An Agent-based Connection Management Protocol for Ad Hoc Wireless Networks

Romit RoyChoudhury
Dept. of Computer Science
Haldia Institute of Technology
West Bengal
INDIA

Krishna Paul
CTS
Sector V, Saltlake
Calcutta 700 091
INDIA

S. Bandyopadhyay*
PricewaterhouseCoopers Ltd.
Sector V, Saltlake
Calcutta 700091
INDIA

* E-mail: somprakash.bandyopadhyay@in.pwcglobal.comT

ABSTRACT

Realizing high volume of data transmission in real time communication in a highly dynamic architecture like Mobile Ad hoc Networks (MANET) still remains a major point of research. In this paper, we have attempted to address the issue of managing an uninterrupted connection between a source and a destination through multiple paths in a temporal domain. We have developed an agent-centric protocol that would accomplish uninterrupted communication over adaptively selected routes. The protocol ensures minimal consumption of network resources. We have organized this work in two logical steps. In the first part, we have devised an agent-based framework with its associated protocols and mechanisms. The agents in the framework move from one node to another in the MANET architecture. The primary objective of this mobile multi-agent framework is to asynchronously collect network topology information (like node velocity, node location etc.) and distribute them into the information cache of all other nodes. This is accomplished proactively with the aim of making all nodes in the system TOPOLOGY-AWARE. We have used the GPS (Global Positioning System) support at each node for the extraction of geographical co-ordinates, velocity and direction of movement of each node. The second part of the work attempts to make use of this topology awareness in the context of establishing and managing a connection between two nodes. The agent-enabled proactive replenishment of fresh topology information enables the node to constantly evaluate network conditions. This facilitates native nodal intelligence to take decisions on adaptive route selection, foresee route errors etc. through mere execution of algorithms on the local information cache.

1.0 Introduction

Ad hoc networks [1,2,3,4,5,6] are envisioned as infrastructure-less networks where each node is a mobile router, equipped with a wireless transceiver. A message transfer in an ad hoc network environment could either take place between two nodes that are within the transmission range of each other or between nodes that are indirectly connected via multiple hops through some intermediate nodes. This implies that the nodes, which act as intermediate nodes in the data transfer process, must be willing to participate in communication until successful message transfer has been accomplished. The failure of such an event would amount to messages getting lost or message transfer getting interrupted, thus generating unproductive congestion. Thus, any protocol in this context should be able to manage an uninterrupted connection, if possible, between any source-destination pair.

The dynamics of wireless ad hoc networks as a consequence of mobility and disconnection of mobile hosts pose a number of problems in designing proper routing schemes for managing effective communication between any source and destination [5]. To manage a connection between two nodes for a long span of time, the source node needs to be aware of the changing route status and subsequently of newly available routes. This points to a form of topology awareness [13] that either should be incorporated proactively or on-demand. In this paper we have assayed the issues of geographical-position-discovery through the implementation of a *mobile-multi-agent based framework*. In other words, we have developed an agent based information exchange and navigation protocol (proactive in nature) in order to make each node in the network, aware of the positions of other nodes, without consuming large portion of network capacity. We have used the GPS (Global Positioning System) support at each node for the extraction of geographical co-ordinates, velocity and direction of movement of each node. This local awareness gives each node a proactively refreshed perception of the network topology. Based on this perception, we have designed an adaptive routing protocol ensuring the system to manage an uninterrupted connection in a distributed fashion between any source and destination.

2.0 Related Work

Existing work on routing protocols in ad hoc networks have failed to provide an optimal solution towards connection management between any source and destination. The conventional proactive routing protocols that require to know the topology of the entire network is not suitable in such a highly dynamic environment, since the topology update information needs to be broadcast frequently throughout the network. Also, in a highly dynamic system, protocols need to broadcast updates too frequently simply because the link information changes quickly. On the other hand, a demand-based, reactive route discovery procedure generates large volume of control traffic and the actual data transmission is delayed until the route is determined [5]. In addition, route re-discovery in the event of route errors consumes time; this might prove to be intolerable in the case of a real time system.

In [3], a preemptive route discovery has been proposed in order to discover best routes dynamically and then adaptively use them for managing connectivity between any source and destination. However, the route-discovery mechanism generates substantial (and recurring) control packets through flooding. This causes unproductive traffic in the environment and may thus increase end-to-end delay for communication. Also, loss of control packets would mean the unavailability of stable routes and may interrupt the ongoing process of communication. There are proposals to reduce the control traffic generated in reactive protocols. For example, Location-Aided Routing (LAR) Protocols [8] suggest an approach to decrease overhead of route discovery by utilizing location information for mobile hosts. Such location information may be obtained using the global positioning system (GPS). The LAR protocols use location information to reduce the search space for a desired route. Limiting the search space results in fewer route discovery messages. In a recent work, new MAC protocols [9] using directional antennas has been proposed to improve the routing performance of Location-aided Routing. However, if source node S does not know a previous location of destination node D, then node S cannot reasonably determine the expected zone. In this case, this algorithm reduces to the basic flooding algorithm.

Mobile agents are a novel effective paradigm for distributed applications, and are particularly attractive in a dynamic network environment involving partially connected computing elements. The notion of computation mobility against conventional data mobility governs the underlying philosophy of agencies. Most research examples of the mobile agent paradigm as reported in the current literatures have two general goals: reduction of network traffic and asynchronous interaction. Some authors have suggested that agents can be used to implement network management and to deliver network services [11]. Intensive research on the “Insect-like Systems” has been done over the last few years. The mobile agent systems have been popularly simulated in close resemblance to an ant colony [12]. Of particular interest is a technique for indirect inter-agent communication, called stigmergy, in which agents populate information cache of nodes, which other agents can use. The technique of overwriting a set of information with more appropriate information has been popularly called blackboard communication [10,12]. Stigmergy serves as a robust mechanism for information sharing. In our protocol, we have used a multi agent framework that incorporates stigmergy for mutual interaction. The blackboard form of communication has also been implemented for agent-node communication.

The basic idea of using agents for topology discovery has been explored in MIT Media Lab [14] earlier with certain limitations : first, node mobility and its effect on system performance has not been quantified. Second, the information convergence (convergence of the difference between actual topology information and the topology information as perceived by a node at any point of time) and its relationship with number of agents and agent migration frequency has not been clearly defined. Third, the navigation strategies used do not ensure a balanced distribution of recent topology information among all the nodes. We have tried to overcome these difficulties. Moreover, we have defined a concept of link stability and information aging based on which a predictive algorithm running on each node can predict the current network topology based on the current network information stored at that node.

