

ANT SYSTEM MODEL FOR THE N-QUEEN PROBLEM

*Md. Mujibur Rahman Khan**, *Md. Ahshan Razib*, *Md. Altaf Hossain*, *Shah Md. Arifur Rahman* and
Nasif Mahmud

*Department of Computer Science and Engineering, Bangladesh University of Engineering and
Technology (BUET),
Dhaka, Bangladesh

Department of Computer Science and Engineering, East West University,
Mohakhali, Dhaka, Bangladesh

E-mail : {mujib1295, ahsanrazib, altaf008bd, fahim21bd, nasifmahmud}@yahoo.com

ABSTRACT

An analogy with the way ant colonies function has suggested the definition of a new computation paradigm - known as Ant System. It is a viable new approach to stochastic combinatorial optimization. This paper presents a model of the Ant System (AS) that can efficiently be applied to solve the classical n-Queen problem. The main characteristics of the model are positive feedback, distributed computation, and the use of constructive greedy heuristic. Positive feedback accounts for rapid discovery of good solutions, distributed computation avoids premature convergence, and the greedy heuristic helps find acceptable solutions in the early stage of the search process.

1. INTRODUCTION

The intelligent behavior of some insects like ants has been amazed the scientist for hundreds of years. The more research and observations are performed, the more interesting facts are coming out. In fact, research on the behavior of ants has greatly inspired the works of many scientists all over the world (see [1], [2], [4]). From the observation [3] proposed a new paradigm of computation – model of real ants – the Ant System. It is proposed as a viable new approach to stochastic combinatorial optimization. In this approach the search activities are distributed over so-called "ants", that is, agents with very simple basic capabilities which, to some extent, mimic the behavior of real ants. The main characteristics of this model are positive feedback, distributed computation, and the use of a constructive greedy heuristic [3]. Positive feedback accounts for rapid discovery of good solutions,

distributed computation avoids premature convergence, and the greedy heuristic helps find acceptable solutions in the early stages of the search process.

In this paper we present a solution to the classical n-Queen Problem using the Ant System. For a better understanding of our proposed method, the behavior of ant colonies and the Ant System is briefly described in the following sub sections.

1.1 The Ant System

The AS is introduced and best described in [3]. Here we present a model to solve the n-Queen problem. The model is derived from the study of real ant colonies. As the model is not intended to simulating ant colonies, but to use of artificial ant colonies as an optimization tool, our AS have some major differences with a real (natural) one:

- artificial ants have some memory,
- they are not be completely blind, and
- they live in an environment where time is discrete.

The basic idea is to leave a number of ants in the problem space, then let them wonder around with some given rules to find out the optimal solution. Each ant is a simple agent with the following characteristics:

- At each cycle every ant moves to its adjacent cell or its own cell. We consider the movement to own cell in order to avoid local maxima.
- It chooses the cell to go to with a probability that is a function of the number of conflicts in the

local solution¹ and the intensity of attacking cells.

- To prevent all the ants collapsing into a single cell, transitions to the cells that already containing n number of ants are disallowed;
- When it completes a move, it lays a substance called *trail* on the leaving cell.

1.2 The model

The problem space is consists of an $n \times n$ grid. Individual cells are identified as $cell_{ij}$. Here we considered cycle as time. Let $b_{ij}(t)(i, j = 1, 2, \dots, n)$ be the number of ants in $cell_{ij}$

at time t and let $m = \sum_{i=1}^n \sum_{j=1}^n b_{ij}(t)$ be the total number

of ants. Let $\tau_{ij}(t)$ be the *intensity of trail* on $cell_{ij}$ at time t . Each ant at time t chooses the next cell, where it will be at time $t+1$. Therefore, in one *iteration* of the AS algorithm the m ants take m moves in the interval $(t, t+1)$. In every cycle the cell intensity is updated according to the following formula

$$\tau_{ij}(t+1) = \rho \cdot \tau_{ij}(t) + \Delta \tau_{ij}, \quad \dots \dots (1)$$

and $\Delta \tau_{ij} = \sum_{k=1}^{m \times n} \Delta \tau_{ij}^k$ where, $(1 - \rho)$ represents the

evaporation of trail between time t and $t+1$. Here $\Delta \tau_{ij}^k$ is the intensity of $cell_{ij}$ laid by the k -th ant and is equals to $\frac{C^k}{Q}$ if the k -th ant has moved from $cell_{ij}$ to another cell or its own cell in this cycle. C^k is the number of ants attacking the k -th ant and Q is a constant. The laid intensity $\Delta \tau_{ij}$ is added to the intensity of the leaving cell of the ants to give the cell some information about the adjacent cells of their current locations.

