IMPLEMENTATION OF ACO ALGORITHM FOR EDGE DETECTION AND SORTING SALESMAN PROBLEM

Er. Priya Darshni

Assiociate Prof. ECE Deptt.
Ludhiana Chandigarh highway
Ludhiana College Of Engg. And Technology
Katani Kalan, Ludhiana,Punjab, INDIA
Punjab Technical University Jalandhar (Pb.) INDIA
Email:priyabatra_80@yahoo.co.in

Gurpreet Kaur

Student ECE deptt.
Ludhiana Chandigarh highway
Ludhiana College Of Engg. And Technology
Katani Kalan, Ludhiana,Punjab, INDIA
Punjab Technical University Jalandhar (Pb.) INDIA
Email: kaur.dhindsa@gmail.com

Abstract

This paper presents an Ant Colony Optimization-based technique for image edge detection and solve TSP.The proposed method establishes a pheromone matrix that represents the edge information at each pixel based on the routes formed by ants dispatched on the image. The movement of the ants is guided by the local variation of the image's intensity values.In TSP, a set of cities is given and distance between each of them is known. The goal is to find the shortest tour that allows each city to be visited once and only once. ACO is a population-based metaheuristic that mimics the foraging behaviour of ants to find approximate solutions to difficult optimization problems. In ACO, the problem is tackled by simulating a number of artificial ants moving on a graph that encodes the problem itself.

Keywords: ACO, TSP, metaheuristic, Pheromone and ACS.

1. INRODUCTION

1.1 ACO (Ant Colony Optimization)

The **ant colony optimization** algorithm **(ACO)**[3] is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. This algorithm is a member of **ant colony algorithms** family, in swarm intelligence methods, and it constitutes some metaheuristic [4]optimizations.

- Ant Colony Optimization (ACO) is a population-based, general search technique for the solution of difficult combinatorial problems which is inspired by the pheromone trail laying behaviour of real ant colonies. The behaviour of ant is exploited in artificial ant colonies for the search of approximate solutions to discrete optimization problems, to continuous optimization problems, and to important problems in telecommunications, such as routing and load balancing. Initially proposed by Marco Dorigo in 1992 in his PhD thesis, the first algorithm was aiming to search for an optimal path in a graph, based on the behaviour of ants seeking a path between their colony and a source of food.

- The ant colony optimization (ACO) meta-heuristic a colony of artificial ants cooperate in finding good solutions to difficult discrete optimization problems. Cooperation is a key design component of ACO algorithms: The choice is to allocate the computational resources to a set of relatively simple agents (artificial ants) that communicate indirectly by stigmergy. Good solutions are an emergent property of the agents' cooperative interaction.
- The original idea has since diversified to solve a wider class of numerical problems, and as a result, several problems have emerged, drawing on various aspects of the behaviour of ants. The main underlying idea, loosely inspired by the behaviour of real ants, is that of a parallel search over several constructive computational threads based on local problem data and on a dynamic memory structure containing information on the quality of previously obtained result. The collective behaviour emerging from the interaction of the different search threads has proved effective in solving combinatorial optimization (CO) problems.

1.2 AS (Ant system algorithm):

- Ant System (AS) [5]was the first (1991) ACO algorithm. Its importance resides mainly in being the prototype of a number of ant algorithms which have found many interesting and successful applications.
- Three AS algorithms have been defined, which differ by the way pheromone trails are updated. These algorithms are called ant-density, ant-quantity, and ant-cycle. In ant-density and ant-quantity ants deposit pheromone while building a solution, while in ant-cycle ants deposit pheromone after they have built a complete tour.

