagram 13

Fig. 3.4 State Transition Diagram for the detecting behavior 14

Fig. 3.5 State Transition Diagram for the finding path behavior 15

Fig. 3.6 State Transition Diagram for the obstacle avoidance behavior 15

Fig. 3.7 State Transition Diagram for the restoring path behavior 16

Fig. 3.8 Activity Diagram for the system 17

1. **Chapter 4**

Fig. 4.1 Boe-Bot 18

Fig. 4.2. Basic Stamp® 2 Module on a Boe-Bot Robot 20

Fig. 4.3 Basic Stamp 2 21

Fig. 4.4 Basic Stamp 2 OEM 22

Fig. 4.5 BS1-IC,BS1 Carrier Board, and BS1 Serial Adapter 25

Fig. 4.6 BS2-IC and Board of Education 26

Fig. 4.7 Test your PC connection to the BASIC Stamp 27

Fig. 4.8 Entering the $STAMP and $PBASIC directives 27

from the toolbar

Fig. 4.9 To run the program,you may use the task bar menu 28

Or the Run icon

Fig. 4.10 Debug Terminal displaying Program Output 29

Fig. 4.11 Parallax Continuous Rotation Servos 30

Fig. 4.12 Object Detection with IR Headlights 31

Fig. 4.13 Infrared Module Parts 32

Fig. 4.14 Ultrasonic Distance Sensor 32

**5. Chapter 5**

Fig. 5.1 Path tracing without obstacles 35

Fig. 5.2 Path tracing with obstacles 36

Fig. 5.3 Scenario-1 Co-ordinate based seeking 37

Fig. 5.4 Scenario-2 Lookup Based Seeking 38

Fig. 5.5 Scenario-3 Serpentine Scanning 38

1. **Chapter 6**

Fig. 6.1 The Cartesian Co-ordinate System 44

Fig. 6.2 Scenario-1 without obstacles (1) 45

Fig. 6.3 Scenario-1 without obstacles (2) 46

Fig. 6.4 Scenario-1 without obstacles (3) 47

Fig. 6.5 Scenario-1 without obstacles (4) 48

Fig. 6.6 Scenario-1 without obstacles (5) 49

Fig. 6.7 Scenario-1 without obstacles (6) 50  
Fig. 6.8 Scenario-1 with obstacles (1) 51