3.0 Problem Statement

Any multimedia data transfer is time-dependent, voluminous and persisting. This has correspondingly three significant demands from the performance of the network:

1. Minimal delay in routing and fault recovery.
2. Heavy traffic due to voluminous data necessitates efficient load balancing (in order to minimize delay).
3. Uninterrupted connection retention over a long period of time, indicating the importance of preventing route errors.

Although these three issues are significant in all networks, the third issue seems to be of foremost importance in any mobile environment simply because the uncertainty of connectivity is high in it. In other words, it becomes challenging in a MANET to make the connection between a pair of nodes, span over a long period of time without interruption [3,5].

The motivation for this paper is drawn from the above problem statement. We have addressed the crisis of managing an uninterrupted stable session between two nodes (through multiple routes) until the accomplishment of multimedia data transfer. This of course has meant that the source node be aware of the routes through which it communicates to the destination and even adaptively select new ones at the possibility of a route failure.

Taking a step further into the problem, we perceive that the standard reactive protocols have certain consequences in addressing the multimedia issue. Since reactive protocols are event driven, a route error would interrupt an ongoing multimedia communication and a route rediscovery is needed all over again. This increases delay and thus degrades performance. Notably, route errors are likely in mobile systems and especially in an architecture like MANET. The control packets generated in the process increase load on the system and may also aggrandize end-to-end delay. As against these shortcomings, our agent-based proactive protocol seems to resolve the issues in this context better. The multi-agent framework takes into consideration the question of proper utilization of network resources, and ways of minimizing congestion.

4.0 Description of Relevant Terms

4.1 Affinity and Stability

Affinity a_{nm}, associated with a link l_{nm}, is a prediction about the span of life of the link l_{nm} in a particular context [7]. For simplicity, we assume a_{nm} to be equal to a_{mn} and the transmission range R for all the nodes are equal. To find out the affinity a_{nm} of node n and its neighboring node m, node n gets the position of node m periodically using GPS. If R_n is the transmission range of n, and d_{nm} is the current distance between n and m which can be computed, we can predict the current distance d_{nm} at time t between n and m. If M is the average velocity of the nodes, the average worst-case affinity a_{nm} at time t is $(R_n-d_{nm})/M$, assuming that at time t, the node m has started moving outwards with an average velocity M.

Given any path p = (i, j, k, …, l, m), the **stability of p** [7] at a given instant of time will be determined by the lowest-affinity link (since that is the bottleneck for the path) and is defined as min[a_{ij}, a_{jk}, …, a_{lm}]. In other words, stability of path p between source s and destination d, η^{p}_{sd}, is given by $\eta^{p}_{sd} = \min_{\forall i,j} a^{p}_{ij}$

However, the notion of stability of a path is dynamic and context-sensitive. As indicated earlier, stability of a path is the span of life of that path from a given instant of time. But stability has to be seen in the context of providing a service. A path between a source and destination would be stable if its span of life is sufficient to complete a required volume of data transfer from source to destination. Hence, a given path may be sufficiently stable to transfer a small volume of data between source and destination; but the same path may be unstable in a context where a large volume of data needs to be transferred.

4.2 Recency

A major aspect underlying the infiltration of topology information into mobile nodes is that the information carried must be recognized with a degree of correctness. Since the agent navigation is asynchronous and there is an obvious time gap between the procurement of information by an agent from one node and its delivery by the same agent to another node, it becomes imperative to introduce a concept of recency of information. For example, let us assume two agents A_1 and A_2 arrive at node n, both of them carrying information about node m which is multi-hop away from node n. In order to update the topology information at node n about node m, there has to be a mechanism to find out who carries the most recent information about node m : agent A_1 or agent A_2 ?

To implement this, every node in the network has a counter that is initialized to 0. When an agent leaves a node after completing all its tasks at the node, it increments that counter by one. We term this counter as ***recency token***. An agent while leaving a node (after completion of all the necessary information exchange and calculations) stores the new value of the incremented recency token against the node's ID within its data structures and continues navigating. Thus at any point of time, the magnitude of the recency token of any node represents the number of times that node was visited by agents since the commencement of the network. This also implies that if two agents have a set of data concerning the same node, say node n, then the agent carrying the higher recency token value of node n has more current information about it.

4.3 Time to Migrate (TtM)

An agent visiting a node is not allowed to migrate immediately to another node. In other words, an agent will be forced to stay in a node for a pre-specified period of time, termed as **time-to-migrate (TtM)**, before migrating to another node. By controlling TtM, the network congestion due to agent traffic can be controlled. For example, if TtM = 100 msec., for a single-agent system, it implies that the wireless medium will see one agent in every 100 msec. In our simulation, it has been assumed that an agent would take 1 msec. to physically migrate from one node to another. So, in this example, the wireless medium would be free from agent traffic 99% of the time.

4.4 Average Topology Deviation

To quantify the deviation or error between the actual network topology and the network topology perceived by individual nodes at any instant of time, we have developed a metric **average topology deviation.** Let us assume that (x, y) is the actual GPS co-ordinates of a particular node A at a certain instance of time. Let (x_1,y_1) be the co-ordinate of the same node A as per the information that node P carries about it. Deviation has been defined as the distance between the points (x, y) and (x_1, y_1) assuming that both are on the same 2-Dimensional plane. Let this deviation be denoted by d_p^A. The average nodal deviation is thus the average of the deviations that node P perceives about all the other nodes in the system. Quite obviously, its own deviation will always be nil as the node P exactly knows its own location and direction of motion at any point of time. Denoting average nodal deviation as perceived by node P as ∂_p, we have

$\partial_p = (\sum_{\text{for } A = 1 \text{ to } n} d_p^A) /$ number of nodes.

Thus the average topology deviation, D, has been defined by the average of the nodal deviations for each of the nodes. Formally,

$D = (\sum_{\text{for } p = 1 \text{ to } n} \partial_p) /$ number of nodes.

