# Proposal of Backtracked Ant System to Solve 4-Color Graph Problem

Dr. Ahmed T. Sadiq, Amjed Abbas Ahmed, Ashty M. Aaref

***Abstract*--** This paper presents a new ant system approach to solve 4-color graph problem. This approach called backtracked ant system because it is based on the backtracking strategy in artificial intelligent. 4-color graph problem solved using several backtracked ant systems (Ant System (AS), Ant Colony Optimization (ACO), Elitist AS, Rank-Based AS and Max-Min AS). The experimental results appear that the Backtracked ACO gives the best results compare with the other types.

***Keywords*--** Ant System, 4-Color Graph Problem. ACO.

## I. Introduction

SWARM Intelligence [**1**] is a property of systems of unintelligent agents of limited individual capabilities exhibiting collectively intelligent behavior. An agent in this definition represents an entity capable of sensing its environment and undertaking simple processing of environmental observations in order to perform an action chosen from those available to it. These actions include modification of the environment in which the agent operates. Intelligent behavior frequently arises through indirect communication between the agents, this being the principle of stigmergy [**2**]. Individual ants are behaviorally simple insects with limited memory and exhibiting activity that has a stochastic component. However, collectively ants manage to perform several complicated tasks with a high degree of consistency [**3, 4**].

## II. Ant System Algorithms

Ant algorithms represent a relatively new heuristic search technique that has been successfully applied to solving NP hard problems [**5**]. Perhaps not surprisingly ant algorithms are biologically inspired from the behavior of colonies of real ants, and in particular how they forage for food. One of the main ideas behind this approach is that the ants can communicate with one another wholly through indirect means by making modifications to the pheromone level in their immediate environment.

Dr. Ahmed T. Sadiq is with Computer Science Department, University of Technology, Baghdad, Iraq.

Amjed Abbas Ahmed is with Post Graduate School Department, Master Computer Science in IT, IT Management Department, Binary University College, Malaysia. Email: aliraqi.amjed@yahoo.com

Ashty M. Aaref is with Computer Science Department, Kirkuk Technical College, Kirkuk, Iraq

Ant Colony Optimization (ACO) is the so-called meta-heuristic for ant algorithms applied to optimization problems and as these are the problems we're generally working with we tend to use the terms interchangeably.

The study of Deneubourg et al. [**6**] found that the ants use pheromone trails to communicate information amongst themselves. Though any single ant moves essentially at random, it will make a decision on its direction based on the "strength" of the pheromone on the paths that lie before it. As an ant traverses a path, it reinforces that path with its own pheromone.

Therefore the "shortest" paths will maintain a higher amount of pheromone as opposed to the “longer” paths. A collective autocatalytic [**7**] behavior emerges as more ants will choose the shortest trails (because the short trails have a higher amount of pheromone), which in turn creates an even larger amount of pheromone on those short trails. This means that those short trails will be even more likely to be chosen by future ants.

There are five types of ant systems and the most known types are like the following [**8**]:

### *2.1 Ant System*

The two main phases of the (AS) algorithm constitute the ants solution construction and the pheromone update. In (AS) a good heuristic to initialize the pheromone trails is to set them to a value slightly higher than the expected amount of pheromone deposited by the ants in one iteration; a rough estimate of this value can be obtained by setting, $(\forall (i,j),\ \tau_{ij} = \tau_o = m/C^{nn})$ where ($m$) is the number of ants, and ($C^{nn}$) is the length of a tour generated by the nearest-neighbor heuristic, (In fact, any other reasonable tour construction procedure would work fine).