Fig. 6.9 Scenario-1 with obstacles (2) 52

Fig. 6.10 Scenario-1 with obstacles (3) 53

Fig. 6.11 Scenario-1 with obstacles (4) 54

Fig. 6.12 Scenario-1 with obstacles (5) 55

Fig. 6.13 Scenario-1 with obstacles (6) 56

Fig. 6.14 Scenario-1 with obstacles (7) 57

Fig. 6.15 Scenario-1 with obstacles (8) 58

Fig. 6.16 Scenario-2 without obstacles(1) 59

Fig. 6.16 Scenario-2 without obstacles(1) 59

Fig. 6.17 Scenario-2 without obstacles(2) 60

Fig. 6.18 Scenario-2 without obstacles(3) 61

Fig. 6.19 Scenario-2 without obstacles(4) 62

Fig. 6.20 Scenario-2 without obstacles(5) 63

Fig. 6.21 Scenario-3(1) 64

Fig. 6.22 Scenario-3(2) 65

Fig. 6.23 Scenario-3(3) 66

Fig. 6.24 Scenario-3(4) 67

**Appendix-A :BASIC Stamp and Carrier Board**

**Components and Features**

Fig. A-1 BASIC Stamp® 2 Microcontroller Module 70

Components and Their functions

Fig. A-2 Board of Education® Rev C Carrier Board 71

**Appendix-B Resistor Color Codes**

Fig. B-1 Resistor Color codes 72

**Appendix-C Breadboarding Rules**

Fig. C-1 Prototyping Area 73

Fig. C-2 Part Drawings and Schematic Symbols 74

Fig. C-3 Example Schematic and Wiring Diagrams 75

**LIST OF TABLES**

**S.No. Table Page No**

1. **Chapter 4**

Table 4.1 BS2 Features 21

Table 4.2 Basic Stamp 2 Pin Descriptions 23

Table 4.3 RAM organization for all BS2 models 24

1. **Appendix B:Resistor Color Codes**

Table B-1 Resistor Color code values 72

1. **INTRODUCTION**

The traveling salesman problem (TSP) is the most well-known combinatorial optimization problem. TSP is to find a routing of a salesman who starts from home location, visits a prescribed set of cities and returns to the original location in such a way that the total distance travelled is minimal and each city will be visited exactly once.There are mainly two reasons for TSP being an active research area. Firstly, a large number of real world problems can be modeled by TSP. Second, it was proved to be NP-Complete problem. There are many algorithms for solving TSP but we will use Genetic Algorithm which gives the best solutions in reasonable time. It will also provide better way to solve problem in very efficient method.

There are mainly three reasons why TSP has been attracted the attention of many researcher’s and remains an active research area. First, a large number of real-world problems can be modeled by TSP. Second, it was proved to be NP-Complete problem [1]. Third, NP-Complete problems are intractable in the sense that no one has found any really efficient way of solving them for large problem size. Also, NP-complete problems are known to be more or less equivalent to each other; if one knew how to solve one of them one could solve the lot.The TSP finds application in a variety of situations such as automatic drilling of printed circuit boards

and threading of scan cells in a testable VLSI circuit [2], X-ray crystallography [3], etc. The methods that provide the exact optimal solution to the problem are called exact methods. An implicit way of solving the TSP is simply to list all the feasible solutions, evaluate their objective function values and pick out the best. However it is obvious that this “exhaustive search” is grossly inefficient and impracticable because of vast number of possible solutions to the TSP even for problem of moderate size. Since practical applications require solving larger problems, hence emphasis has shifted from the aim of finding exactly optimal solutions to TSP, to the aim of getting, heuristically, ‘good solutions in reasonable time and ‘establishing the degree of goodness’. Genetic algorithm (GA) is one of the best heuristic algorithms that have been used widely to solve the TSP instances.

* 1. **History**

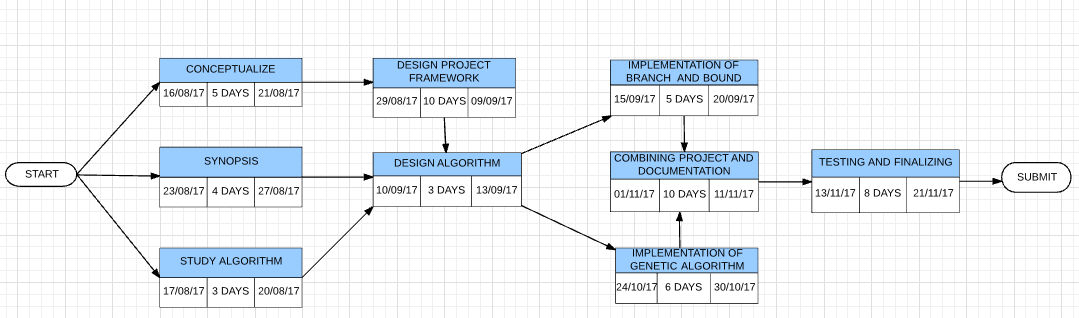
The problem was first formulated as a mathematical problem by Menger (1930) and is one of the most intensively studied problems in optimization. However it was unnoticed till Menger (1994) published a book where he narrated the foundation of mathematical problem for the travelling salesman problem. It is used as a benchmark for many optimization methods.

The origins of the traveling salesman problem are unclear.. Hamilton (1800) and Kirkman (1800) expressed the concept of Mathematical problems related to the travelling salesman problem. The general form of the TSP appears to have been first studied by mathematicians notably by Menger (1930). Further Menger (1930) also defines the problem related with salesman ship based on brute-force algorithm, and observes the non-optimality of the nearest neighbor heuristic. However Whitney (1930) introduced the name travelling salesman problem.

During the period 1950 to 1960, the travelling salesman problem started getting popularity in scientific circle is especially in Europe and the USA. Many researchers like Dantzig, Fulkerson and Johnson (1954) at the RAND Corporation in Santa Monica expressed the problem as an integer linear program and developed the cutting plane method for its solution. With these new methods they solved an instance with 49 cities to optimality by constructing a tour and proving that no other tour could be shorter. In the following decades, the problem was studied by many researchers from mathematics, science, chemistry, physics, and other sciences. Karp (1972) showed that the Hamiltonian cycle problem was NP-complete, which implies the NP-hardness of TSP. This supplied a scientific explanation for the apparent computational difficulty of finding optimal tours.