Physically, D, the average topology deviation at any instant of time is a formal measure of the average deviation or error between the actual topology information and the topology information that the nodes posses about all the other nodes in the system. As a direct translation of the metric, we could say that if the average topology deviation for a particular system is x unit length, it signifies that, on an average, each node can conclusively pinpoint a circular region of x meters within which the desired node would be located. This metric has been used merely to analyze the performance issues, and nodes individually do not (rather cannot) calculate it.

5.0 Mobile Agents

5.1 Issues in Implementing the Agent Paradigm

The topology traversing could well be performed using a single agent. However this strategy fails to perform well in conditions of low transmission range where clusters get formed due to groups of nodes, moving to some spatially remote portion of the bounded region. Quite obviously, since the agent is going to be in only one of the clusters, the other clusters would have no agents at all in them although the members belonging to those clusters may be well connected amongst themselves.

The above mentioned issues cause no serious concern in the case of a multi-agent system. It might be of interest to observe that convergence does not necessarily improve with the increase in number of agents in the system. It has been observed in our previous work [13] that the optimum number of agents should be *half the number of nodes in the system.*

Another issue is to control the agent TtM(Time to Migrate) to optimise congestion. We look here into the congestion introduced in the system due to variation in TtM. Let us assume that agents would take t millisecond to physically migrate from one node to another. Let us assume that our bounded region of ad hoc operation is A , our transmission range R, the agent population P and the Time to migrate T msec. In an average case where the topology is evenly distributed over the region A, the number of areas in which agents could migrate between nodes simultaneously, without mutual interference, equals $A / (\pi .R^2)$. Now since the nodes are distributed, the agents would also be found equally distributed in each of these areas in an average case. Thus in any area the number of agents would equal P_a where,

$P_a = P.(\pi .R^2) / A$

As each agent migrates at a time gap of T milliseconds and takes only t millisecond to do so, the medium will be free from agent traffic $[(T - t.P_a) *100 / T]$ % of the time. For example, if the bounded region of operation is 1500×1000 sq. m. and R is 400 m, and P and T are 15 and 100 milliseconds respectively for a 30 node network and t = 1msec., then $P_a = 5$ (approx.) and the medium would be free from agent traffic during 95 % of the time.

5.2 Agent Navigation: least-visited-neighbor-first algorithm

The agent navigation algorithm must ensure that all member nodes of the network update themselves uniformly, irrespective of their position and state in the network. More precisely, a node on the periphery of the network topology should be equally well aware of the overall changes in the system, as any other node in the center of the topology. We have designed an algorithm, called the **least-visited-neighbor-first** algorithm, for controlling the navigation strategy of the agents. An agent invokes this algorithm on the information cache of its host node to determine its next destination. Of course the agent would only be able to hop onto a neighboring node of its host node.

On reaching a node N, an agent performs the following steps:

1. Updates information cache of node N with the information available in its information cache (we look at the information exchange protocol in the following section)
2. Selects all the nodes that are neighbors of N.
3. Determines the node (among these selected neighbors) that has the least recency-token value. This is the least visited neighbor, as perceived by the node N at that instant of time.
4. If this neighbor of N has not been visited in the last 3 visits by other agents from node N, the agent selects this neighbor as its next destination. This history information is stored in the nodes. Else, the agent selects the second least-visited neighbor, and so on. This will ensure that multiple agents from same host node do not choose the same destination consecutively.
5. After choosing the right destination, the agent updates its next_detination node id with the destination node's id, changes the history table of the host node N with this newly selected node id.
6. Increments the host node's recency token value and stores this value against the host node's id within its own data structures. The agent resumes navigation.

Thus, if we consider a host node of an agent and its neighbors to be a sub-graph, then the agent always migrates to the node, which has had the least number of agent visits among the members of this sub-graph. Since the agent navigation is a distributed protocol, we can envision that overall the agency attempts to visit all nodes in the network with equal frequency. This in turn facilitates homogeneous topology-awareness.

5.3 Handling the Event of Agent Oscillation between Nodes

Let us consider a case where a node gets isolated from the topology and as a result does not receive agent visits for a long time. Now let us assume that it gets reconnected to the network after some time. Obviously, the recency token value of this node would be much less than the values of the others, which have continuously received agents at regular intervals. Now this isolated node on getting connected to the network would obviously have a rush of agents towards it. The agents would get serviced, go to some next destination and again come back to this newly connected node until the recency value of the newly connected node is no longer the least among all the recency tokens of the other nodes in the network. This means that if a node gets isolated and then rejoins, the agents would oscillate over it and its neighbors until its recency value surpasses some other neighbor's recency value. Oscillation might cause other nodes in the network to starve of agent visits and is unnecessary from the perspective of percolating current information.

In order to eliminate this oscillation, we have incorporated the following strategy. If a node finds that it does not receive agent visits for longer than a specific span of time, it resets its recency token value to zero. An agent that visits a node, finding that the node has a recency_token value of zero recognizes that the node has performed a reset. The agent performs the standard node-agent information interchange and then assigns the average of all its recency token values to this node's recency token. This prevents the oscillation of the same agent. However other agents may independently come to this node since they might still have a recency value for this node as the least value. This is desirable in order to percolate a new node's information fast. Greater agent visits (i.e different agents visiting it) would catalyze this quick infiltration of information.

A means of evaluating the performance of the agent navigation protocol would be to examine the rapidity with which a topology change information gets propagated into the nodes of the system. We have defined a notion of percolation to test this issue. Percolation (P) has been defined as the rate at which the information regarding the entry of a new node into the system gets propagated into the existing nodes of the network. P_T is the percentage of nodes in the system that is aware of the new node entry after T milliseconds of the entry. The percolation curve in Fig. 6 indicates that the agent navigation protocol performs well even in an highly dynamic system.

5.4 Information Exchange Protocols in Node-Agent and Agent-Agent Interaction

Infiltration of partial network information into the nodes is an asynchronous process, as the agents visit the nodes asynchronously. Thus it becomes acutely necessary to develop strategies for information exchange (i.e. to accept only that information which is more recent than what the node / agent already possesses). It is a two-step process.

In step 1, the recency tokens of all the nodes stored in the information cache of the current host node is compared with the corresponding recency token of that node, stored in the information cache of the agent. If the recency token of any node, say X, in the host node’s information cache happens to be less than that in the agent’s information cache, then it is obvious that the agent is carrying more recent information about node X. So, the entire information about node X in the host node’s information cache is overwritten by that in the information cache of the agent. This step is performed asynchronously for all the agents as they arrive at that host node. This step helps the node to acquire all the recent information that it can gather from the agents. The agents however have not yet updated their data structures from the host node’s information cache. During this time if other agents come in into the node's agent queue, they perform this step as well. Thus the node's information cache get populated with all the more recent information fetched by multiple agents.

