JCS Vol. 1 No. 3 Nov - Dec 2005

History-Aware Anticipated Route Maintenance in Mobile Ad Hoc Networks

M.V. Sivannarayana[1], M.P. Sebastian[2]

ABSTRACT

In a mobile ad hoc network (MANET), for hosts that are not within each other's range, multi-hop routes must be established. Because of the node movement, the routes can fail randomly. A host that was part of a route may move away from its upstream and downstream partners, thus breaking the route. The Anticipated Route Maintenance (ARM) protocol decreases the link failures and enforces route maintenance. The major drawback of the ARM is its control overhead. In this paper, we investigate methods to reduce the control overhead of ARM without compromising on other performance metrics. We use the History-Aware Multi-path Routing (HAMR) algorithm to enhance the ARM. We call the resulting protocol as History-Aware Anticipated Route Maintenance (HAARM) protocol. We compare its performance with the other protocols and observe that HAARM reduces the control overhead in ARM, keeping the other performance metrics intact.

KEY WORDS : Ad hoc mobile networks, history-aware anticipated routing, multi-hop routing, route maintenance

1. INTRODUCTION

A number of routing protocols [12] have been proposed for ad hoc wireless networks, which are proactive and reactive. Proactive protocols, also called table-driven protocols, attempt to continuously determine the network connectivity so that the route is already available when a packet needs to be forwarded. Examples include the Destination-Sequenced Distance Vector (DSDV) [13, 14, 8] protocol, Wireless Routing Protocol (WRP) [15], Temporally-Ordered Routing Algorithm (TORA) [9] and Lightweight Mobile Routing (LMR) [16] protocol. The advantage of proactive scheme is that when a route is needed, there is little delay until the route is determined. However, purely proactive schemes are not appropriate for reconfigurable wireless ad hoc network environment, as they continuously use a large portion of the network capacity to keep the routing information current.

Reactive protocols, also called on-demand protocols, on the other hand, invoke a route determination procedure only on demand. Examples include the Ad hoc On Demand Distance Vector (AODV) [10, 11] protocol, Dynamic Source Routing (DSR) [3, 17] protocol and Anticipated Route Maintenance (ARM) [1, 4] Protocol. In these protocols, when a route is needed, some sort of global search (e.g., flooding) procedure is employed. Because route information may not be available at the time a route request is received, the delay to determine a route can be quite significant. Furthermore, the global search procedure requires significant traffic, making pure reactive routing protocols not suitable for real-time communication.

Finding a route in wireless networks requires considerable resources (time, bandwidth, and power). Based on the anticipation of the failure of routes, the ARM [1, 4] has been proposed that allows routes to have longer

---

[1]Department of Computer Engineering, National Institute of Technology Calicut, Calicut, Kerala - 673 601. India.

[2]Department of Computer Engineering, National Institute of Technology Calicut, Calicut, Kerala - 673 601. India.

E-mail : [1]muppalla999@rediffmail.com, [2]sebasmp@nitc.ac.in

route lifespan without interrupting service by replacing unsafe link(s) with stable one(s). The approach is best integrated with reactive routing algorithms for hosts that have geographic location information [5, 6]. In ARM, a predefined value T (expected time to find replacement link(s) for an unsafe link) is a very important parameter. ARM improves the packet delivery and reduces the end-to-end delay, but it increases the control overhead in a network. This control overhead is due to the continuous route maintenance and to anticipate the route failures. In this paper, we propose the History-Aware Anticipated Route Maintenance (HAARM) to reduce the control overhead in the network. HAARM is a combination of Anticipated Route Maintenance [1, 4] and History-Aware Multi-path Routing [2].

We compare the performance [7] of the three protocols, HAARM, ARM and DSR [3, 17] through simulation. We show that HAARM/ARM improves the functionality of DSR by preventing the links in the route from breaking, and improve the packet delivery ratio using route maintenance. The simulations also show that both HAARM and ARM achieve noticeable improvements in dropped packets and links stability over DSR, even though there is more traffic and channel overhead in HAARM/ARM. Choosing the right value for T, which depends on the node density in the network and reflects the network sparseness, could control the channel overhead.

2. THE HISTORY-AWARE ANTICIPATED ROUTE MAINTENANCE PROTOCOL

HAARM combines the advantages of ARM and HAMR. HAARM is not a stand-alone protocol; it must work with a reactive routing protocol. We run HAARM over DSR [3, 17] with bi-directional links and using the geographical information at each node. Each node must have the additional information such as its location, speed and direction of movement, an extended routing cache to include location and speed information for other nodes, and a list to keep information about the neighbor nodes within its transmission range.

Most routing protocols rely on the MAC layer protocol (e.g., IEEE 802.11), which depends on explicit acknowledgments to ensure a packet delivery. Lost acknowledgments are the key for the MAC layer protocol to detect failures. Thus, when the node does not receive any acknowledgment from the next node within a given period of time, the link will be treated as broken and the recovery system will start finding a new route. This is an expensive solution in terms of channel overhead and network reliability. Thus, the route failure recovery is triggered by lost acknowledgments, which imply lost packets, and the recovery may take place after the actual link failures. HAARM introduces a better solution by preventing the links from failures.