* 1. **Main Objectives**
* To find out the optimized path for Travelling Salesman Problem by applying the Genetic Algorithm and the Branch and Bound Algorithm.
* To compare the efficiency of both the algorithms in solving TSP.
  1. **Pert Chart Legend**

The project is scheduled to be completed according to the pert chart provided

****

**Fig.1**

1. **SYSTEM ANALYSIS** 
   1. **Existing System**

The Existing system proposed implements some of the much known techniques in computer sciences for solving the graph. As travelling salesman problem is solved by the graph theory methods, following are the proposed solution techniques of the problem:

.

* + 1. **Greedy Approach**

The Greedy Approach can be the first method which can be used to solve TSP. To solve TSP using Greedy Approach, we look at all the arcs coming out of the city (node) and choose the n cheapest arcs. If those n cheapest arcs forms a Hamiltonian cycle than we have an optimal solution.

Using Fig.2, best solution using greedy approach is not calculated because of not returning to the start node. Therefore there is no solution using greedy approach**.**

* + 1. **Nearest Neighbour**

The Nearest Neighbor heuristic is a simple approach for solving the Traveling Salesman Problem. To solve TSP with a Nearest Neighbor heuristic we look at all the arcs coming out of the city (node) that have not been visited and choose the next closest city, then return to the starting city when all the other cities are visited.

Using Fig.2 the best solution by nearest neighbor technique is A-B-D-C-A with the distance of 18.

Thus on comparing all these solution and analyzing them on the basis of time complexity and optimization the following conclusion is made:

* The paths mentioned and the total cost i.e. the distance of the solution is not perfect and the techniques fail for the large no of input cities.

Thus these approaches are not apt for realizing the problem into real life applications and thus need to be calculated on the particular parameterized basis.

Hence these approaches fail to optimize the solution and a new system needs to be proposed.

* 1. **Motivation**

The Design and Analysis of Algorithms has been the base structure for any Computer's application and contemplating it as a course made it all the more clear and started an idea of doing likewise individually and inferring out some imperative highlights of an issue that can be connected, in actuality, that to by utilizing a portion of the methods indicated us by nature and utilizing them as calculations in PCs. Our Mentor motivated us to look on the proposed issue and encouraged to continue for a few procedures conveying forward his exploration.

* 1. **Proposed System**

Evolution and enhancement in the computers makes it necessary for us to be in the race and compete hand to hand. Thus this project proposes some new techniques to efficiently solve the travelling salesman problem which are the real life processes used as the algorithms provided by the nature itself.

It focuses on the heuristic techniques which are expected to show the optimal solution even with the large input of cities which was the failure in the existing system.

* + 1. **Genetic Algorithm**

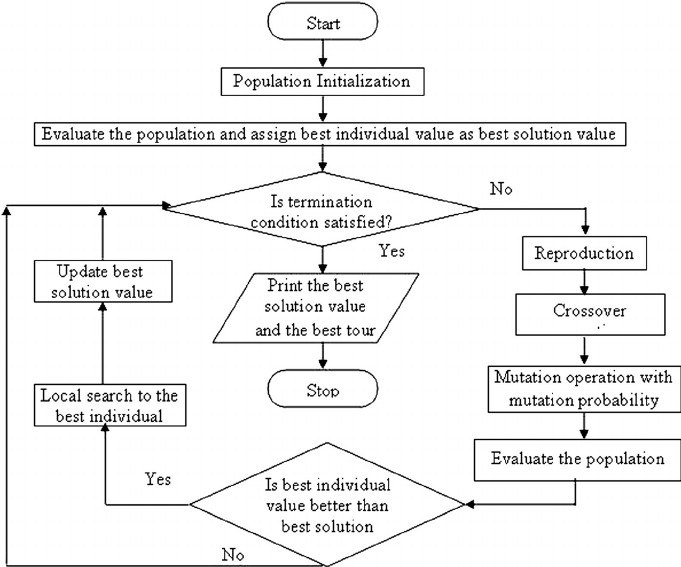
A traditional GA works by randomly generating an initial population of strings, which is referred as gene pool and then applying (possibly three) operators to create new, and hopefully, better populations as successive generations.