#### 2.1.1 Tour Construction

In (AS), ($m$) (artificial) ants concurrently build a tour of the traveling ant. Initially, ants are put on randomly chosen cities. In particular, the probability with which ant ($k$), currently at city ($i$), chooses to go to city ($j$) is:

$$P_{ij}^{k} = \frac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k}[\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}} \text{ If: } j \in N_i^k \qquad (1)$$

2.1.2 Update Of Pheromone Trails

After all the ants have constructed their tours, the pheromone trails are updated. This is done by first lowering the pheromone value on all arcs by a constant factor, and then adding pheromone to the arcs the ants have crossed in their tours. Pheromone evaporation is implemented by:

$$\tau_{ij} \leftarrow (1-\rho)\,\tau_{ij}, \quad \forall (i,j) \in L, \tag{2}$$

Where; $(0 < \rho \leq 1)$ is the pheromone evaporation rate.

After evaporation, all ants deposit pheromone on the arcs they have crossed in their tour:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k}, \qquad \forall (i,j) \in L, \tag{3}$$

Where; $(\Delta\tau_{ij}^{k})$ is the amount of ant (k) deposits on the arcs it has visited, it is defined as follows:

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{1}{C^{k}}, & \text{if arc}(i,j)\ \text{belongs to}\ T^{k}; \\ 0, & \text{otherwise}; \end{cases} \tag{4}$$

*2.2 Elitist Ant System*

2.2.1 Update Pheromone Trails

The additional reinforcement of tour ($T^{bs}$) is achieved by adding a quantity ($e/C^{bs}$) to its arcs, where (e) is a parameter that defines the weight given to the best-so-far tour ($T^{bs}$), and ($C^{bs}$) is its length [**8**]. Thus, equation (4) for the pheromone deposit becomes:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^{k} + e\,\Delta\tau_{ij}^{bs} \tag{5}$$

*2.3 Rank-Based Ant System*

2.3.1 Update Pheromone Trails

The best-so-far tour gives the strongest feedback, with weight (i.e., its contribution ($1/C^{bs}$) is multiplied by *w*); the r-th best ant of the current iteration contributes to pheromone updating with the value ($1/C^{r}$) multiplied by a weight given by max {0, *w* – r}. Thus, the ($AS_{rank}$) pheromone update rule is:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{r=1}^{w-1} (w-r)\,\Delta\tau_{ij}^{r} + w\,\Delta\tau_{ij}^{bs} \tag{6}$$

*2.4 MAX-MIN Ant Systems*

2.4.1 Update Pheromone Trails

After all ants have constructed a tour, pheromones are updated by applying evaporation as in AS [equation (2)], followed by the deposit of new pheromone as follows:

$$\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau_{ij}^{best} \tag{7}$$

*2.5 Ant Colony System*

(ACS) differs from (AS) in three main points. First, it exploits the search experience accumulated by the ants more strongly than (AS) does through the use of a more aggressive action choice rule. Second, pheromone evaporation and pheromone deposit take place only on the arcs belonging to the best-so-far tour. Third, each time an ant uses an arc (*i, j)* to move from city *i* to city *j*, it removes some pheromone from the arc to increase the exploration of alternative paths.

2.5.1 Tour Construction

In ACS, when located at city ( i ), ant ( k ) moves to a city ( j ) chosen according to the so called ( pseudorandom proportional ) rule, given by:

$$j = \begin{cases} \arg\max_{l \in N_i^k} \{\tau_{il}[\eta_{il}]^{\beta}\}, & \text{if } q \leq q_o; \\ J & \text{Otherwise}; \end{cases} \tag{8}$$

2.5.2 Global Pheromone Trail Update

In ACS only one ant (the best-so-far ant) is allowed to add pheromone after each iteration. Thus, the update in ACS is implemented by the following equation:

$$\tau_{ij} \leftarrow (1-\rho)\,\tau_{ij} + \rho\,\Delta\tau_{ij}^{bs}, \qquad \forall (i,j) \in T^{bs}, \tag{9}$$

2.5.3 Local Pheromone Trail Update Results

In addition to the global pheromone trail updating rule, in ACS the ants use a local pheromone update rule that they apply immediately after having crossed an arc (i, j) during the tour construction:

$$\tau_{ij} \leftarrow (1-\xi)\,\tau_{ij} + \xi\,\tau_{0}, \tag{10}$$

## III. The Proposed Backtracked Ant Algorithm to Solve 4-Color Graph Problem

One of the most studied problems in graph theory, the graph coloring problem, happens also to be one of the most computationally difficult.

