CELLANT: AGENT BASED DYNAMIC ROUTING FOR CELLULAR COMMUNICATION NETWORK

Md. Rakib Hassan, K. Hassan and MMA Hashem

Dept. of Computer Science and Mathematics, BAU, Mymensingh; Dept. of Computer Science & Engineering, Khulna University of Engineering & Technology, Khulna

E-mail: rakibkuet@yahoo.com

Abstract: *Cellular telecommunication is one of the fastest growing and most demanding telecommunication applications ever. In cellular telecommunication, an adaptive call establishment between two BTS (Base Transceiver Station) which are under different MSC (Mobile Switching Center) is an intelligent routing procedure. This paper presents CellAnt, an agent based routing algorithm for BTS search in the Mobile communication network. In CellAnt, a set of cooperating agents called ants cooperate to find shortest path. It's a mobile agent based algorithm which is inspired by the Ant Colony System (ACS). CellAnt algorithm is used for dynamic routing in the mobile communication network. The experiments are run for various traffic distributions. CellAnt have showed very good performances with respect to its competitors.*

Keywords: *Cellular Telecommunication, Ant Colony System (ACS), Base Transceiver Station (BTS), Mobile Switching Center (MSC).*

1. Introduction

Cellular telecommunication represents a large and continuously increasing percentage of all new telephone subscriptions around the world. The use of digital radio transmission and the advanced handover algorithms between radio cells in GSM networks allow for significant usage than in analogue cellular systems, thus increasing the number of subscribers that can be served.

Advance Mobile Phone Service (AMPS) [1] provide telephone communications to thousands of mobile users within a greater metropolitan area. And to accomplish this goal, it uses a minimal amount of the frequency spectrum. In this type of network, each geographical area is divided into small regions called cells.

At the center of the cell is a base station (Base Transceiver Station – BTS) [2] to which all the telephones in the cell transmit. In a small system, all the base stations are connected to a single device called a Mobile Switching Center (MSC). In a larger one, several MSC maybe needed, all of which are connected to a second level MSC and so on. At any instance, each mobile telephone is logically in one specific cell's base station.

Real ants have been shown to be able to find shortest paths using only the pheromone trail deposited by other ants. The artificial ant uses this pheromone to find out the BTS under any MSC. When the called person is under another MSC' BTS then the caller's home MSC generates an ant to find out the rout and search MSC under where the destination BTS is present.

The ant uses the probability from Probability Table to go from one MSC to another MSC. When any mobile user move from one cell to another cell then the HLR (Home Location Register) of that cell's home MSC is upgraded automatically. VLR (Visitor Location Register) of the MSC contains the routing table sometimes denoted here as Database Table.

The CellAnt algorithm is so used to find out the destination MSC (BTS) and to upgrade the VLR of source cell's home MSC. The shortest route is stored in the Database Table (VLR). The Database Table is upgraded dynamically as ant can launch from any other MSC for searching another MSC, a new mobile user. Among the conventional routing algorithms, the CellAnt algorithm is faster to find out the destination BTS.

The routing algorithm that is proposed in this thesis was inspired by previous works on ant system, ant colony system [3], [4], where pheromone is used for the shortest path direction. In ant colony system each ant can move from one place to another place and deposit pheromone on the way. So, next ant can be able to find shortest

Date of submission : 04. 12.2008 Date of acceptance : 06. 02. 2011

path using only this pheromone trail deposited by other ants.

In CellAnt algorithm, each artificial ant build a path from source BTS to destination BTS. While building the path, it collects explicit and implicit information about load status of each MSC. This information upgrades the Probability Table of the visited MSC. The behavior of CellAnt algorithm is compared to some effective shortest path algorithm. CellAnt shows the best performance and more stable behavior for all traffic distribution. Absolute performance is scored according to a scale defined by an ideal algorithm giving an empirical bound. Competing algorithms performed poorly for heavy traffic conditions and showed more sensitivity to internal parameters tuning.

2. An Overview of the Routing Algorithms

The goal of every routing algorithm is to direct traffic from sources to destinations maximizing network performance while minimizing costs. In this way, the general problem of determining an optimal routing algorithm can be stated as a multi-objective optimization problem in a non-stationary stochastic environment. Additional constraints are posed by the underlying network switching and transmission technology.