The first operator is reproduction where strings are copied to the next generation with some probability based on their objective function value.

The second operator is crossover where randomly selected pairs of strings are mated, creating new strings. [1]

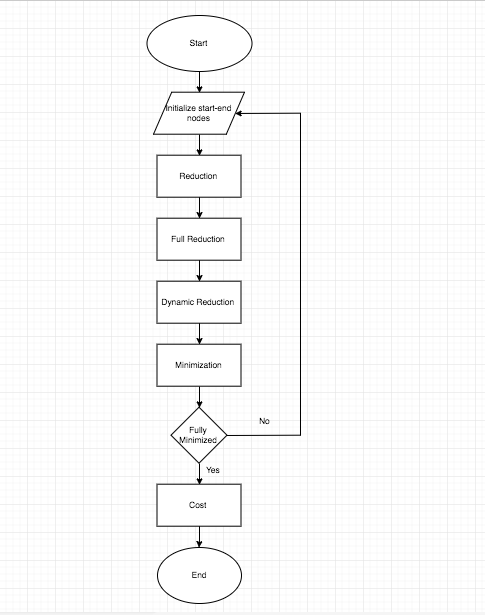
The third operator, mutation, is the occasional random alteration of the value at a string position. The crossover operator together with reproduction is the most powerful process in the GA search. Mutation diversifies the search space and protects from loss of genetic material that can be caused by reproduction and crossover. So, the probability of applying mutation is set very low, whereas the probability of crossover is set very high.

In **Genetic Algorithm** the best selection method RWS and Tournament selection with Elitism will be used. SCX and OX crossover operators and finally mutation will take place to eventually reach the optimum solution.



**Fig.2**

* + 1. **Branch And Bound Method**

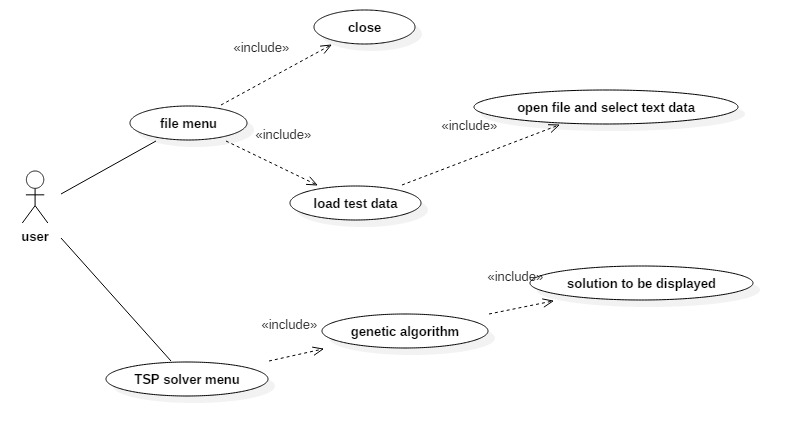
.

**Fig.3**

1. **DESIGN**

Modeling is the essential part of the project design. Different models represent different phases of the project development and provide the detailed study of the project.

* 1. **Use Case Model**

****

**Fig.4**

This model establishes the use cases of the user with the development system

1. **IMPLEMENTATION**
   1. **Genetic Algorithm**

A simple GA works by randomly generating an initial population of strings, which is referred as gene pool and then applying (possibly three) operators to create new, and hopefully, better populations as successive generations. The first operator is reproduction where strings are copied to the next generation with some probability based on their objective function value. The second operator is crossover where randomly selected pairs of strings are mated, creating new strings. The third operator, mutation, is the occasional random alteration of the value at a string position. The crossover operator together with reproduction is the most powerful process in the GA search. Mutation diversifies the search space and protects from loss of genetic material that can be caused by reproduction and crossover. So, the probability of applying mutation is set very low, whereas the probability of crossover is set very high.