When an agent is ready to migrate (i.e. after a waiting time defined earlier as TtM), the step 2 is performed. In step 2, the agent copies the entire information cache of the host node on their own information cache. This contains the most recent information since the data set contains a combination of all the recent information that could be collected from the visiting agents and those that were already present in the node. With these updated values, the agents select their destination on the basis of the navigation algorithm previously described.

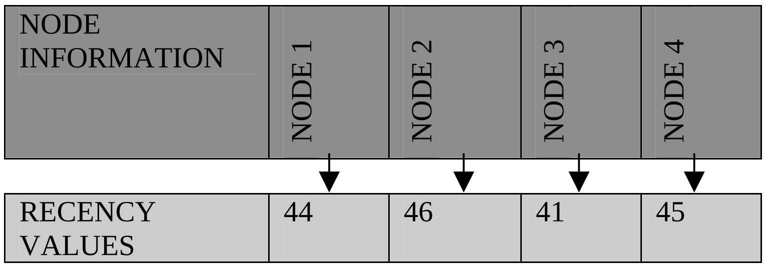

Figure 1. Representation of the data structures of the nodes and agents. Both agents as well as nodes contain this structure and their recency values are compared as explained in the above discussion.

5.5 GPS-based Location-Prediction Mechanism

The foremost characteristic of a dynamic environment is that information is never absolute. Information collected by an agent or node regarding the spatial distribution of the network is constantly aging due to the mobility of the entire system. However, the agents are proactively replenishing the cache of each node with newer or more recent information. This naturally indicates that a node would be better topology aware, if the agent visit frequency could be increased upon it. The vice versa is true as well.

In other words, we could say that each time an agent visits a node, the node gets a snapshot of the network topology which is not accurate but less inaccurate than the previous snapshot it possessed. (It must be observed that a new snapshot might not have new information for all the nodes). Now it could be said that nodes would have to remain satisfied with a snapshot of the topology until a new agent visits it. Thus information ages in between two agent-visits. Hence topology awareness degrades. However, a node can well trace the movement of other nodes through local calculations, simply because it knows their speed and direction of motion. As indicated earlier, we have used the GPS (Global Positioning System) support at each node for the extraction of geographical co-ordinates, velocity and direction of movement of each node. Use of local prediction mechanism improves topology awareness, except in the case of a node, which has changed its direction of motion. In such an event, erroneous prediction does occur but only to get emended as soon as the direction-change-information is fetched to the node by a new agent visit.

More the number of snapshots a node gets in a unit time, lesser is the information aging i.e. greater topology awareness. Moreover, to maintain optimal topology awareness in our system, each node traces the movement of the other nodes in the system, based on the velocity and direction information that it has about them. We have incorporated this mechanism in our protocol and each node locally performs a prediction at intervals of 10 milliseconds.

6.0 Uninterrupted Communication Management

We are now in a position in which each of the nodes has a local view of the network topology i.e. each node is topology aware. Also the mobile multi-agent framework is proactively replenishing the information cache of each node with fresh topology updates. This leads to a scenario where conventional route discovery is no longer necessary. More explicitly, the nodes can now determine the most stable route locally and initiate the sending of data packets through it. The way to calculate the stability of a path is indicated in section 4.1. After a point of time, if the source node finds that the chosen route has attained a low stability (indicating that it would soon cease to exist), the node computes a new, more stable route from the local information cache and redirects data packets through the later. This adaptive route selection facilitates continuous communication through multiple paths in the temporal domain. Thus we can envision that as long as two nodes remain connected, they will always be able to get at least one route through which communication can continue. In the case of multi-route availability, the best route can always be selected. Quite perceivably, the adaptive selection of best routes guarantees an uninterrupted communication session between two nodes thus ensuring multimedia data transfer to occur.

In our simulation, the source node computes the best stable route (say R) and initiates data transfer through it. Then on, at regular intervals, it invokes a modified link state algorithm over its refreshed topology information locally to re-compute the best stable route. It must be observed that due to the mobility in the network, the source node after some time would find that the best stable route (R) has changed; a new route (R_1) would be more stable. Our protocol redirects data packets through this new route R_1. The process continues until completion of data transfer or when a route error occurs.

From the above discussion we see that a node initiates and even redirects message communication through appropriate routes without consuming any network bandwidth. However, the analysis of the mechanism would be incomplete unless it is discussed in the context of maintaining a stable QoS. Let us thus consider the occurrence of route errors.

A route error can occur when either the perceived route does not exist at all or when the perceived stability is substantially higher in comparison to the actual route stability. The later case indicates that the node is misled to believe that it would remain connected to its destination (through a particular route) for a specific span of time (T) whereas this path would actually cease earlier. Now both these situations would be the result of inaccurate topology information. In other words, if the node’s topology information is significantly incongruous with the actual topology of the ad hoc architecture, the deviation of the perceived path and the perceived stability would be proportionally deviant from the actual path and stability, thus increasing the route error frequency.