The attacking probability is expressed as –

$$A_{ij}^k = \frac{[N_{ij}]^\alpha \cdot [\tau_{ij}(t)]^\beta}{\sum [N_{ik}]^\alpha \cdot [\tau_{ik}(t)]^\beta}, \text{ where } k \in \text{allowed}_k \dots (2)$$

Here N_{ij} stands for the number of conflicts in the *local solution* and k denotes that the equation is applicable for the cells which are allowed. Here allowed cell means the cell where number of ants is

¹ For each column, the cell with highest number of ants is considered as the cell with the queen. These cells are making up the local solution after a cycle.

less than n^* and which are adjacent to the cell from which an ant is supposed to move. For example, if $cell_{ij}$ is one of the centered cells of the board and each adjacent cell of it has ants less than n then the number of allowed cell will be nine including the $cell_{ij}$ itself. α and β are parameters that control the relative importance of number of conflicts in the *local solution* versus intensity.

2. THE ALGORITHM

2.1 Overview

From each column of the board we choose a cell which has the highest number of ants as our *local solution* in each cycle. Ties are broken randomly. A sample state of the board at some intermediate cycle is shown in Figure 1. The numbers in the cells

0	0	0	2	0	5	0	3
0	0	8	0	2	0	0	0
0	1	0	0	0	0	4	0
5	0	0	2	0	0	0	0
0	0	0	0	0	0	0	2
0	0	0	0	6	0	0	0
0	0	0	2	0	0	3	0
5	7	0	0	2	5	0	0

Fig. 1 A sample distribution of ants and local solution of 8-Queen.

indicate the number of ants in respective cells. The cells with boldfaced numbers are making up the *local solution*. At first the procedure selects a cell randomly from each column of the board and put n ants on that cell. Then at each cycle the ants move to the adjacent cells whose attacking probability is lowest. The ant does not move if the attacking probability of its own cell is lowest with compare to the adjacent cells. The process will continue until the outcome of the program is saturated. It happens when the attacking probability of the current cells of all the ants are lower with correspond to their adjacent cells. At that moment no ants will move to another cell. They will belong to the current cell and the program will terminate.

2.2 Outline

1. Initialize:
 - Randomly choose a cell in each column;
 - Set n number of ants in that selected cell;

* Assuming that there is a total number of $n \times n$ ants on the board

Set each cell with a low constant intensity;
 Mark the local solution;
 Compute the number conflict in the local solution and number of ants' conflict of each cell;

2. For each k from 1 to $n*n$
 - k -th ant choose the cell to move with lowest attacking probability;
 - The cell k -th ant moved is stored into a temporary ant list;
 - Update the number of ants of that cell;
 - Update the cell intensity (Multiplying the intensity of each cell by ρ);
3. For each k from 1 to $n*n$
 - Set the number of ants conflict of all cells that are affected by the k -th ant for movement;
 - Update the cell information of k -th ant in the original ant list named *ant*;
 - Update the intensity of the cell that k -th ant has just left (Adding $\Delta\tau_{ij}^k$ with $\rho\tau_{ij}(t)$);
4. Identify the local solution;
 Calculate the conflicts in the local solution;
 Increase the cycle counter;
5. If maximum cycle is reached or local solution conflict in the goal cells is absent or ants are not leaving their own cell then return the local solution and stop. Otherwise go to step 2.

3. EXPERIMENTAL RESULTS

We have implemented our proposed ant algorithm primarily for $n = 4, 8, 9, 10,$ and 11 . But we are still conducting experimentations to identify the range of the parameters within which the AS performs best. Figure 3 shows the minimum number of ant cycles needed to get a solution for different values of n .

These minimum numbers of ant cycles are found at different combination of the parameters. Table 1

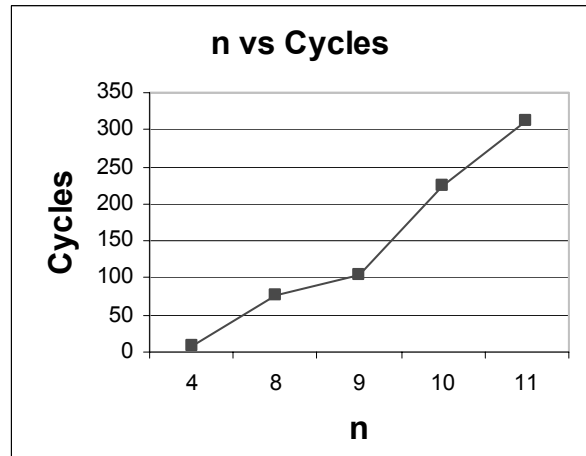


Fig. 3 Number of Queens Vs Minimum Ant Cycles required.