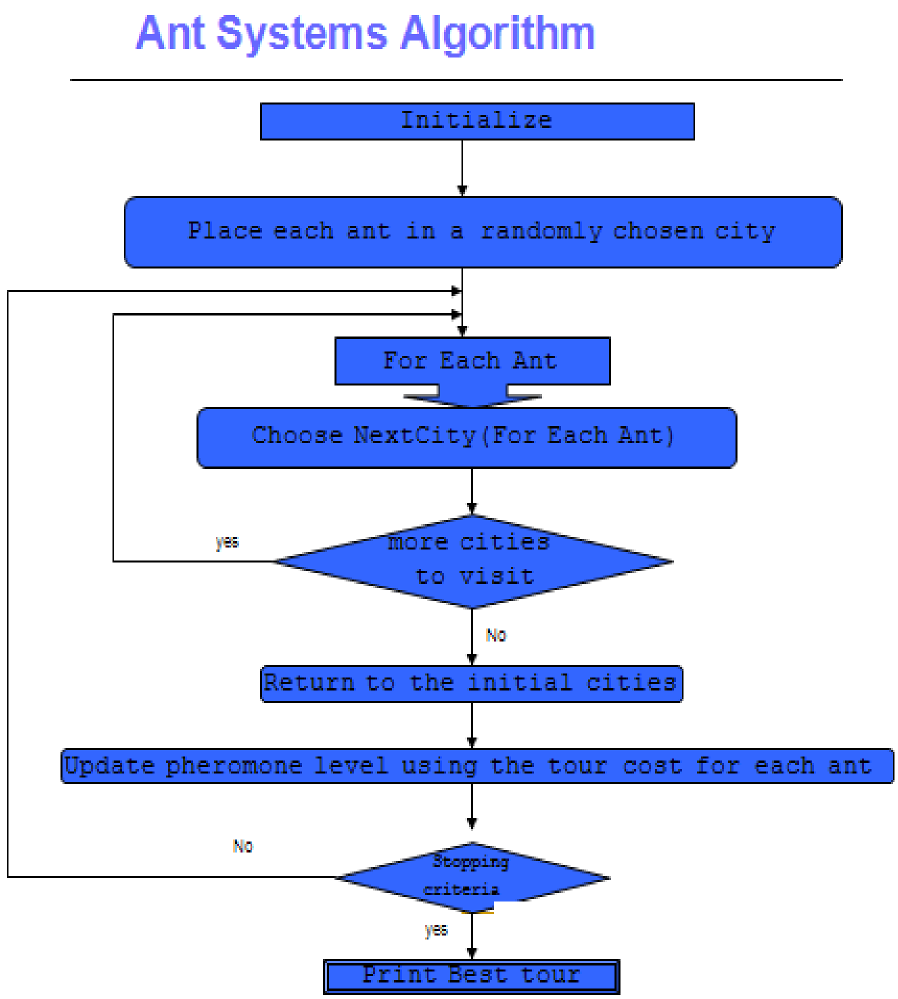

Figure 1:- Ant Systems Algorithm

1.3 ACS (Ant Colony System):

ACS was the first algorithm inspired by real ants behaviour. The merit to introduce ACO algorithms and to show the potentiality of using artificial pheromone and artificial ants to drive the search of always better solutions for complex optimization problems.

- **Pheromone:**
  In ACS once all ants have computed their tour (i.e. at the end of each iteration) AS updates the pheromone trail using all the solutions produced by the ant colony. Each edge belonging to one of the computed solutions is modified by an amount of pheromone[1] proportional to its solution value. At the end of this phase the pheromone of the entire system evaporates and the process of construction and update is iterated. On the contrary, in ACS only the best solution computed since the beginning of the computation is used to globally update the pheromone.
  As was the case in AS, global updating is intended to increase the attractiveness of promising route but ACS mechanism is more effective since it avoids long convergence time by directly concentrate the search in a neighbourhood of the best tour found up to the current iteration of the algorithm.

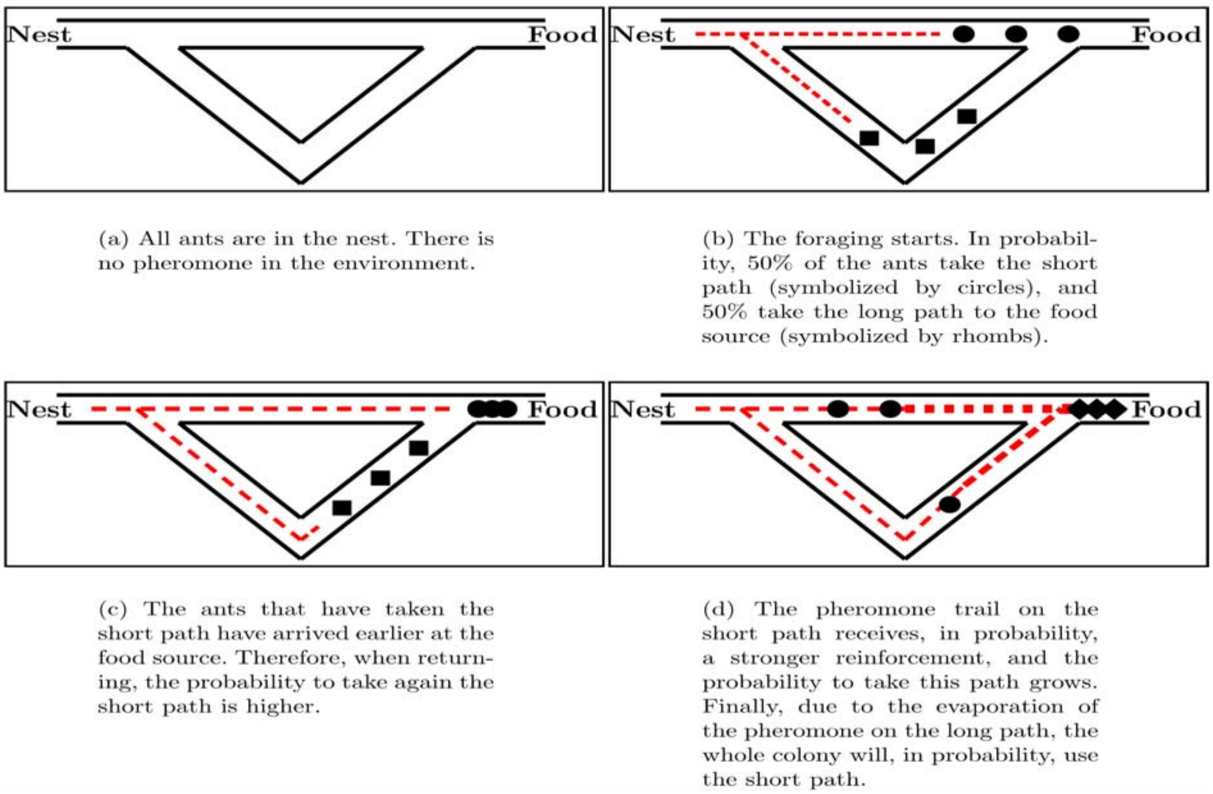

Figure 2.: An experimental setting that demonstrates the shortest path finding capability of ant colonies. Between the ants' nest and the only food source exist two paths of different lengths. In the four graphics, the pheromone trails are shown as dashed lines whose thickness indicates the trails strength.