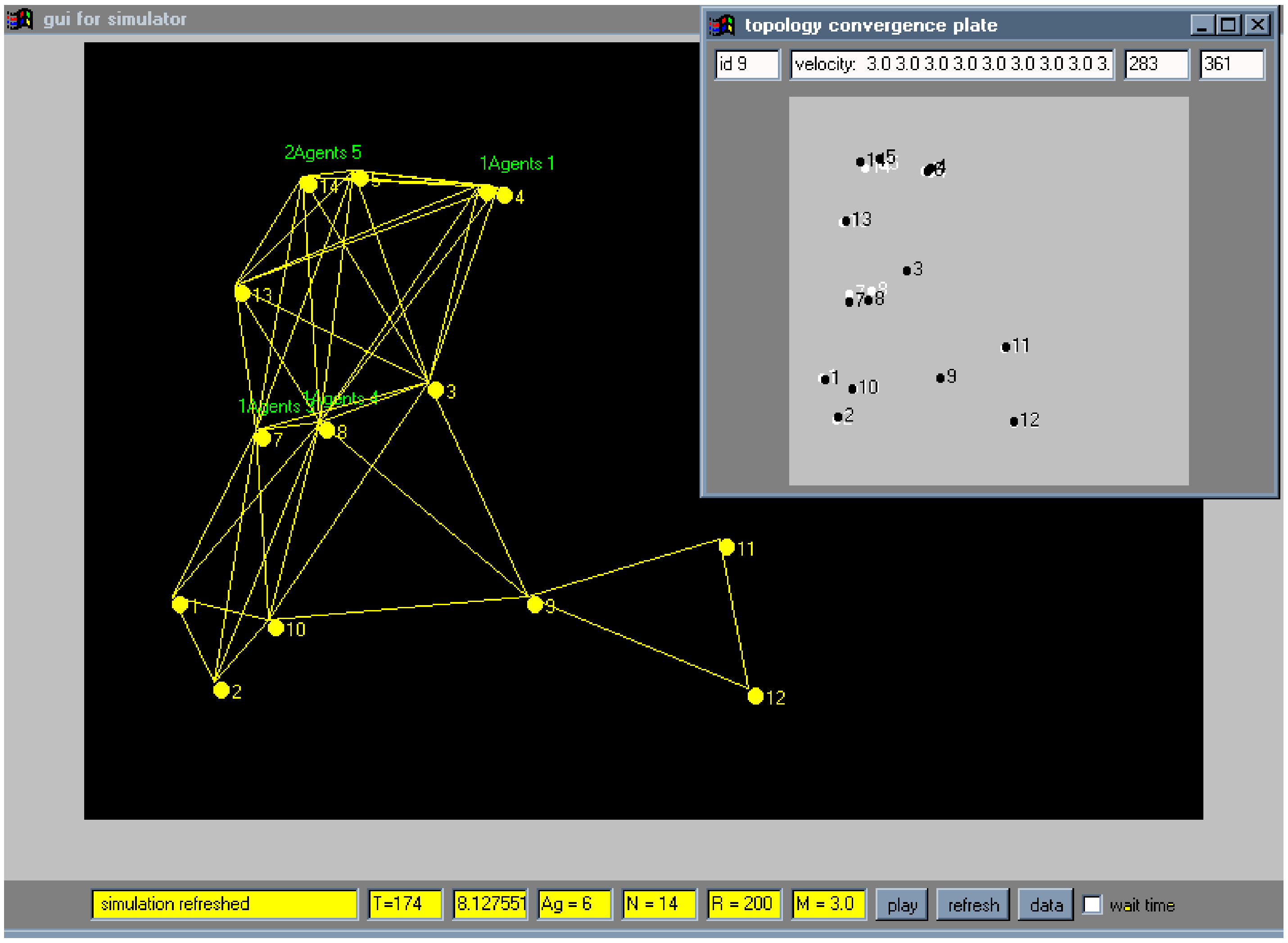

Fig. 2.This is a snap-shot of the simulator running with a set of 14 nodes. The text boxes at the bottom displays the time tick, the average topology deviation, the number of agents in the system, the number of nodes, the transmission range and the mobility of the system. In the inset, the local topology perception of the node 9 is shown. This window has been invoked by clicking on node 9 during the execution of the simulator. The nodes shown in black in the inset represents the positions of the nodes as perceived by node 9 and the nodes in white depict the actual position of the nodes in the network topology. It is to be noticed that since the node 9 exactly knows its own position, the white and black representation of this node has converged. Thus only the black node can be seen.

This does not pose to be a cause of serious concern in this proposal. First of all, even if a route error occurs, it does not necessitate a route-rediscovery process. The source node merely selects another route from its route cache. Secondly, we understand that the route error frequency bears a direct relation with the average topology deviation and thus indirectly with the performance of the agent-based-framework. We will show that for a high-mobility system, the average topology deviation is near 10 percent of the transmission range for a high TtM, and 5 percent for lower TtM. This implies that even for a highly mobile system of mobility 30 m/s and with transmission range 300 m, the topology deviation would be less than 30m. Therefore, if node A perceives that node B lies at a distance of x meters from it, in reality, the average case would mean that the node is (x + 30) meters away from it along the straight line joining node A and the perceived location of node B. Thus in a high-mobility system of 30 m/s, the actual stability of the link A-B is (R – x – 30) / 30, (R being the transmission range). The perceived stability however is (R – x) / 30. The average case difference in the perceived and the actual stability is thus 1 second. In a low mobility system or a low TtM system, the average topology deviation is significantly lower and thus the stability difference between the perceived value and the actual is still lesser. If the source node takes care of this fact during adaptive route selection, the chance of route errors would be reduced.

7.0 Simulation Set-Up

We have used our own simulator for the purpose of analysis and performance evaluation of our protocols. In the simulation, the environment is assumed to be a closed area of 1000 x 1000 square meters. At the point of commencement the mobile nodes are distributed randomly over this area. Each node chooses its destination and optionally chooses its velocity. To observe the impact of mobility we have, in certain cases, assigned predefined uniform velocity to all the nodes. The simulation commences with each node moving linearly towards their chosen destination. On reaching their destinations, each node optionally waits for a random span of time, chooses a new destination and starts moving towards this newly chosen destination. The nodes announce themselves in the environment through a beacon broadcast at regular intervals. This enables each node to be aware of their neighbors at any point of time. In each of the simulation run, we have assumed the transmission range to remain constant.

The multi-agent framework is operational over this backbone of mobile nodes. During commencement, each node generates a random number locally. If the generated number happens to be even, the node spawns an agent, otherwise it does not. This indicates that the number of agents that appear within the network infrastructure is approximately half the number of nodes in the network [13]. The agents then jump asynchronously from one node to another at an interval of TtM milliseconds, carrying information that it gathers from its host. On reaching the next node, the agents exchange information with its new host, waits for TtM milliseconds and then again jumps to a newly selected node. Thus nodes that were totally unaware of the network topology at the point of commencement, begin to gather network information through the agents. Since the multi-agent framework is a proactive scheme, the nodes in the network are always kept updated with partial(or approximate) topology information.

The agent bandwidth has been assumed to be 1 agent/millisecond. The simulated clock for our simulator ticks at a unit of 1 millisecond. The simulation runs for a pre-specified span of time or until a complete data transfer from a source node to a destination node has successfully been accomplished.