A coloring of a simple graph is the assignment of a color to each vertex of the graph so that no two adjacent vertices are assigned the same color. The chromatic number of a graph is the least number of colors required to do a coloring of a graph. It is well known that determining of a graph can be colored by a certain number of colors is NP-complete, but it is also known that even approximating the chromatic number of a graph is NP-hard.

Since the time that mapmakers began making maps that show distinct regions (such as countries or states), it has been known among those in that trade, that if you plan well

enough, you will never need more than four colors to color the maps that you make.

The celebrated Four Color Map theorem states that any map in the plane or on the sphere can be colored with only four colors such that no two neighboring countries are of the same color. The problem has a long history and inspired many people (including many non-mathematicians and in particular countless high school students) to attempt a solution.

The basic rule for coloring a map is that no two regions that share a boundary can have the same color (The map would look ambiguous from a distance). It is okay for two regions that only meet at a single point to be colored the same color, however. If you look at a some maps or an atlas, you can verify that this is how all familiar maps are colored. Mapmakers are not mathematicians, so the assertion that only four colors would be necessary for all maps gained acceptance in the map-making community over the years because no one ever stumbled upon a map that required the use of five colors. When mathematicians picked up the thread of the conversation, they began by asking questions like: Are you sure that four colors are enough? How do you know that no one can draw a map that requires five colors? What is it about the way that regions are arranged and touch each other in a map that would make such a thing true?

When the question came to the European mathematics community at the end of the 19th century, it was perceived as interesting but solvable. Prominent and experienced mathematicians who tackled the problem were surprised by their inability to solve it.

Four Color ants algorithm is a metaheuristic to near optimal solution to solve the graph coloring problem. In four color ants a colony of artificial ants iteratively colors a specific graph, at each iteration. Initially ants produce feasible colorings by considering pheromone trails and heuristic information and afterword pheromone trails are updated according to the quality of colorings. The quality of colorings is measured using the following evaluation function:

$$f(s) = \frac{1}{g(s)} \tag{11}$$

Where, g (s) denotes the number of colors applied in coloring (s). Pheromone trails are related to pairs of non-adjacent vertices. Therefore, each pair of non-adjacent vertices $\{v_i, v_j\}$ has an associated phero-trail $\tau_{ij}$ that represents the colony experience of colorings in which the two mentioned vertices have the same color, i.e. belong to same color class. There exist several stages: At stage k;

- The artificial ant constructs four color class C4.
- There are several steps, at each step the artificial ant determines which uncolored vertex is to be added to the color class C4.
- Let w be the set of uncolored vertices that can be added to C4.
- Let B be the set of uncolored vertices which are not allowed to be added to C4.

Note there are three methods we can use to select a node from the set of w for coloring it:

1. $\eta_{ik} = \deg_w(v_i)$, this equation means that the node has maximum neighbors which make it possible to select it and adding the ant to begin the search with it.
2. Randomly selecting an uncolored vertex from w.
3. The probabilistic decision rule is expressed as:

$$P_{ik} = \begin{cases} \dfrac{\tau_{ik}^{\alpha}\,\eta_{ik}^{\beta}}{\sum_{j \in w}\tau_{jk}^{\alpha}\,\eta_{jk}^{\beta}} & v_i \in w \\ 0 & v_i \notin w \end{cases} \tag{12}$$

Where, $P_{ik}$ is the probability of selecting vertex $v_i$. The probability of adding any node from set w to any class depends on the above equation, if it is zero there is no node to add else we check the node with the content nodes in the class to avoid the conflict that it has a neighbor in such class and with the remaining class. Consequently; at each step of stage k, the probabilistic decision rule determines which uncolored vertex $v_i \in w$ is to be added to the color class 4.