1.ANTS algorithm within the ACO framework has two mechanisms:

i. Attractiveness
ii. Trail update

- The attractiveness of a move can be effectively estimated by means of lower bounds (upper bounds in the case of maximization problems) on the cost of the completion of a partial solution. In fact, if a state ι corresponds to a partial problem solution it is possible to compute a lower bound on the cost of a complete solution containing ι.
- A good trail updating mechanism avoids stagnation, the undesirable situation in which all ants repeatedly construct the same solutions making any further exploration in the search process impossible. Stagnation derives from an excessive trail level on the moves of one solution, and can be observed in advanced phases of the search process, if parameters are not well tuned to the problem. The trail updating procedure evaluates each solution against the last k solutions globally constructed by ANTS. As soon as k solutions are available, their moving average z is computed; each new solution z is compared with z (and then used to compute the new moving average value). If z is lower than z , the trail level of the last solution's moves is increased, otherwise it is decreased.

$$\Delta\tau_{i,j} = \tau_0 \cdot \left(1 - \frac{z\ curr - LB}{z - LB} \right)$$

where z is the average of the last k solutions and LB is a lower bound on the optimal problem solution cost.

2. ACO algorithm for Edge detection

Edges are points where there is a boundary (or an edge) between two image regions. In general, an edge can be of almost arbitrary shape, and may include junctions. In practice, edges are usually defined as sets of points in the image which have a strong gradient magnitude. Furthermore, some common algorithms will then chain high gradient points together to form a more complete description of an edge. These algorithms usually place some constraints on the properties of an edge, such as shape, smoothness, and gradient value. Edge detection[2] is a technique for marking sharp intensity changes, and is important in further analyzing image content. However, traditional edge detection approaches always result in broken pieces, possibly the loss of some important edges.

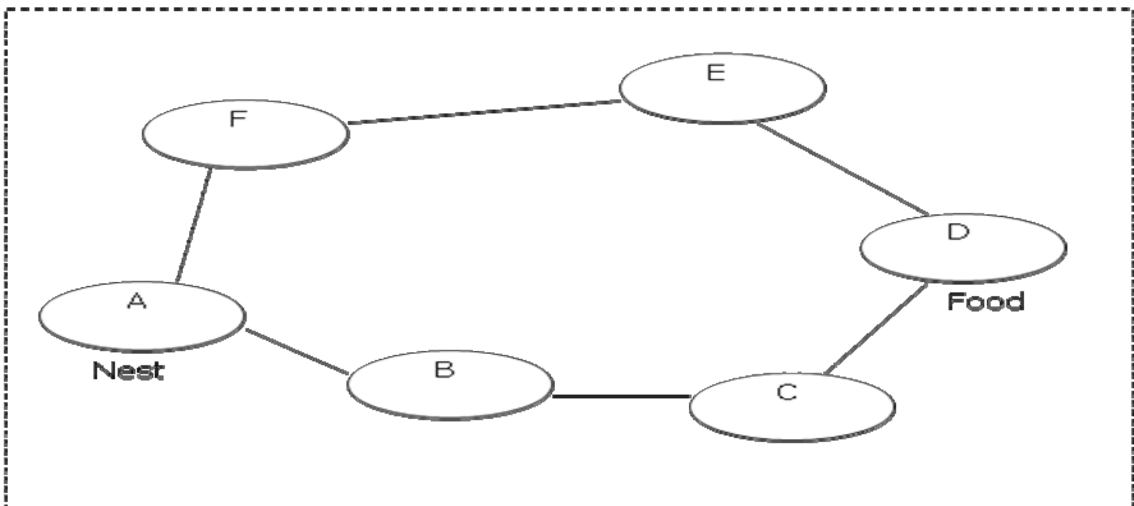

Figure 3.: Ants will start from A the nest and look for D the food. At every step, they will upgrade the routing tables and as soon as the first one reaches the food, the best path will be known, thus allowing communication from D to A.

A method of extracting desired features from a cellular image including the steps of: (a) selecting an initial cell within
the image; (b) selecting an additional cell, near the initial cell, appearing to be associated with a desired feature; (c) repeating step for further cells, near at least one of the previously selected cells, appearing to be associated with said feature, until selection termination criteria are satisfied; and (d) repeating steps through for other initial cells. The method is particularly adept at extracting relatively weakly defined features in relatively noisy images, such as extracting faults or geologic horizons from 2D or 3D seismic data. A method of editing/filtering the features utilising a stereo net is also disclosed. Related computer system and computer program products for implementing the method are also described. The method is particularly adept at extracting relatively weakly defined features from relatively noisy images, such as extracting faults or geologic horizons from 2D or 3D seismic data. A related method of editing/filtering the features utilising a stereo net is also disclosed. The invention further includes a computer system and computer program product for implementing the method. The invention and its benefits will be better understood with reference to the detailed description below and the accompanying drawings.