****

**Fig.7**

* + 1. **Fitness Function**

Fitness function is the important parameter of a genetic algorithm that defines the fitness of each chromosome where the values of genetic parameters are adapted as the genetic evolution progresses. At every generation, fitness value of each chromosome is calculated using fitness function. If fitness of two chromosomes is equal, then the mutation rate is increased, in order to help the genetic evolution get out of issues like local maxima or local minima whichever is applicable. Once there is an improvement in the overall fitness, the original mutation rate is restored to continue evolution as normal. The random fitness values are generated to initiate the algorithm.

* + 1. **Selection Operator**

Selection is the process of choosing two parents from the population for crossing. The purpose of selection is to emphasize fitter individuals in the population in hopes that their off springs have higher fitness. Chromosomes are selected from the initial population to be parents for reproduction. Selection is a method that randomly picks chromosomes out of the population according to their evaluation function. The higher the fitness function, the more chance an individual has to be selected.

We use **ROULETTE WHEEL SELECTION** method for performing the selection operator.Roulette method selection method works similarly to a roulette wheel, where the likelihood that an individual is chosen is proportional to its fitness value. Because a 0 is considered the ideal fitness for individuals in this algorithm, the size of each individuals „slice‟ of the roulette wheel will be inversely proportional to their fitness value. Once the slices have been determined, a number is generated at random. The individual with the range of numbers that contains this randomly generated number will be one parent. This continues until the desired number of parents is found.

**Algorithm:** void rouletteParents(numParents, myPopulation, parents){

int high = 0;

for each(chromosome : myPopulation)

if(chromosome.fitness > high)

high = chromosome.fitness;

create vector of integers = size of myPopulation;

for(0 <= i < myPopulation.size())

for(0 <= j < high – myPopulation.elementAt(i).fitness)

add i to vector of integers;

for(0 <= i < numParents)

shuffle vector of integers;

parents.add(myPopulation.elementAt(vector[i]); }

* + 1. **Crossover Operator**

After the completion of the selection process, the chromosomes chosen to be parents for the next generation are recombined to form children that are new chromosomes. This combination can take many forms and the crossover operator can greatly affects the result of the search.

We use the **SEQUENTIAL CONSTRUCTIVE CROSSOVER (SCX)**

The sequential constructive crossover (SCX) operator constructs an offspring using better edges on the basis of their values present in the parents' structure. It also uses the better edges, which are present neither in the parents' structure. As the ERX and GNX, the SCX does not depend only on the parents' structure; it sometimes introduces new, but good, edges to the offspring, which are not even present in the present population. Hence, the chances of producing a better offspring are more than those of ERX and GNX. A preliminary version of the operator is reported as local improvement technique.

* + 1. **Mutation Operator**

The operation of mutation allow new individual to be created. It begins by selecting an individual from the population based on its fitness. A point long the string is selected at random and the character at that point is randomly changed, the alternate individual is then copied in to the next generation of the population. Mutation is performed after crossover by randomly choosing a chromosome in the new generation to mutate. We randomly choose a point to mutate and switch that point. Many types of mutation operators exist. Here we are using the bit flip method which is only used with a binary chromosome representation, that changes a particular point on the chromosome to its opposite.

This is an optional operator for our project as we used the scx crossover hence mutation becomes optional.

* + 1. **Control Parameters**

These are the parameters that govern the GA search process. Some of them are:

* + - Population size: - It 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.
    - Crossover probability: - It specifies the probability of crossover occurring between two chromosomes.
    - Mutation probability: - It specifies the probability of doing bit-wise mutation.
    - Termination criteria: - It specifies when to terminate the genetic search.
  1. **Branch And Bound Method**

In Branch and Bound method, for the current node in a tree we compute a bound on best possible solution that we can get if we traverse down this node. If the bound on best possible solution itself is worse than current best (best computed so far), then we ignore the sub-tree rooted with the node.

Note that the cost through a node includes two costs.