- The third method is used to select the node from set w in our four color algorithm.
- Stage k continues while remaining non-empty.
- Pheromone trails are initially set to 1 and at the end of each iteration, they become updated considering the following rule:

$$\tau_{ij} = (1-\rho)\,\tau_{ij} + \sum_{S \in S_{ij}} \frac{1}{q(s)} \tag{13}$$

Where, q (s) represents the number of colors applied to coloring s and $\rho$ denotes the pheromone evaporation rate. $S_{ij}$ is the subset of colorings in which the two non-adjacent vertices $v_i$ and $v_j$ belong to the same color class.

- In order to choose an uncolored vertex $v_i$ to be added to the color class C4, pheromone trail $\tau_{ik}$ is defined as follows:

$$\tau_{ik} = \frac{\sum_{j \in C_k}\tau_{ij}}{|C_k|} \tag{14}$$

$\tau_{ik}$ contains all the pheromone trails between vertex $v_i$ and so far added vertices in color class C4, in other words, it represents the colony experience of settling vertex $v_i$ with other vertices of C4 in the same color class.

- Stage k continues while remaining non-empty.

The main algorithm of this problem is in detail :

```
Begin                    //main
Integer        N % Number of cities
integer        M % number of Ants
Integer        G [N] [N] % Array of Graph
Real           t [N] [N] % pheromone trail
integer        CC[4][N]  % array of 4 color class
integer        W[N] % uncolored city
integer        Max_Iteration = 100
integer        Iter_Counter = 0
Real           SumP,Pr,MaxP;
Set            α = 2, β = 4, and ρ = 0.5
Boolean        Complete =FALSE
Integer        Wlength
```

Set all elements in t [N] [N] equal to 1
Choose randomly 4 cities and place each one in 4 classes such as each class has one city.

```
// RC = Rand (1 to N)
// C  = W [Rc]
While (Not Complete and Iter_Counter < Max_Iteration) do
 {
 For  i = 1 to N do
          Sump = 0.0
          For v = 1 to Wlength do
          SumP =Sum P + t[v] [i]^α × i^β
          Endfor
          MaxP = 0
        For v = 1 to W.length do
                Pr = (t[v][i]^α * I ^β) / SumP
              If   Pr >MaxP   then   MaxP=Pr and Pc=v
          Endfor
    C=W [Pc]
 For k=1 to 4 do
       ClassColr=CC[k]
    Flag=TRUE
    tSum=0.0
 For j=1 to ClassColr.lenth do
      ClassColorCity= ClassColr[j]
      If G[C] [ClassColorCity] = 1 then Flag = False
      tSum=tSum + t[k][j]
    End-For
    If Flag then
      CC[k] = C
      T[C] [k] = tSum / CC[k].length
      Remove C from W [N]
    Exitfor
  Endif
  Endfor
  End for
        For i=1 to N do
          For j=1 to N do
          t [i][j]=(1-p) *t[i][j]+1
          End for
  End for
       if w.length =0 then Complete=True
      Iter_Counter= Iter_Counter+1
      End of While
      End
```

## IV. Experimental Results

Practically, five methods of AS [Ant Colony Optimization (ACO), Ant System (AS), Elitist AS, Rank-Based AS and Max-Min AS] are used, then a comparison is made between the old algorithm of AS and the updated Backtracked AS algorithm. Best results are obtained in the updated algorithm for all the five methods in comparison with the old algorithm. From these five methods it is noted that the ACO method has the best results among them. In the range of

$$\left[ \frac{3}{4} n \rightarrow n, n = \text{No.of ants} \right]$$ generally the (BACO) is the best method compare with the other types.