8.0 Discussion

Proactive monitoring of the topology through agents enables a node to forestall a prospective route error (caused most often in a MANET due to the separation of two intermediate nodes beyond their transmission ranges) and subsequently adopt preventive measures by redirecting communication through a new, more-stable route. The real gain is significant and discussed henceforth.

In order to support real time/multimedia communication, the most momentous factor could be envisioned as un-interrupted communication management with delay minimization. Our technique ensures that two nodes can choose to maintain an uninterrupted communication session for almost an indefinite span of time, unless they get clustered or confront a route error due to an intermediate node suddenly switching off. This means that a huge stream of data could be packet-switched, with the stream flowing through appropriately selected channels in the temporal domain. Thus what seems to be a free flowing stream of continuous data packets is actually an outcome of zero-latency, periodic, route computation, performed locally and almost parallely with message transfer. The net effect is apparent un-interruption.

Further, in the much rare eventuality of a route error, communication does get interrupted only to resume almost instantaneously. The delay in resuming communication equals only the time to discover the error and is much less in comparison to conventional protocols.

Additional to the advantage of uninterrupted communication and minimized delay, the consumption of network capacity gets significantly excised. The generation of control packets in accomplishing route discovery in conventional systems is a serious overhead. This is successfully bypassed in this proposal, as the media capture of agents is substantially lower in comparison.

8.1 Implicit Load Balancing

Another issue of relevance in this mechanism is that, in transporting data from the source to the destination, a route is used only for that span of time for which it poses to be the best route. Now, we have used link stability and number of hops as parameters that define the quality of a route. This means that a route that might be sufficiently stable, would still get discarded in the event of a better-route availability. The advantage in doing this is that no link is held indefinitely, causing the network resources to be shared amongst the prospective claimants. Now, the consideration of the link load as one of the contributing parameters to decide upon the best route would result in to an automatic load balanced network. In other words, paths that are less stable would no more be discarded by source nodes simply because they would in most cases be less loaded as well. Similarly, a route, however stable it may be, would not be selected for communication if it happens to be highly loaded as well. Presently, we are working towards developing a cost function that would quantify the best route in terms of the relative importance of these three parameters, namely: stability, number of hops and link-load.

8.2 Protocol Performance

We have presented the performance analyses of our simulation in two steps. We have first presented the performance of the agent system and the corresponding protocol. And then we have shown the application of this agent-based scheme in the context of managing an uninterrupted connection.

The central parameter, which we have used to evaluate the agent-based protocol for our simulation is average topology deviation. Quite understandably, the magnitude of average topology deviation varies inversely with the degree of accuracy with which a node would be able to pinpoint the geographical location of any other node in the system. Additionally, to examine the efficacy of the prediction mechanism, we have performed the simulation under a non-prediction mode as well. We have termed this non-prediction mode as the basic protocol and this happens to be the experimental baseline for our experiments.

In fig 3, we have shown the variation of average topology deviation against time for high and low TtM, on the basic protocol. From the graph it is evident that the deviation quite naturally decreases with decrease in TtM. This is in harmony with our expectations simply because it supports our analysis that a node would be better topology-aware when the agent visit frequency is increased upon it. We find that this system characteristic is maintained in all cases, i.e. where several other parameters are varied.

In fig. 4, the comparison of the predictive protocol and the basic protocol has been presented for a high mobility model and high TtM value. The substantial difference in the average topology deviation supports the efficacy of the predictive protocol. We find that the average topology deviation in the worst case of high mobility and high TtM does not exceed 10 percent of the transmission range. This argument is further bolstered by fig. 5, which plots average topology deviation against time for varying mobility. In all the graphs we see that in the worst case analysis of high mobility and high TtM, the deviation is no greater than 8 percent of the transmission range. Further the deviation decreases with lower mobility values. Thus we conclude from the above discussion that the predictive mechanism ensures a better degree of topology awareness in comparison to the basic protocol.

In fig. 6 we present the effect of lowering TtM in a highly mobile system in order to highlight the best performance achievable in a worst case scenario. As against the high TtM (=100) performance we see that the deviation significantly improves to less than 5 percent of the transmission range for TtM = 50, indicating that agent migration frequency could be used as one of the key regulatory factors in maintaining a stable QoS.

We have also defined percolation as another metric that reflects the rate at which information propagates in the network. To capture this metric we have thrown in new nodes in the network while the simulation is running and we have examined the rate at which the nodes in the system learn about this new entry. A point that requires to be mentioned is that, contrary to intuitive belief that a node entry in the center of the topology would have its information propagated much faster, than if it enters in the periphery, we find that our agent system more or less carries the new information to other nodes with equal celerity. Thus the curve in the figure 7 returns positive feedback about the agent protocol. From this graph we find that the new node entry gets propagated to all the other nodes within a span of approximately 3 seconds and the difference between peripheral appearance and central appearance is small.

In context to multimedia communication, which indirectly points to maintaining an uninterrupted session between two nodes, we have experimented on whether a node actually is able to ping a desired node for a continued span of time through a selected route. Obviously this route selection is done based on the outcome of invoking the routing algorithm on the local information cache.

In fig 8, we trace the routes in the temporal domain that a source node uses to ping its destination node. The curve shows the perceived stability of the path that the node uses to ping the destination. From the graph we see that whenever there is a new path selected, the new path is more stable than the existing path. This shows the mechanism of our protocol in which the node selects the best stable path to ping the destination.

An interesting observation is that, the stability difference between the actual path and the perceived path is never greater than 1 second. This implies that the average topology deviation is numerically equal to the magnitude of the velocity, when the stability difference is exactly 1 second. Thus for a connected graph of transmission range 300 meters and mobility 20 m/s, the average topology deviation would be less than 20, in other words less than 10 percent of the transmission range.

9.0 Conclusion

In this study, we have attempted to address the issue of managing a stable communication path between two nodes irrespective of the mobility of the system. We have developed the multi-agent-based framework to make the nodes in the network position aware. Nodes are equipped with GPS support. In addition, we have proposed a concept of local route computation and corresponding packet redirection through stable routes. In this study however, we have not assumed the case of agent losses or agent transmission-reception errors. We have also assumed that nodes move in a straight line towards their respective directions. We understand that the nodes taking curvilinear paths would require a more complicated mechanism for location prediction in our predictive protocol. However, our results for a comparatively simplistic model indicate that an adaptive multi-agent protocol could be effective in successful infiltration of topology information and corresponding implementation of local route computation. The subsequent advantages could be the initial steps towards managing a connectivity for a stable QoS in mobile ad hoc wireless networks.