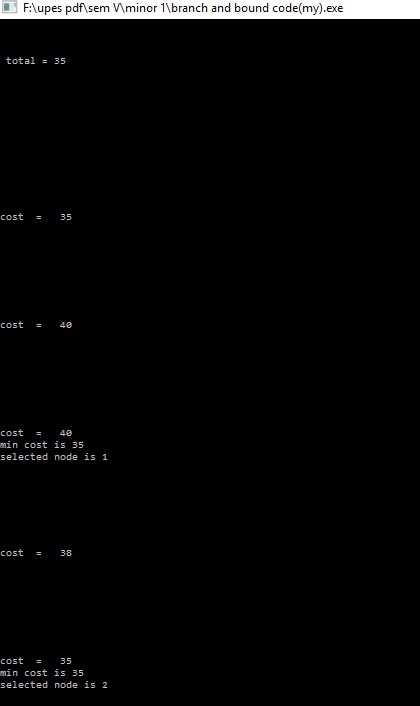
1) Cost of reaching the node from the root (when we reach a node, we have this cost computed)  
2) Cost of reaching an answer from current node to a leaf (we compute a bound on this cost to decide whether to ignore sub-tree with this node or not).

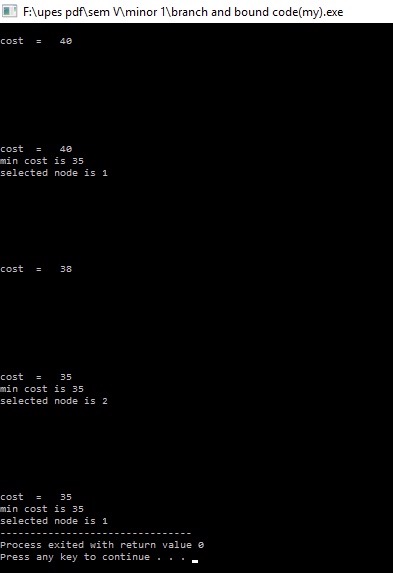
* In cases of a **maximization problem**, an upper bound tells us the maximum possible solution if we follow the given node.
* In cases of a **minimization problem**, a lower bound tells us the minimum possible solution if we follow the given node.

**Algorithm:**

* Step 1: Start
* Step 2: Generate adjacency matrix of the given inputs.
* Step 3: We reduce the minimum value in each row from each element in that row.
* Step 4: We then reduce the minimum value in each column from each element in that column.
* Step 5: The total expected cost at the root node is the sum of all reductions.
* Step 6: As we are adding edge(0,1) to our search space , we set outgoing edges for city 0 to infinity and all incoming edges to city 1 to infinity. We also set (1,0) to infinity.
* Step 7: We calculate lower bound of the path starting node 1 using above resulting cost matrix.
* Step 8: Repeat from Step 6 to find out the optimum cost for remaining all nodes.
* Step 9: The minimum cost among them is selected and is the final cost.
* Step 10: Stop.

1. **OUTPUT SCREENS**
   1. **Output Screen for Genetic Algorithm**
   2. **Output Screens for Branch And Bound**





1. **LIMITATIONS AND FUTURE ENHANCEMENTS**
   1. **Limitations**

The results of the calculations are the most ideal and ideal in setting to the distinction between them. There might be possible of the presence of some other system which could deliver stunningly better outcomes.

* 1. **Future Enhancements**

1. **CONCLUSION**
2. **REFERENCES**
3. Chetan Chudasama, S M Shah and Mahesh Panchal. Article: Comparison of Parents Selection Methods of Genetic Algorithm for TSP. *COMNET-2011* 102-105, 2011.
4. S.N. Sivananadam, S.N. Deepa, Introduction to Genetic Algorithm, Springer- Verlag Berlin Heidelberg 2008.
5. CS.indstate.edu/cpothineni/alg.pdf December 13, 2013
6. https://en.wikipedia.org/wiki/Mutation\_(genetic\_algorithm)
7. <https://en.wikipedia.org/wiki/Crossover_(genetic_algorithm)>
8. <https://en.wikipedia.org/wiki/Travelling_salesman_problem>
9. <http://lcm.csa.iisc.ernet.in/dsa/node187.html>
10. <http://www.geeksforgeeks.org/travelling-salesman-problem-set-1/>
11. M. Mataija, M. Rakamarić Šegić, F. Jozić: Solving the Travelling Salesman Problem using the Branch and... Zbornik Veleučilišta u Rijeci, Vol. 4 (2016.), No. 1, pp. 259-270.
12. <https://www.youtube.com/watch?v=nYCU5c1miUw>