3.How does this relate to extrema line and surface extraction?

In this application, it will be cleared how these lines and surfaces can be found by the behaviour and interaction of intelligent agents. Earlier in this application, it was concluded that in order to overcome the challenges the relatively noisy data presents, prior knowledge of the desired feature needs to be taken into consideration. This knowledge is coded as the behaviour of an intelligent agent. The agent will act very similar to an ant in the foraging situation described above, by making decisions based on its pre-coded behaviour and emitting “electronic pheromone” along its trail. The idea is to distribute a number of agents in the 2D or 3D image and let each agent move along an extrema ridge while emitting pheromone as long as the ridge fulfils the criteria encoded in the agent. Agents which are deployed at a point where there is no extrema ridge, or where the ridge is poorly defined, will be terminated shortly or immediately after their deployment, whereas agents deployed at points on a well-defined ridge will be able to follow this ridge for a while before being terminated. It is assumed that a line or surface having the properties of the desired structures, which are captured in the prior knowledge, will be clearly marked by pheromone whereas non-extrema ridges, or extrema ridges not fulfilling the requirements will be unmarked or only weakly marked.

$$z\,(\text{pixel})=\omega_1\,\text{chk grey(pixel)}+\omega_2\,\text{chk width(pixel)}+\omega_3\,\text{chk maxwidth(pixel)}+\omega_4\ \ \text{pheromone(pixel)}$$

where pixel is the pixel that is currently being considered; the chkgrey, chkwidth and chkmaxwidth functions evaluate the properties listed above; and ω_i the weight of each function. The outputs of these sub-functions depend on the grey-value and width at the point where the agent is deployed. The last term in the function, pheromone(pixel), provides communication between the agents in terms of the “pheromone” trace each agent emits along its trace.

Edge Selection

An ant will move from node *i* to node *j* with probability

$$p_{i,j} = \frac{(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}{\sum(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}$$

where

$\tau_{i,j}$ is the amount of pheromone on edge *i*,*j*

α is a parameter to control the influence of $\tau_{i,j}$

$\eta_{i,j}$ is the desirability of edge *i*,*j* (a priori knowledge, typically 1 / $d_{i,j}$, where d is the distance)

β is a parameter to control the influence of $\eta_{i,j}$

Variables involved in the selection of an edge detection operator include:

- **Edge orientation:** The geometry of the operator determines a characteristic direction in which it is most sensitive to edges. Operators can be optimized to look for horizontal, vertical, or diagonal edges.
- **Noise environment:** Edge detection is difficult in noisy images, since both the noise and the edges contain high-frequency content. Attempts to reduce the noise result in blurred and distorted edges. Operators used on noisy images are typically larger in scope, so they can average enough data to discount localized noisy pixels. This results in less accurate localization of the detected edges.
- **Edge structure:** Not all edges involve a step change in intensity. Effects such as refraction or poor focus can result in objects with boundaries defined by a gradual change in intensity. The operator needs to be chosen to be responsive to such a gradual change in those cases. Newer wavelet-based techniques actually characterize the nature of the transition for each edge in order to distinguish, for example, edges associated with hair from edges associated with a face.

4.ACO algorithm for TS problem:

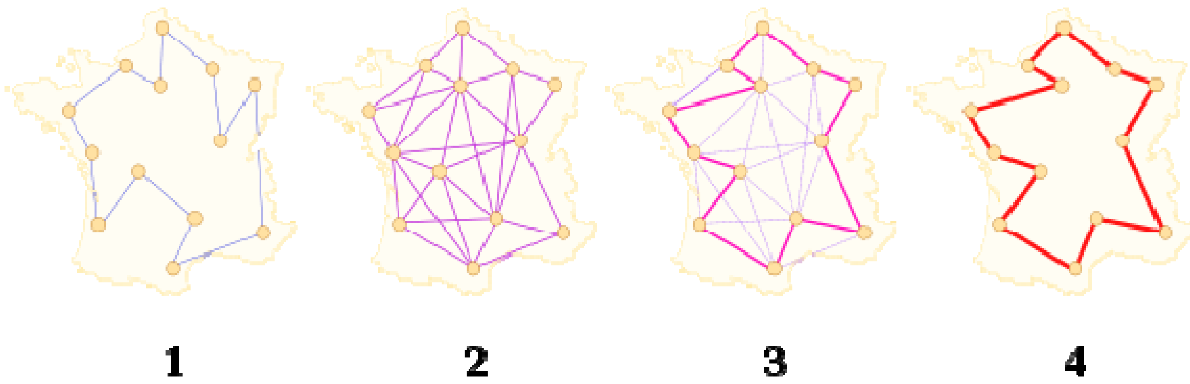

Figure 4:- Travelling Salesman example

Ant Colony Optimization algorithms have been used to produce near-optimal solutions to the travelling salesman problem[6]. The first ACO algorithm was called the Ant system and it was aimed to solve the travelling salesman problem, in which the goal is to find the shortest round-trip to link a series of cities. The general algorithm is relatively simple and based on a set of ants, each making one of the possible round-trips along the cities. In the traveling salesman problem, a set of cities is given and the distance between each of them is known. The goal is to find the shortest tour that allows each city to be visited once and only once At each stage, the ant chooses to move from one city to another according to some rules:

- It must visit each city exactly once;
- A distant city has less chance of being chosen (the visibility);
- The more intense the pheromone trail laid out on an edge between two cities, the greater the probability that that edge will be chosen;
- Having completed its journey, the ant deposits more pheromones on all edges it traversed, if the journey is short

The general Ant colony algorithm in optimization problem described as following rules:

- Initialization
- Define Phoremone of every move
- Phoremone updating rule
- Ants moving rule
- Stopping rule

1.Initialization:
In TSP, suppose *N* is a set of cities , and *E* is also a set connecting edges of two cities. Define a set of paths is
w,feasible soloutions of TSP, which connecting an initiative city and terminal city through a series of interim city by edges. Ants are placed in a cities arbitrarily. In this problem, every ant will choose a path of *w* according to rules. Numbering cities from 1 to *N* if there are *N* cities. Define (i, j) is an edge of city i and city j.
2.Define Phoremone of every move:
The pheromone's quqntity of each ant is based on optimization problem. Every ant is a simple agent to travel through cities. According to probobolity, an ant chooses one city to move into. This probibility is a

function of citie's distance and edge's sum pheromone.Every ant has a recording citites which ant has been accessed.

3.Phoremone Updating rule

Ants leave their pheromone on edges at their every travelling when ants complete one iteration. The sum pheromone of one edge is defined as following:

$\tau_{i,j}(t + 1) = \Delta\tau_{i,j}(t) + (1 - \rho)\, \tau_{i,j}(t)$

$(1 - \rho)$ is persistance rate of previous pheromone . ρ is defined as evaporation of pheromone.

4.Ants moving rule

Ants move from one city to another city according to probibility.Cities accessed must be placed in table which defines a set of cities never access of *k*th ant as allowed allowed$_{k}$.

5.Stopping rule:

There are many conditions for ants to stop their travelling. Such as number of limitations iterations, CPU time limitations or best solutions From the above description, wre can get details procedure of ant colony system algorithms like ant-cycle algorithm, ant-density algorithm, ant-quantity algorithm.

5. RESULTS & DISCUSSION

- **Steps of compilation:**

1.For image edge detection:

i. Implementation of ACO
ii. Implement image Edge detection
iii. Implementation of algorithm
iv. Implementation of ACO algorithm
v. Implementation of Edge detection algorithm
vi. Result form for Edge detection of an image using ACO