The performance measures that usually are taken into account are average delay and error in finding the destination. The former quantify the quantity of service that the network has been able to offer in a certain amount of time, while the latter defines the quality of service produced at the same time.

Routing algorithms can be at first broadly classified as static or adaptive. In static (or oblivious) routers the path taken by an ant is determined only on the basis of the source and destination, without regard to the current network state. This path is usually chosen as the shortest one according to some cost criterion.

Adaptive routers are, in principle, more attractive, because they try to adapt the routing policy to the varying traffic conditions. As a drawback, they can cause oscillations in selected paths. This can generate circular routes, as well as large fluctuations in performances, especially for what concerns average delays.

The most widely used shortest path algorithms are shortest path algorithms .Shortest path searching has a source destination pair perspective: there is no global cost function to optimize. Here our algorithms objective is to determine the shortest path between two BTS, where the link delays are computed adaptively following some statistical description of the link.

The conventional searching method is called BLIND search where no intelligence is used. But here each ant (agent) can move from one MSC to another MSC, choosing the lower probability MSC and change the Probability Table. So every time each ant uses the modified Probability Table and is allowed to modify it. Going to the nearest MSC, change the Probability Table having the summation of the probability to go from one MSC to another MSC is equal to 1.00(one) [5], [6].

3. CellAnt: The Proposed Approach

3.1 Overview of CellAnt

As emphasized before, the routing problem is a stochastic distributed multi-objective problem. Information propagation delays and the difficulty to model the network dynamic under arbitrary traffic patterns, make the general routing problem intrinsically distributed. Routing decisions can only be made on the basis of local and approximate information about the current and the future network states.

These features make the problem well adapted to be solved following a multi-agent approach like our CellAnt algorithm, composed by two sets of homogeneous mobile agents, called in the following respectively forward and backward ants.

Agents in each set possess the same structure, but they are differently situated in the environment; that is, they can sense different inputs and they can produce different, independent outputs. Agents behave reactively retrieving a pre-compiled set of behaviors to select the rout and to modify the Probability Table, but at same time they maintain a complete internal state description.

3.2 The CellAnt Algorithm

1. At regular intervals, from every MSC s (under where the source BTS is connected), a mobile agent (ant) is launched with a randomly selected destination MSC d (under where a destination BTS is connected) .The identifier of every visited MSC k and the time elapsed since its launching time to arrive at this k-th MSC are pushed onto a memory $STACK_{s\to d}(k)$.
2. At first an ant selects a MSC which is reachable from current position and test whether it is destination MSC. If it is, it must be go to the destination MSC. And this path and the delay is pushed on the $STACK_{s\to d}(k)$.
3. Each traveling ant selects the next hop MSC using the information stored in the Probability Table. The route is selected, following a random scheme, proportionally to the goodness (probability of each neighbor MSC) or with a tiny probability (exploration probability), assigning the same selection probability to each of the neighbor MSC. If, in the proportional case, the chosen MSC is already been visited, a uniformly random selection among the neighbors is applied.
4. If a cycle is detected, that is, if an ant is forced to return in an already visited MSC, the cycles MSCs are popped from the ants $STACK(k)$ and all the memory about them destroyed.
5. When the destination MSC d is reached, the agent $F_{s\to d}$ generates another agent (backward ant) $B_{d\to s}$ transferring to it all its memory.
6. The backward ant makes the same path as that of its corresponding forward ant, but in the opposite direction. At each MSC k along the path it pops its stack $STACK_{s\to d}(k)$ to know the next hop MSC.

   Arriving in a MSC k coming from a neighbor MSC f, the backward ant updates the following two data structures maintained by every MSC:

a) A Probability Table organized as in vector distance algorithms; in the table, a probability value $\Pr ob(i,n)$ which expresses the goodness of choosing n as next MSC when the destination MSC is i, is stored for each pair (i, n) with the constraint:

$$\sum_{n\in N_k} \Pr ob(i,n) = 1, \quad i \in [1,N], \quad N_k = \{neighbor\ (k)\} \quad \ldots (1)$$

b) A list $Memory_k(mean, \text{var}iance^2)$ of estimates of arithmetic mean values $mean_i$ and associated variance $\text{var}iance_i$ for trip times from itself to all the MSCs i in the network. This data structure represents a memory of the network state as seen by MSC k.