HAARM uses two functions: expand and shrink. We use $p > q$ to indicate that node q appears after node p in the path. For any neighbor nodes, b_i and b_{i+1} in a path, b_i is called the upstream of b_{i+1} and b_{i+1} is the downstream of b_i. Note that every bridge node, b_i, has exactly two neighbor nodes, b_{i-1} and b_{i+1}, in the path. Now we introduce the main features of the History-Aware Anticipated Route Maintenance protocol.

2.1 Choosing the *T* Value

The HAARM protocol runs periodically at every node. First, it will calculate the expected time that the link between the nodes that may break. Then, based on whether this time is less than or equal to the predefined value T (expected time to recover from failure, in the worst case), the HAARM-expand function executes and finds a bridge node before a failure, without interrupting the service. Suppose we choose a large value for T, then the network is assumed to be sparse and it takes substantial amount of time to find a new bridge. So HAARM-expand will be active most of the time, maintaining the routes from failures. When we choose large T value, then time to fail (TTF) of all hops will be less than T and HAARM-expand will perform the maintenance for each hop. If we choose a small value of T, then no TTF will be less than T, and so HAARM-expand will not run this time (then the maintenance procedure will be DSR). Note that the actual time to recover the route is not easy to measure. However, we can use HAARM-expand to measure the actual time to recover a route.

Then the question arises: why we do not assign large values for T and make the ARM always performing the route maintenance? If we choose a large value for T, then the ARM will expand the routes that are stable, increasing the path length (by adding redundant nodes). In addition to performing maintenance, the ARM needs to send control packets, which can increase the channel overhead. On the other hand, if we choose a very small value for T, then ARM may stop performing the expand function in most of cases, and no performance improvement and no failure avoidance can be expected. Thus, another motivation of this work is to find an optimum value for T through computer simulation.

2.2 The Expand Function

If two adjacent nodes in a route move out of each other's transmission range, then the link is broken, and so is the route. Let p and q be upstream and downstream nodes of an unsafe link in a route from source s to destination d (the route is s, p, q, d), respectively. The expand function prevents the route from breaking. To do this, the protocol must find additional bridge nodes between p and q before the break occurs and adjust the route accordingly. Suppose, the link connecting node p and node q, $p > q$, becomes unsafe at time t_u and is estimated to break at time t_b. Since each node has a table, which contains information about all neighbor nodes, node p selects a bridge node r from within the transmission range of both p and q at time t_b. Then, just before the disconnection of the link, node p informs node r that r has been chosen as part of the route. Since q is still in the range of p, it listens to the notification from p to r, and prepares to receive the packets from r. Therefore, after q moves to position q_1, the new route will be s, p, r, q at q_1, d.

If there are no nodes in the common intersecting area of p and q, then finding the bridge nodes is difficult. In such cases, the protocol selects two bridge nodes to form a path to node q. Let us call these nodes b_1 and b_2. Let b_1 be in the transmission range of p and p will broadcast a control message to all nodes in its range requesting to reply if they have a neighbor b_2 that will be in position to talk to q. If there are nodes b_1 and b_2 to fulfill these requirements, a bridge path p, b_1, b_2 is formed that keeps the link alive. Node p then notifies b_1, which in turn notifies b_2, of the new link. If p is unsuccessful in fixing the anticipated failure, p has the option of notifying another node j, where $j > p$, of the failure and allowing it to execute the expand algorithm.

2.3 The Shrink Function

The shrink function handles the case when the bridge nodes in a long path have come close to each other, allowing a possible shortcut. This optimization is useful in general, but also serves to help prune paths that may become long due to the expand phase. In order for the shrink phase to work, nodes must exchange some route table information when they come within the range of each other. Specifically, when two nodes n_i and n_j come within the range of each other they will exchange control packets that contain the routes they are part of, and their position in the routing order. If the nodes discover that they are both part of the same path then they are called friend nodes. There are two cases for shrinking the path. The easiest case is when the two (and only two) nodes discover that they are friend nodes at the same time. If n_i and n_j are friend nodes they can shorten the path by directly routing from one another (skipping n_{i+1}, $n_{i+2} \cdots n_{j-1}$). A more pathological case exists when several nodes determine that they are all friend nodes at the same time. If there are f friend nodes, we could pick $(f^2-f)/2$ different short cuts. However, by choosing the node, which is earliest in the path and making a route to the node latest in the path, will be the best short cut.

2.4 Incorporating History Awareness in ARM

History-Aware Multi-path Routing (HAMR) employs session history as one of the routing metrics and provides disjoint paths by permitting multiple replies to a route query, based on session history. To enable history-aware routing, session history must be maintained at each node. During a route discovery process, a route query message transports the session histories of the intermediate nodes to the destination. Thus, the destination node can determine the route considering the session histories. Each node keeps the record of active session information and past session information. Active session information contains the information about communication sessions in which the node is currently involved. Past session information contains the information about communication sessions in which the node was involved in the past. For a node, the active session information is created and maintained when it is part of a route. When a node deletes all the route information due to a route failure or completion of a communication session, it updates its session history. Now, the corresponding active session information is cleared, and the start time and duration of the session are registered in the past session information.