shows the combination of parameter values for which the solutions were found with minimum ant cycles. We also present the data obtained by a

Table 1: Values of the parameters for solutions with minimum cycles.

n	α	β	$1 - \rho$	Q	Cycles
4	1.5	1	0.99	0.4	9
8	1	1	0.99	1.9	76
9	2	1.5	0.97	2.2	105
10	2.8	1.5	0.98	2.7	223
11	2.5	1.4	0.99	3.3	313

typical run for 8-Queen to have a better understanding on the Ant System behavior for nQueen problem (see Figure 2). The parameters are set as $\alpha = 2, \beta = 1, \rho = 0.98$ and $Q = 0.7$. Figure 2 shows graph which expresses the number of attacking vs. the number of ant cycles. In this particular run, a local solution with zero attacking

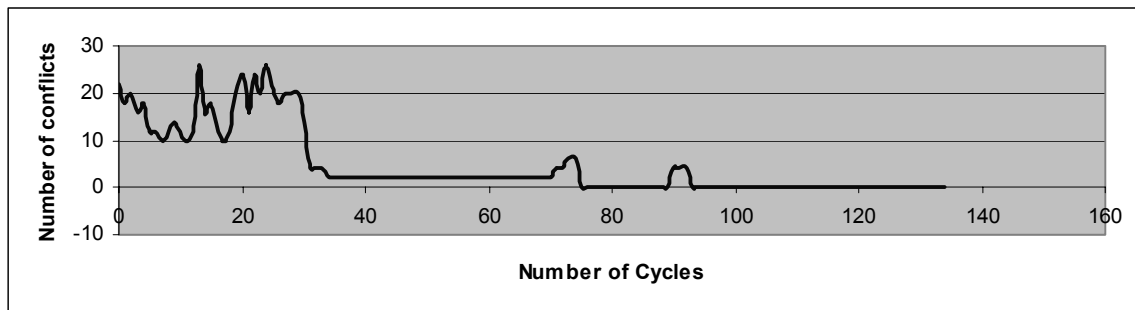


Fig. 2 Solution level in different cycle of typical run for solving 8-Queen problem

was found in 75th cycle. But the program has not got saturated in this cycle. There exist some conflicted ants. So, movement has not stopped. For this, in 90th cycle, some optimality has lost. The program has got saturated after the 95th cycle. After the 133rd cycle, ant's movement is totally stopped. Every ant stays at its own cell. There are n (=8) ants in each goal cell in this stage. This is known as the stagnation behavior.

4. CONCLUSION

We presented a solution to the classical n-Queen problem. We used the recently investigated and discovered paradigm computation, known as the Ant System. We first presented a model of AS for the specific problem. Then we proposed an algorithm to solve the problem using our AS model. Further research could be carried on to find better definition of the model and the algorithm for this specific problem. The ranges of parameter values have a great influence on the performance and efficiency of the system. So finding the optimal ranges of the parameters are also of great importance.

REFERENCES

- [1] Deneubourg, J.L., Pasteels, J.M. and Verhaeghe, J.C. "Probabilistic Behaviour in Ants: a Strategy of Errors?", *Journal of Theoretical Biology*, 105, 259–271, 1983.
- [2] Deneubourg, J.L. and Goss, S. "Collective patterns and decision-making", *Ethology, Ecology & Evolution*, Vol.1, 295–311, 1989.
- [3] Dorigo, M., Maniezzo, V. and Colomi, A. "The Ant System: Optimization by a colony of cooperating agents", *IEEE Transactions on Systems, Man, and Cybernetics–Part B*, Vol.26, No.1, 1996, pp.1-13
- [4] Goss, S., Beckers, R., Deneubourg, J.L., Aron, S. and Pasteels, J.M. "How Trail Laying and Trail Following Can Solve Foraging Problems for Ant Colonies", in *Behavioural Mechanisms of Food Selection*, R.N.Hughes ed., NATO-ASI Series, vol. G 20, Berlin:Springer-Verlag, 1990.
- [5] Horowitz, E. and Sahni, S. *Fundamentals of Computer Algorithms*, Computer Science Press, USA.
- [6] Lawler, E.L., Lenstra, J.K., Rinnooy-Kan, A.H.G. and D.B.Shmoys eds., *The Travelling Salesman Problem*, New York:Wiley, 1985.
- [7] Papadimitriou, CH. and Steiglitz, K. *Combinatorial Optimization Algorithms and Complexity*, Prentice-Hall Inc. USA.
- [8] West, DB. *Introduction to Graph Theory*, Prentice-Hall Inc. USA