These two data structures are updated as follows:

The list $Memory_k(mean, \text{var}iance^2)$ is updated with the values stored in the stack memory $STACK_{s\to d}(k)$; all the times elapsed to arrive in every MSC $k' \in S_{k\to d}$ starting from the current MSC k are used to update the corresponding sample means and variances $Memory_k(mean, \text{var}iance^2)$.

The Probability Table is changed incrementing the probability *Prob(d, f)* associated with MSC f when the destination is MSC d and decrementing the probability *Prob(d, n)* associated with the other MSC n in the neighborhood for the same destination.

The update of the Probability Table happens using the only available feedback signal, that is, the trip time experienced by the forward ant. This time gives a clear indication about the goodness of the followed route because it is proportional to its physical length and to the traffic congestion.

The time measure is used as a reinforcement signal to provide structural and temporal credit assignment. The credit assignment problem is the typical one arising in reinforcement learning field [6]. "Optimal" times depend on traffic and/or components failure states, and they have to be considered from a network wide point of view. An "advice" about the goodness of the observed trip time on the basis of the estimated means values for the agent's trip times, are stored in the list $Memory_k(mean, \mathrm{var}iance^2)$.

Now $mean = \mu$ and $\mathrm{var}iance^2 = \sigma^2$ is used. In this situation, the routing table is updated in the following way:

If T is the observed trip time (delay between i and n when an ant come form i to n) and μ is its (i-th) mean value, as stored in the list $Memory_k(mean, \mathrm{var}iance^2)$, the computation of a raw quantity r' measuring the goodness of T, with small values of r' corresponding to satisfactory trip times [5]:

$$r' = \begin{cases} \frac{T}{c\mu}, & c \geq 1 \quad if \quad \frac{T}{c\mu} < 1 \\ 1 & otherwise \end{cases} \qquad \ldots\ldots \quad (2)$$

r' is a dimensional measure, problem independent, scoring how good is the elapsed trip time with respect to what has been on average observed until now. μ plays the role of a unit of measure and c is a scale factor (setting $c=2$ is a reasonable choice). "Out-scaled-values" are saturated to 1.

A correction strategy is applied to the goodness measure r' taking into account how reliable is the currently observed trip time with respect to the variance in the so far sampled values, that is, considering how stable the trip time mean value is. The observations in the mean are stable if $\frac{\sigma}{\mu} < \varepsilon, \quad \varepsilon << 1$.

In this case, a good trip time (i.e.: r' less than a threshold value t that is set to 0.5) is decreased by subtracting a value [5]:

$$S(\sigma, \mu; a) = e^{\frac{-a\sigma}{\mu}} \qquad \ldots \quad \ldots \quad (3)$$

To the value of r' while a poor trip time is increased adding the same quantity. On the other hand, if the mean is not stable, the raw values r' cannot be completely considered reliable and, in this case, the quantity [5]:

$$U(\sigma, \mu; a') = e^{\frac{-a'\sigma}{\mu}} \qquad \ldots \quad \ldots \qquad (4)$$

with $a' < a$, is added to a good r' value and subtracted from a poor one. In this case, to avoid following the traffic fluctuations, with the risk of amplifying them: adding and subtracting the value U helps to stabilize them.

The above correction strategy, for both cases of $\frac{\sigma}{\mu}$ value can be summarized as [5]:

$$r' \leftarrow r' + sign(t - r')sign\left(\frac{\sigma}{\mu} - \varepsilon\right) f(\sigma, \mu) \qquad \ldots \quad \ldots \quad (5)$$

with f being S or U according to the case. The f functions have been chosen as decreasing/increasing exponential because both the function and its first derivative are monotonically decreasing/increasing with increasing values of the $\frac{\sigma}{\mu}$ ratio. These transformations from the raw value T to the more refined value r' play the role of a local estimation of a traffic model. More sophisticated and computationally-demanding models could be learnt to generate a more effective traffic-dependent correction.

The obtained value r' is used by the current MSC k to define a positive reinforcement, r_+ for the MSC f the ant comes from, and a negative one r_- for the other neighboring [5] MSC n:

$$r_+ \leftarrow (1 - r')(1 - Prob(d, f)) \qquad \ldots\ldots \quad (6)$$

$$r_{-} \leftarrow -(1-r')(\mathrm{Prob}(d,n)) \quad \ldots\ldots \quad (7)$$

where $n \in N$, $n \neq f$, where $\mathrm{Prob}(d,f)$ and $\mathrm{Prob}(d,n)$ are the last probability values assigned to neighbors of MSC k [7]. In this way, the reinforcements are proportional to the obtained goodness measure r' and to the previous value of MSC probabilities.