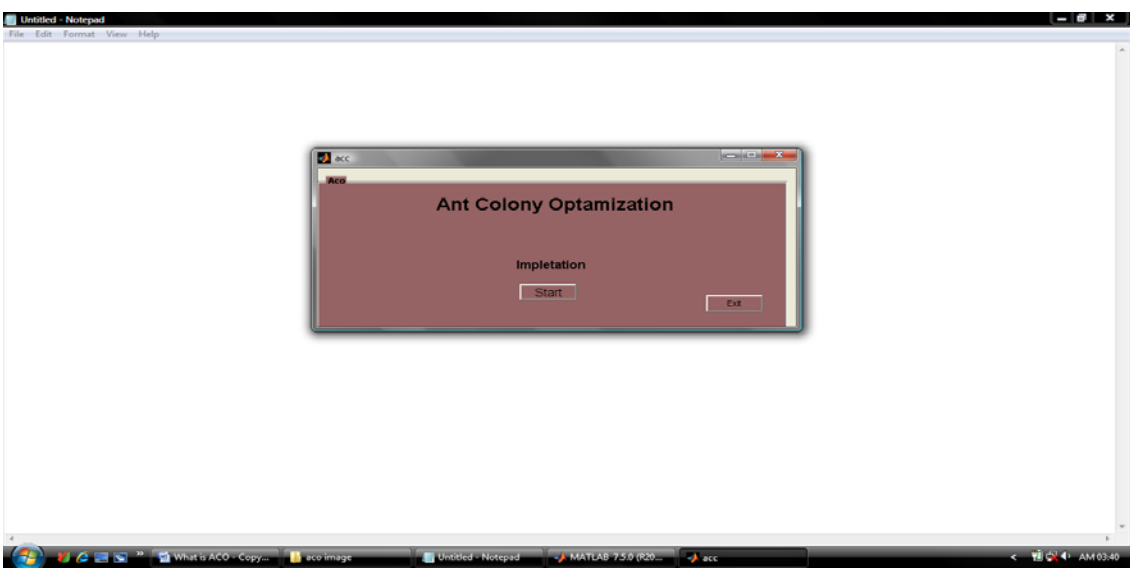

Figure 5: Implementation of ACO

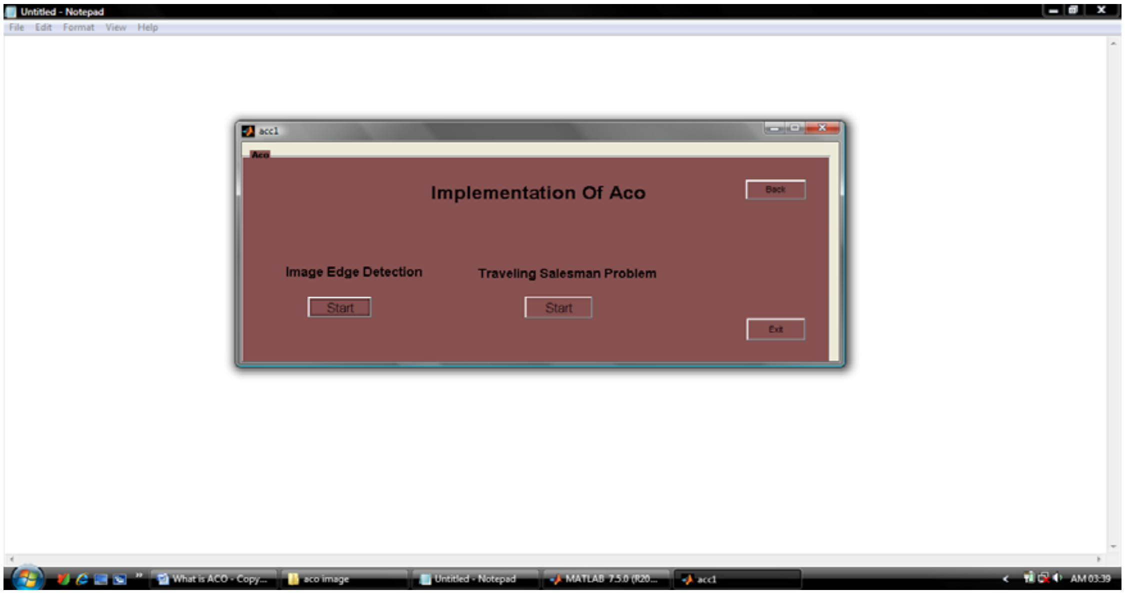

Figure 6: Implementation of Image Edge detection

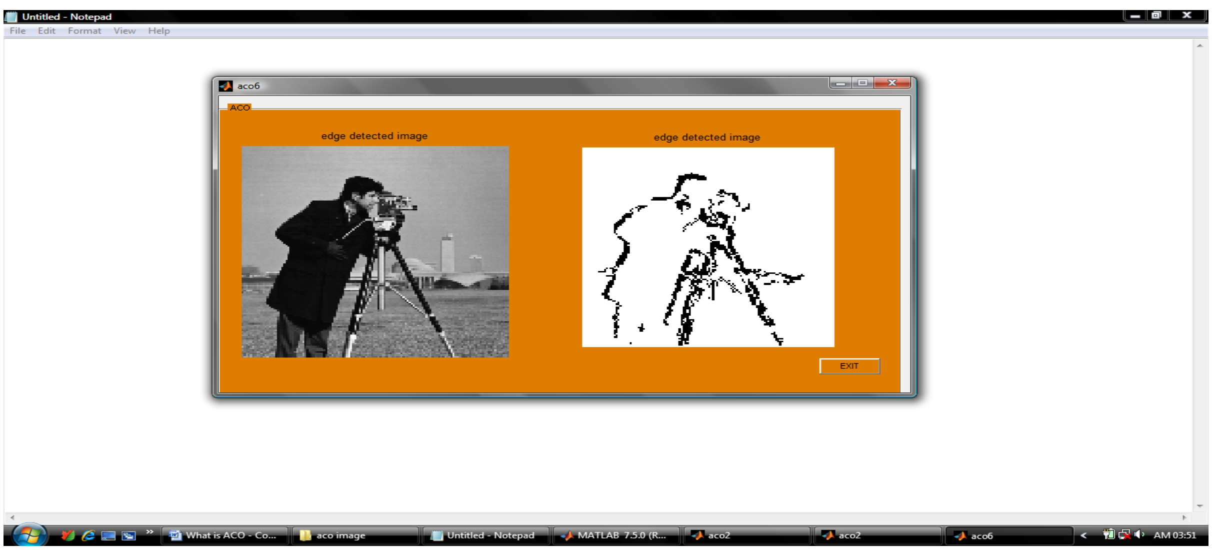

Figure 7: Final output for edge detection of an image

2. For travelling salesman problem:

i. Implementation of ACO
ii. Implement Travelling Salesman Problem algorithm
iii. Result form for Travelling Salesman Problem