Three numbers of cities used in the experimental results 10, 20 and 3 cities respectively. Numbers of ant is very important factor in all ant systems so, we take 5, 8 and 10 ants for 10 nodes. In 20 nodes we take 5, 10, 14, 18 and 20 ants, and 5, 10, 15, 18, 24 and 30 ants for 30 nodes.

Figures 1, 2 and 3 illustrate the maximum iteration of experiments of 10, 20 and 30 nodes using (ACO, Backtracked ACO, AS, Backtracked AS, Elitist, Backtracked Elitist, Rank-Based AS, Backtracked Rank-Based AS, Max-Min AS and Backtracked Max-Min AS) respectively.

Figures 4, 5 and 6 illustrate the average time in seconds of experiments of 10, 20 and 30 nodes using (ACO, Backtracked ACO, AS, Backtracked AS, Elitist, Backtracked Elitist, Rank-Based AS, Backtracked Rank-Based AS, Max-Min AS and Backtracked Max-Min AS) respectively.

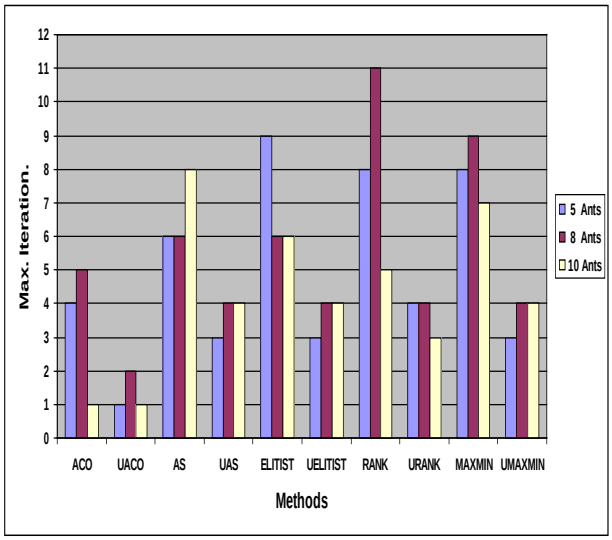
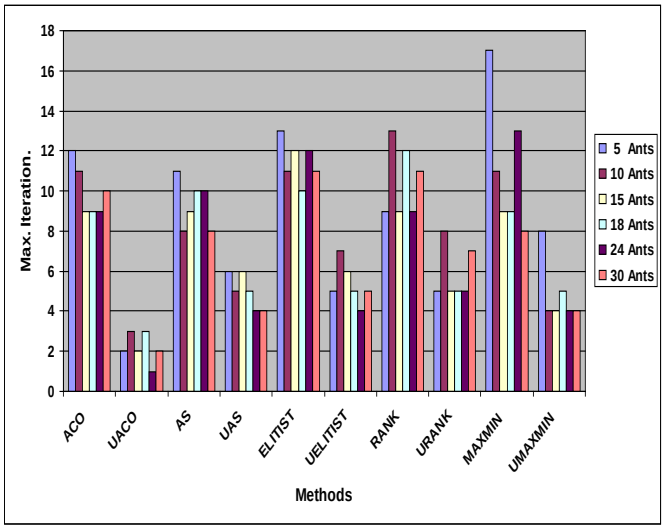
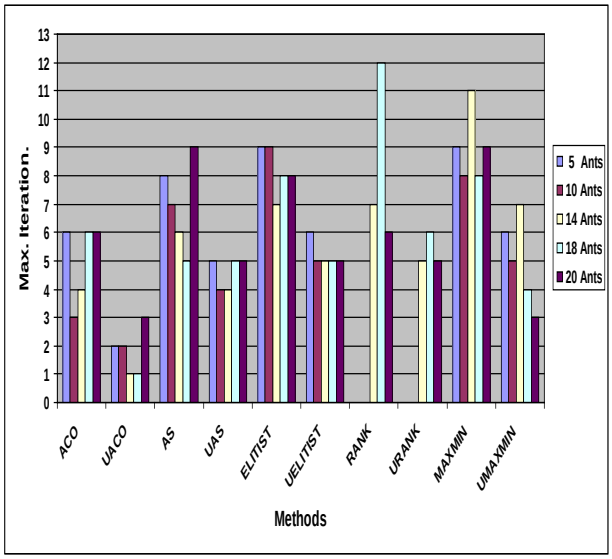