These probabilities are then increased/ decreased by the reinforcement values as follows (their sum will till be 1, being $r' \in [0,1]$:

$$\mathrm{Prob}(d,f) \leftarrow \mathrm{Prob}(d,f) + r_{+} \quad \ldots\ldots \quad (8)$$

$$\mathrm{Prob}(d,n) \leftarrow \mathrm{Prob}(d,n) + r_{-} \quad \ldots\ldots \quad (9)$$

It is now clear that the power law rescaling of the r' value is equivalent to the definition of a learning rate: the scale compression factor and its degree of non linearity determine the final size of the allowed jumps in the probability values.

4. Experimental Results

To evaluate the performances of CellAnt, two competitor algorithms are selected from the shortest path class reflecting network standards and state-of-the-art for routing algorithm.

4.1 Bellman-Ford (BF)

BF is an implementation of the asynchronous distributed Bellman-Ford algorithm with dynamic link matrix [8].Vector distance Bellman-Ford-like algorithms are today in use mainly for intra-domain routing, being used in the Routing Information Protocol (RIP) [9] supplied with the BSD (Berkeley Software Distribution) version of Unix.

4.2 Dijkstra

Dijkstra is a list-cost routing algorithm used in packet switching network, internet, and mobile communication network. In this algorithm, each MSC must have complete topological information about the network. That is, each MSC must know the link delay of all links in the network. Thus for this algorithm, information is exchanged with all other MSCs.

4.3 Experimental Settings

In this experiment a mobile network instance of figure 1 is simulated using graph in Visual C++. The hexagons in the figure are the nodes of the graph. The nodes represent BTS (each BTS is connected with one MSC). Each edge in the graph represents a pair of directed link and the numbers are propagation delay.

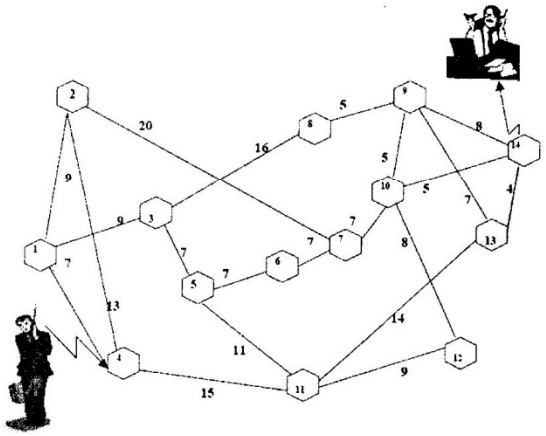

Fig. 1: Numbers in Hexagons (nodes) are BTS Identifier (each BTS is connected with one MSC).

Each edge in the graph represents a pair of directed link and the numbers are propagation delay. Two models, static and dynamic, of temporal traffic patterns have been used. In the static model all the sessions start at the beginning of the simulation and they last until the end. In this way, a situation of stationary is simulated. In the dynamic model, sessions are activated following a negative exponential distribution for the inter-arrival times. The distribution mean value is fixed. In this case sessions are "bursty" and hence data flows are highly irregular.

The constants (c, a, a', h, t) used in this section are not problem-dependent and they simply define an appropriate scaling system for the computed values. They have been set to the following values: $c=2, a=10, a'=9, h=0.04, t=0.5$.

4.4 Experimental Result

The performance of the CellAnt comparing Dijkstra and Bellman-Ford algorithm is evaluated for the source-destination pair with the following BTS pairs (1,7), (2,12), (3,10), (4,14), (5,9), (6,1), (7,11), (8,13), (9,2), (10,4), (11,8), (12,5), (13,3) and (14,6).

Table 1 shows the errors of different algorithms in finding the actual result from 1st BTS to 7th BTS. The table also contains the actual path length. The error is obtained by subtracting the observed value from actual value.