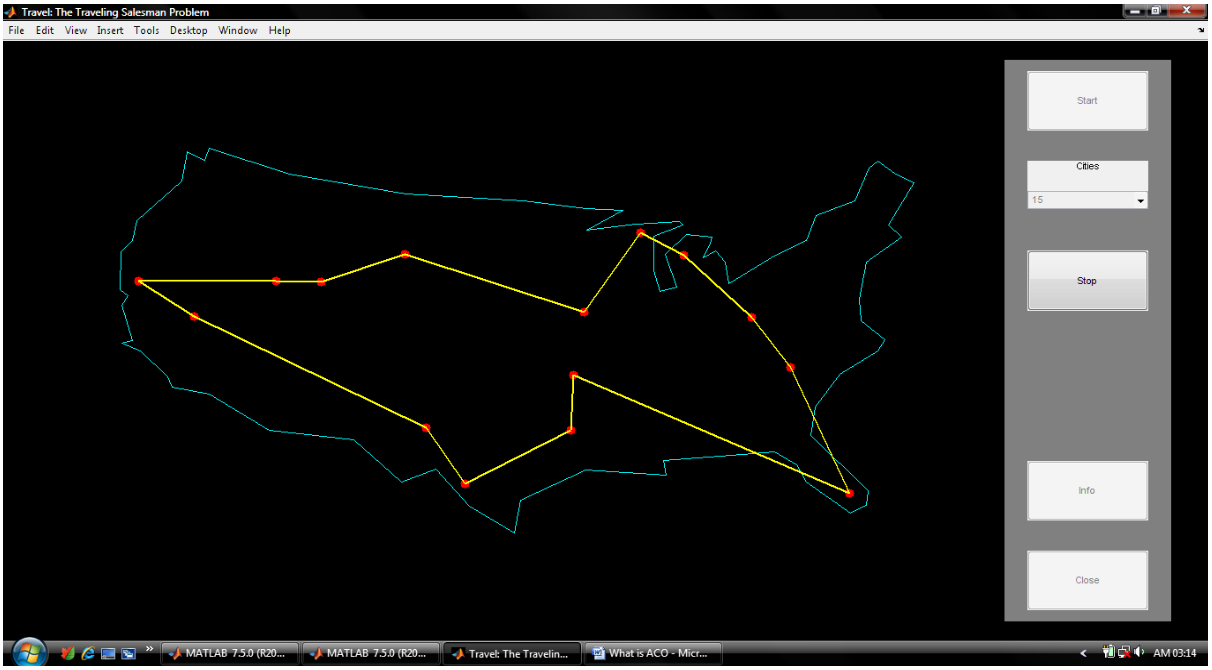

Figure 8: TSP output for 15 cities

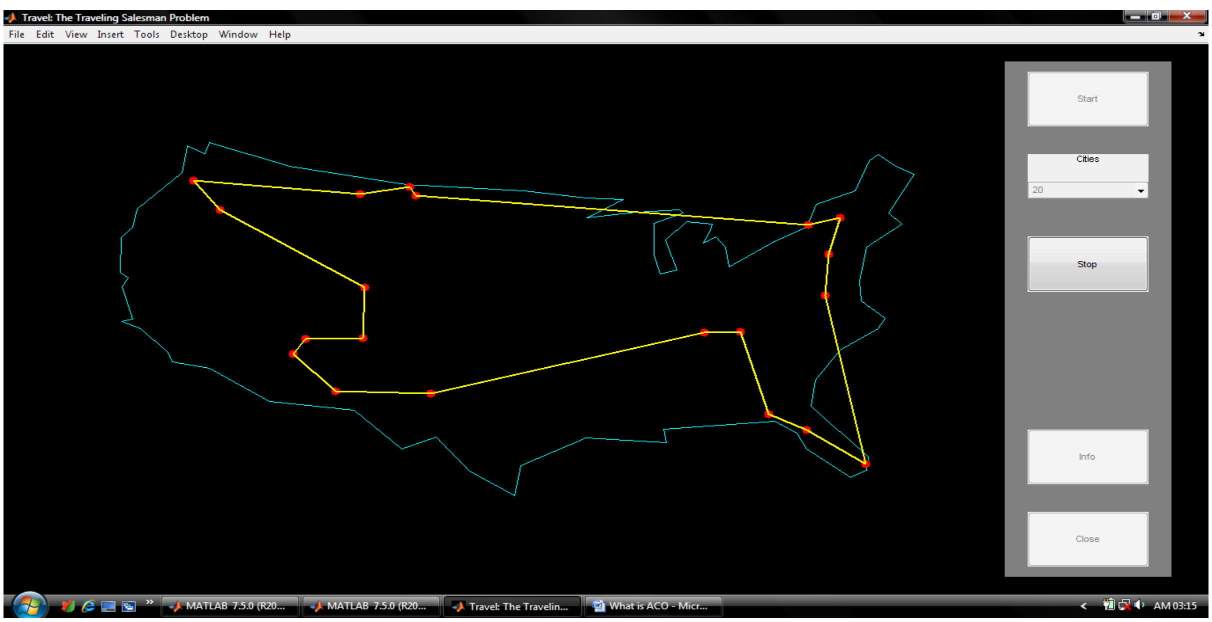

Figure 9: TSP output for 20 cities

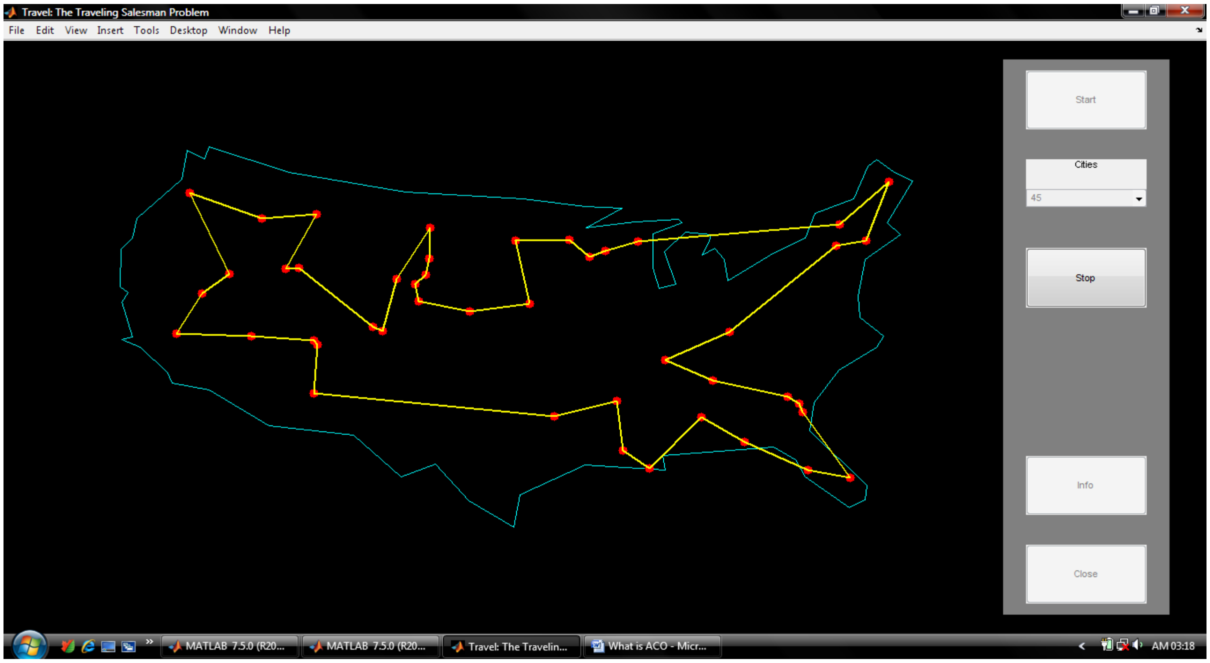

Figure 10: TSP output for 45 cities

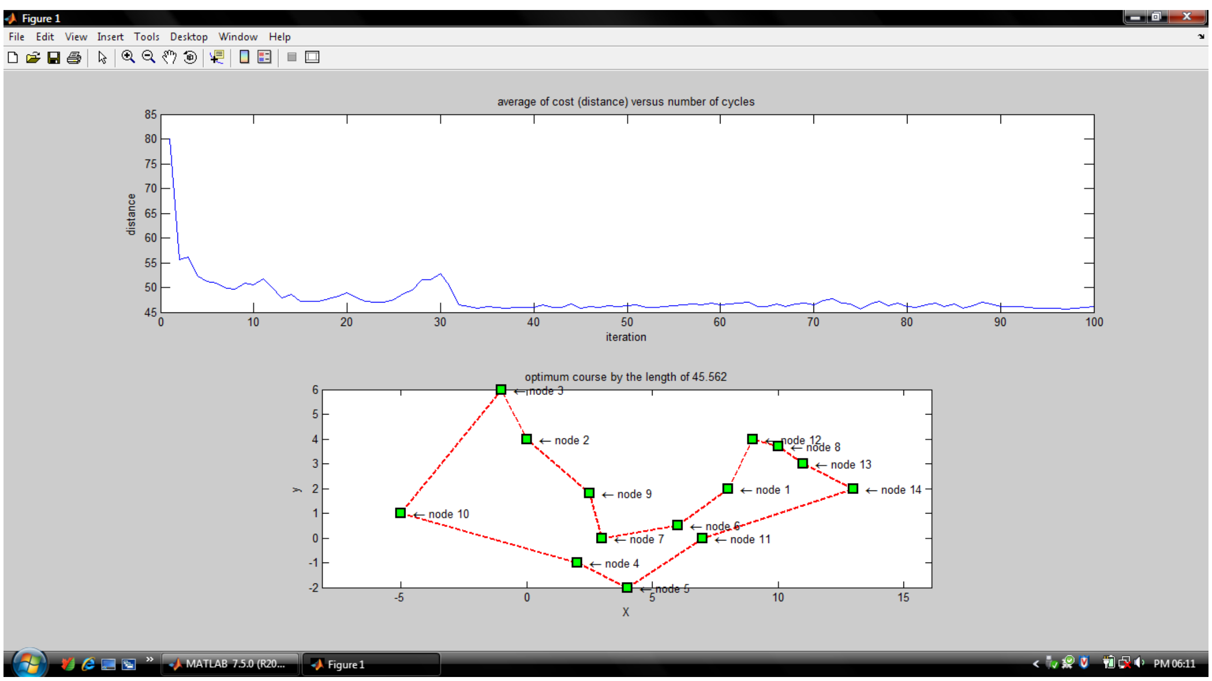

Figure 11: Final TSP output for different cities

6. Conclusion:

This section includes the conclusion on the basis of information gathered and the test results. This project implements the complete ACO algorithm for edge detection of the image. Along this, the project also implement for sorting salesman problem.ACO method solves the difficult computational problems and chooses shortest path to solve the problem.

References:

[1] Dorigo M, Blum C. Ant colony optimization theory: A survey. Theoret Comput Sci 2005;344(2–3):243–78.
[2] Dorigo, Di Caro, Gambardella 1999.
[3] M. Dorigo, Ant colony optimization web page,http://iridia.ulb.ac.be/mdorigo/ACO/ACO.html.
[4] M. Dorigo and G. Di Caro, "The Ant Colony Optimization meta-heuristic", in New Ideas in Optimization, D. Corne et al., Eds., McGraw Hill,London, UK, pp. 11-32, 1999.
[5] Maniezzo V, Colorni A. The Ant System applied to the quadratic assignment problem. IEEE Trans Data Knowledge Engrg 1999;11(5):769–78.
[6] M. Dorigo and L.M. Gambardella, "Ant colonies for the traveling salesman problem,".