Figure 1: Max. Iterations for four Color Graph (Nodes =10) Using Different Ant Methods



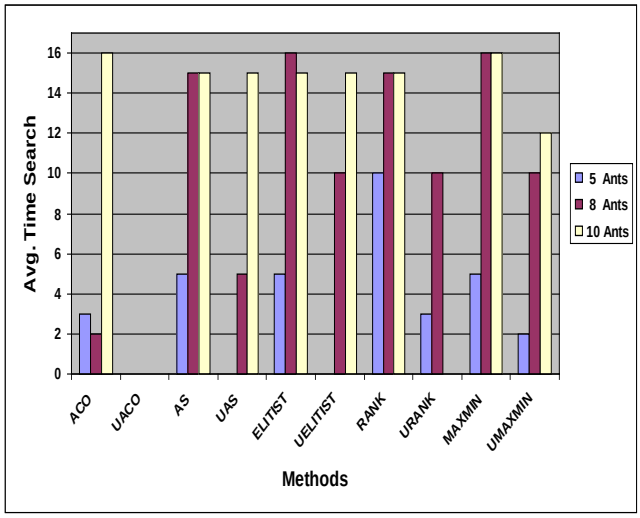

ACO BACO AS BAS ELIT BELIT RANK BRANK MAXMIN BMAXMIN

Figure 3: Max. Iterations for four Color Graph (Nodes =30) Using Different Ant Methods




ACO BACO AS BAS ELIT BELIT RANK BRANK MAXMIN BMAXMIN

Figure 2: Max. Iterations for four Color Graph (Nodes =20) Using Different Ant Methods




ACO BACO AS BAS ELIT BELIT RANK BRANK MAXMIN BMAXMIN

Figure 4: Avg. Time Search for four Color Graph (Nodes =10) Using Different Ant Methods

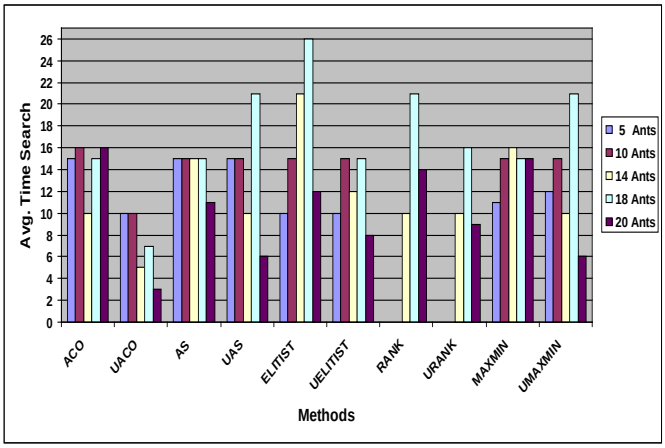



Figure 5: Avg. Time Search for four Color Graph (Nodes =20) Using Different Ant Methods