References

1. Z. J. Haas, Milcom'97 Panel on Ad-hoc Networks, http://www.ee.cornell.edu/~haas/milcom_panel.html

2. J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols,' *Proc. ACM/IEEE Mobile Comput. and Network.*, Dallas, TX, Oct. 1998.

3. Sajal K. Das, A. Mukherjee, S. Bandyopadhyay, Krishna Paul, D. Saha, “Improving Quality-of-Service in Ad hoc Wireless Networks with Adaptive Multi-path Routing”, Accepted in GLOBECOM 2000, San Francisco, California, Nov. 27-Dec 1, 2000.

4. C-K Toh, A novel distributed routing protocol to support ad-hoc mobile computing, IEEE International Phoenix Conference on Computer & Communications (IPCCC’96).

5. E. M. Royer, C.K.Toh. A Review of Current Routing Protocols for Ad-Hoc Mobile Wireless Networks.. IEEE Personal Communications, April 1999.

6. C. Perkins, E. Royer. Ad Hoc On-Demand Distance Vector Routing. Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, New Orleans, LA, February 1999, pp. 90-100. Online. Available. http://beta.ece.ucsb.edu/~eroyer/aodv.html

7. K. Paul, S. Bandyopadhyay, D. Saha and A. Mukherjee, Communication-Aware Mobile Hosts in Ad-hoc Wireless Network, Proc. of the IEEE International Conference on Personal Wireless Communication, Jaipur, India, Feb. 1999.

8. Young-Bae Ko, Nitin Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks". Proceedings MOBICOM 1998, Dallas, Tx.

9. Y-B Ko, V. Shankarkumar, Nitin Vaidya, “Medium Access Control Protocols using Directional Antennas in Ad Hoc Networks” , Proc. Of the IEEE INFOCOM 2000, Tel-Aviv, 26-30 March, 2000.

10. Steve Appeleby and S.Steward. Mobile software agents for control in Telecommunications networks. *BT Technology Journal*,12(2):104-113,April 1994.

11. S. Krause and T. Magedanz, "Mobile Service Agents enabling Intelligence on Demand in Telecommunications", Proc. IEEE GLOBCOM '96, 1996.

12. Schoonderwoerd, R., et al.(1997) 'Ant-based load balancing in telecommunications networks.' Adaptive Behavior,5(2):169207,1997. http://www.uk.hpl.hp.com/people/ruud/abc.html

13. Romit Roy Choudhury, S. Bandyopadhyay, K. Paul. A distributed mechanism for topology discovery in ad hoc wireless networks using mobile agents. Proc. of the First International Workshop on Ad Hoc Networks, ACM/IEEE MobiHoc 2000, in conjunction with MobiCom 2000, Boston, USA.

14.

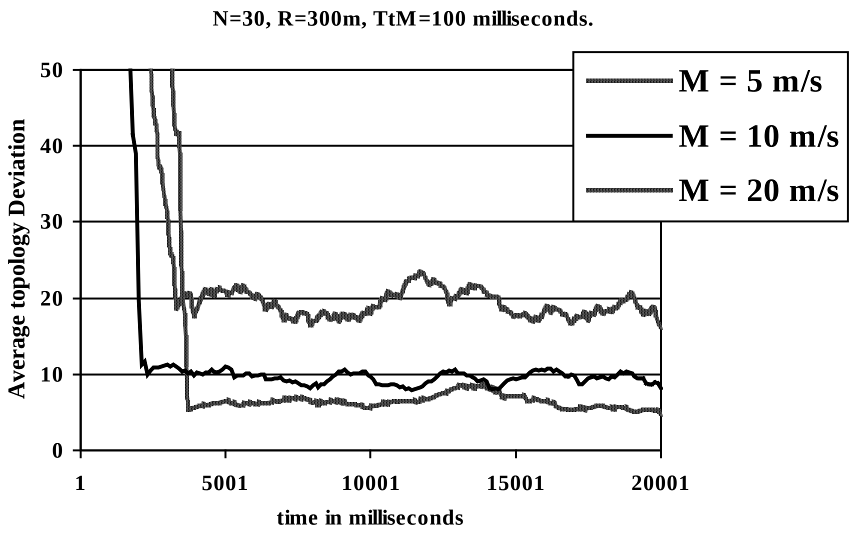

Fig 3. Evaluating the effect of altering TtM on protocol performance. The two plots indicate the improvement in performance caused due to the lowering of TtM on the **BASIC** protocol. In a system with m = 10 m/s, and for a low TtM (50 ms), the average topology deviation is approximately 3 % of the transmission range (R).

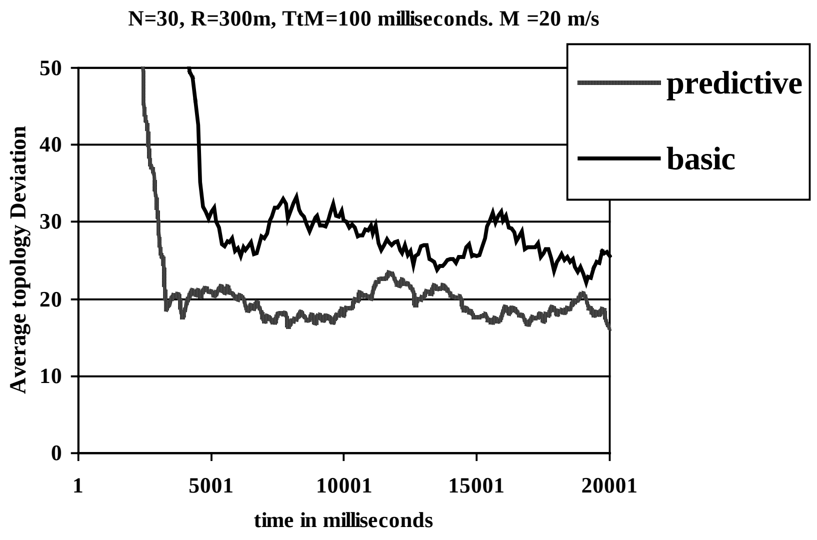

Fig 4. Comparative study of the predictive and basic protocols for average topology deviation in a high mobility system. It is evident from the graphs that the predictive protocol performs better. The performance in the above plot is for TtM = 100 milliseconds. Lowering TtM would further improve performance as is evident from the previous plot.