Table 1: Errors of Different Algorithm in Finding the Shortest Path from 1st BTS to 7th BTS

Iteration	Actual Value (Shortest Path)	Obtained Shortest Path			Errors in the Algorithm		
		CellAnt	Dijkstra	Bellman-Ford	CellAnt	Dijkstra	Bellman-Ford
1	29	40	7	7	21	22	22
2	29	30	9	9	1	20	20
3	29	30	9	9	1	20	20
4	29	29	16	16	0	13	13
5	29	29	22	22	0	7	7
6	29	29	22	22	0	7	7
7	29	29	25	25	0	4	4
8	29	29	29	29	0	0	0

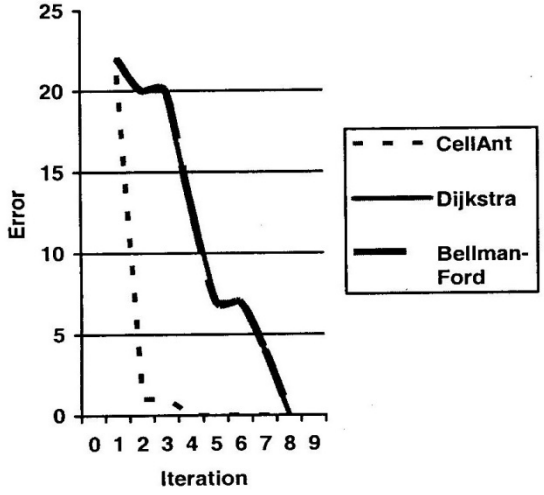

Fig.2: CellAnt , Dijkstra and Bellman-Ford's Error Decreasing Curve from BTS 1 to BTS 7

Fig. 2 shows the error decreasing curve for each algorithm. The figure clearly shows that CellAnt is more efficient in finding the optimum result than other algorithms.

Similarly, for every source and destination BTSs, we find that CellAnt obtains the optimum solution in fewer iterations than Dijkstra and Bellman-Ford algorithms. Table 2 summarizes the needed iterations for CellAnt , Dijkstra and Bellman-Ford algorithm to find the shortest path between different BTSs.

Table 2: The Actual Result in Different Iteration for Different Algorithm

Shortest MSC List =Total Cost	Needed Iterations		
	CellAnt	Dijkstra	Bellman-Ford
1-2-7 =29	4	8	8
2-7-10-12 =35	12	12	12
3-8-9-10=26	4	11	11
4-11-13-14 = 33	7	12	12
5-6-7-10-9 = 26	5	12	12
6-5-3-1 =23	1	9	9
7-10-12-11 =24	2	11	11
8-9-13 =12	2	4	4
9-10-7-2 =32	1	12	12
10-12-11-4=32	9	12	12
11-13-14-8 =26	7	12	12
12-11-5=20	2	10	10
13-14-8-3 =20	8	8	8
14-10-7-6 =19	4	8	8

5. Conclusion

In this paper, CellAnt, a new algorithm for adaptive routing for cellular communication is introduced. Its behavior with respect to iteration and error for finding actual cost has been compared to the behavior of two shortest path routing algorithms. The CellAnt performed always the best among its competitors or at the same level within the statistical fluctuations.

CellAnt shows stable performance and behavior, that is, always moving rapidly toward the optimum solution.

References

[1] H. Taub and S. Donald, Principles of Communication Systems, 1998.

[2] L. Schoofs and B. Naudts, "Swarm Intelligence on the Binary Constraint satisfaction problem," *IEEE World Congress on Computational Intelligence*, [CD-ROM], 2002.

[3] M. Dorigo and L.M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem," *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, 1998, pp. 53-66.

[4] M. Dorigo, V. Maniezzo and A. Colorni, "The Ant System, An Autocatalytic Optimizing Process," *Technical Report 91-016*, Politecnico di Milano (IT) Dipartimento Di Elettronica ed Informatica, 1991.

[5] G.D. Caro and M. Dorigo, "Mobile agents for adaptive routing," http://citeseer.nj.nec.com/dicaro98mobile.html, 1998.

[6] E. Bonabeau, M. Dorigo and G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, Oxford Press, 1999.

[7] A.S. Tanenbaum, Computer Networks, 1998.

[8] D. Bertsekas and R. Gallager, Data Network, Prentice-Hall, 1992.

[9] G. S. Malkin and M. E. Steenstrup, Routing in Communication Networks. Prentice- Hall, 1995.