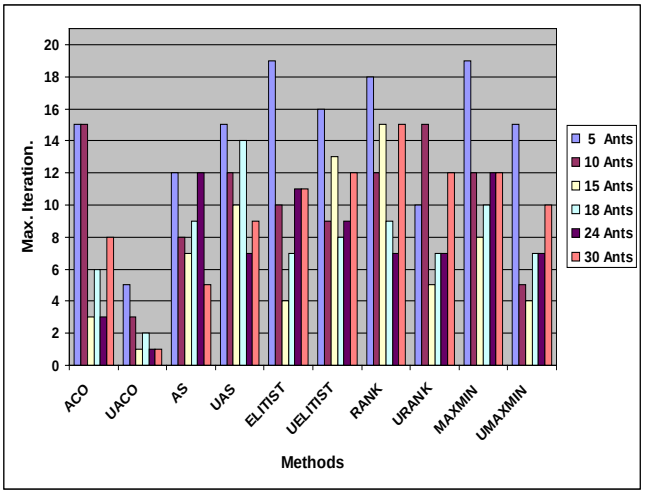



Figure 6: Avg. Time Search for four Color Graph (Nodes =30) Using Different Ant Methods

## V. CONCLUSIONS

1. The distribution time is increased with increasing the number of ants, therefore, the proposed system will decrease the time by decreasing the number of ant by using Backtrack ant systems and blocking some ants.
2. In the proposed work, the operations of the update pheromone are computed in the first, second, third iterations…….. and so on. But the update of pheromone will stop in the last iterations in the proposed work; therefore the execution time will decrease.
3. During the execution and applying the proposed Ant System algorithm it is found that each type of these five types depends on certain parameters which differ from one type to another. Also it is found that the values of these different types of ant systems have (or take) a certain configuration and determined values, the results that give the best performance are like the following:
    a. For ACO: $\xi$ (RHO) = 0.18, qo = 0.95 (increasing the value of qo gives more best results)
    b. For AS: $\xi$ = 0.1, BETA = 5.0
    c. For Rank-Based AS: $\xi$ = 0.001, BETA = 3.0
    d. For Elitist AS: $\xi$ = 0.001, BETA = 3.0
    e. For Max-Min AS: $\xi$ = 0.1, BETA = 2.0
4. The value of $\xi$ is limited between the (0 and 1), during the execution of the proposed algorithm on the ACO ant system. It is found that the best value of $\xi$ is (0.18) which gives the best performance for this method.
5. the experimental application results of the ant systems it is discovered that the ACO method is the best one among them which gives good results and performance, like the following:
    a. Minimum cost
    b. Minimum execution time
    c. Minimum tour length
    d. Nearest iteration
6. In Four Color Graph problem; a conflict node number is noted in some figures with increase in the number of ants, while the same figures are suitable for smaller number of ants.
7. In AS method the deposited pheromone and the updated pheromone are applied to all paths between the nodes in the graph, while in the ACO method these two types of pheromones are applied only to the paths of the best tour solution.

## REFERENCES

[1] Beni, G., and Wang, J.: "**Swarm intelligence in cellular robotics systems**", in Proceeding of NATO Advanced Workshop on Robots and Biological System, 1989.

[2] Grasse, P. P.: "**The reconstruction of the nest and co-ordinations inter-individual's chez belli-coitermes neatens and cubature sp. Theory of stigmergy**", Essai d'interpretation du comportement des termites constructeurs. Insect Sociaux, 6, pp. 41-81, 1959.

[3] Franks, N.R.:"**Army Ants: A Collective Intelligence**", Scientific American, Vol. 77, 1989.

[4] H**Ö**lldobler, B., and Wilson, E.O.:"**Journey to the ants**", Bellknap Press/Harvard University Press, 1994.

[5] Bonabeau, E., Dorigo, M., and Theraulaz, G.: "**Swarm Intelligence, From Natural to Artificial Systems**", New York, Oxford University Press, 1999.

[6] Deneubourg, J.L., and Goss, S.:"**Collective Patterns and Decision Making. Ethology**", Ecology and Evolution, 1:295-311, 1989.

[7] Agrawal, S., and Gupta, R.K.:"**Data-flow Assisted Behavioral Partitioning for Embedded Systems**", In Proceedings of the 34th Annual Conference on Design Automation Conference, 1997.

[8] Dorigo, M., and Stützle, T.: "**Ant Colony Optimization**", a Bradford Book, the MIT Press, London, England, 2004.