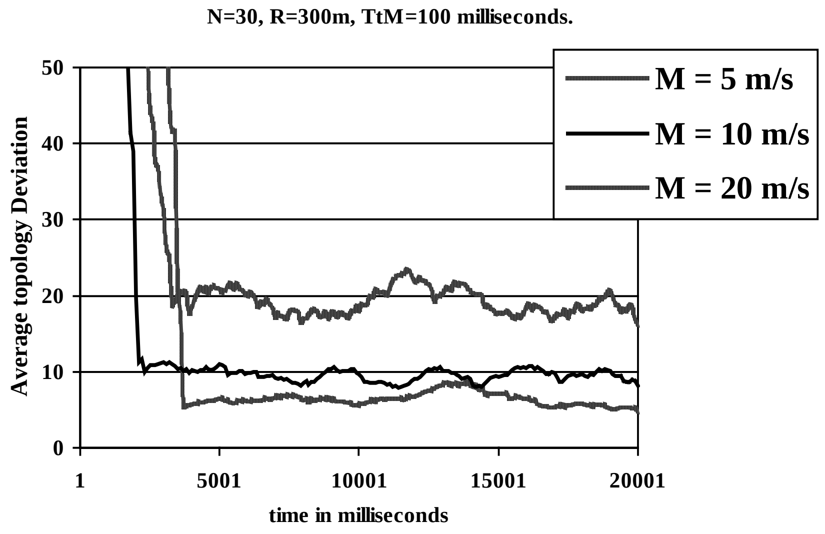

Fig 5. Variation of average topology deviation against time for the predictive protocol for varying mobility and constant TtM. Quite expectedly, the performance improves in the case of less dynamic systems.

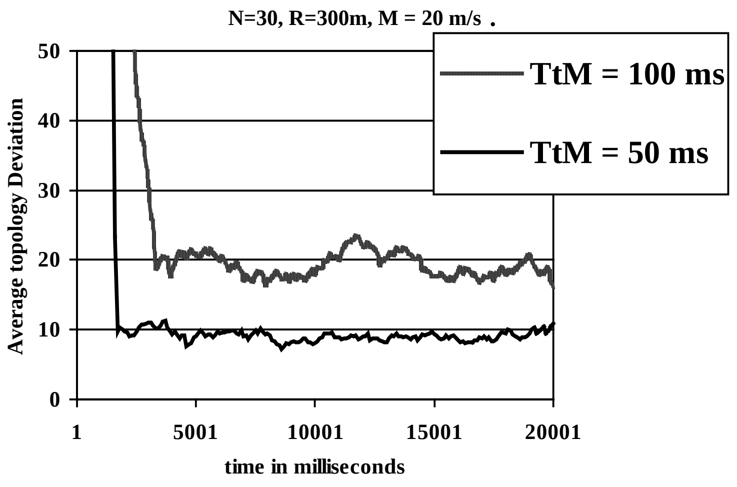

Fig 6. Effect of TtM on performance of the predictive protocol in the worst case i.e. a high mobility model, for a constant number of nodes and a constant transmission range of 300m. Lowering TtM is a positive measure for improving performance to less than 5 % of the transmission range (R).

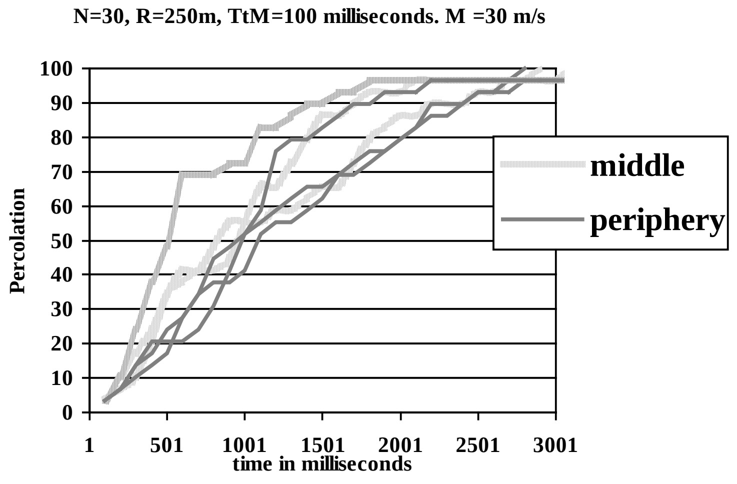

Fig. 7. The percolation of the agent-based protocol shown for node entries at the periphery as well as in the center. Quite evidently from the figure, the difference in the rate of information propagation is not high. Percolation (plotted in the Y axis) is the percentage of total number of nodes that is aware of the new node's entry into the system.

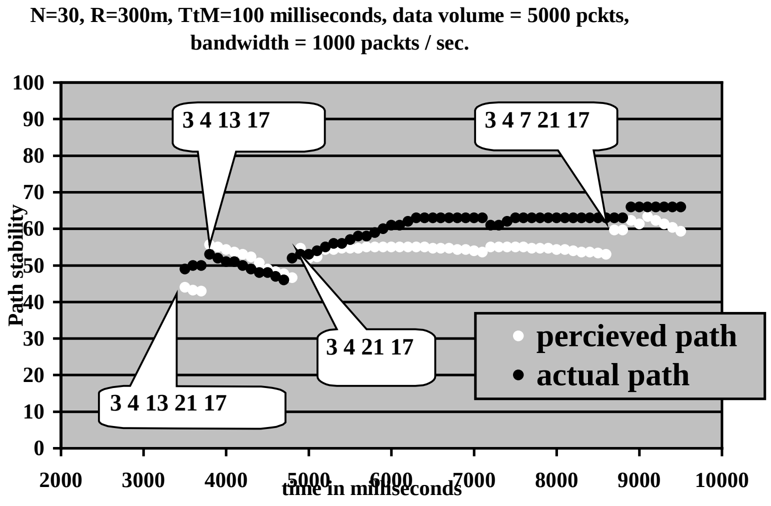

Fig 8. The selection of best-stable paths between a pair of nodes, shown against time. The paths between node 3 and node 7 have been shown inside the call-outs. The perceived path as against the actual path can be compared from the graph in terms of their stability values. The TtM value is 100 ms for a mobility of 20 m/s.