Session history is quantified to use as a route selection criterion. We name the quantified history as "history degree". If a session is currently with active status, the node will be in a relatively low movement. In addition, if a communication session remains active for more than certain duration, it is likely to be more long-lived. Thus, nodes with active sessions will have higher history degree. Path selection depends on the session history. When there is a need to find a route for transmission, a route with high session information is selected for transmission.

3. IMPLEMENTATION

Our simulation model uses ns-2 [18]. A hypothetical network is constructed for the simulation purpose and then monitored for different number of parameters, such as the number of mobile nodes, movement speed, and T. Our model ranges from 10 to 50 mobile nodes, with increments of 20 nodes. The speed ranges from 5 to 70 km/h. Each mobile node in the MANET is assigned an initial position within the simulation dimension (670×670 meters). The simulation is done for 300 seconds in every run. Nodes are normally distributed when initialized and the initial position for the node is specified in a movement scenario file created for the simulation using a feature supported by ns-2. The nodes randomly move within the simulation area. During the simulation, every node is in a continuous movement (i.e., zero pause time) to resemble the realistic, highly dynamic network to evaluate the HAARM protocol.

During the simulation, each node can send and receive packets at random. However, a node can send a packet at any specified time scheduled by the simulator but cannot receive while it is in the sending mode. Each node is equipped with a single transceiver radio antenna that can either send or receive packets at a time, but not both.

With a bandwidth of 2 Mbps, this has no lagging effects to our simulation results because the node traffic generator used here is typically constant bit rate (CBR). It is programmed to generate four packets every second with maximum packet size of 4 KB. Thus, the node transceiver can be busy sending its own packets only for 64 ms per second. Most of the time, a node will be listening to the media to receive and forward packets in the network. The radio transmission range for a node is limited to a circle of radius 50 meters, and the MAC layer uses IEEE 802.11 protocol with carrier sense multiple access with collision avoidance (CSMA/CA) protocol to avoid the hidden terminal problem. The simulation is event-driven, and all the simulation events will be logged in a trace file and then processed to extract the needed statistics.

In this simulation, we evaluate the performance of HAARM, ARM and DSR using the following metrics:

1. Number of dropped packets (which is an indirect measure to the number of disconnections, occurred between the nodes during the simulation).
2. Number of dropping nodes (which measures the actual number of nodes that show a disconnection during the data transmission).
3. Average path length in hops (to measure the effect of HAARM, ARM and DSR on the path length and the packet delay).

4. SIMULATION RESULTS

In the first part of the simulation, the number of dropped packets for HAARM, ARM and DSR are shown. The simulation is done for 50 nodes, 30 nodes and 10 nodes. Fig. 4.1 shows the number of packets dropped Vs speed for 50 mobile nodes. Fig. 4.2 shows the number of packets dropped Vs speed for 10 mobile nodes. One can observe the effect of changing node speeds on the number of dropped packets for HAARM, ARM and DSR. As the speed increases the number of dropped packets also increases, but both HAARM and ARM show improvement with increased value of *T* (the expected time to find a new bridge node). This is because the expand function in HAARM / ARM will fix the route and insert bridge nodes if the link between upper and lower streams is detected to break within a period of time smaller or equal to *T*.

If we assign a large value for T, then the function will run most of the time and expand the routes by inserting the bridge nodes (because all links will satisfy the condition TTF < *T* always). On the other hand, smaller value of T will prevent the expand function from running (i.e., TTF > *T* always) and then the actual maintenance protocol becomes the DSR (this will lead curves for HAARM/ ARM to come close to DSR because our base protocol for HAARM/ ARM is the DSR). The shrink function in HAARM/ ARM has no effect on the number of dropped packets in the network, because it's only function is to reduce the path length by pruning any unnecessary nodes and shorten the path length (this means that the shrink function will not shorten any path if the new path is in unsafe state).

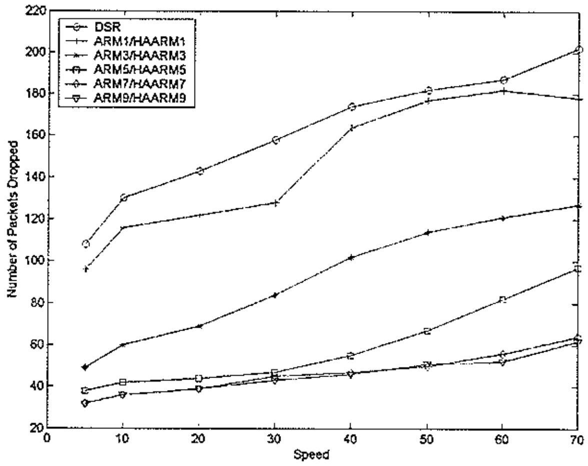

Fig. 4.1. Dropped packets (with T=1, 3, 5, 7 and 9) for HAARM/ ARM and DSR with 50 mobile nodes.

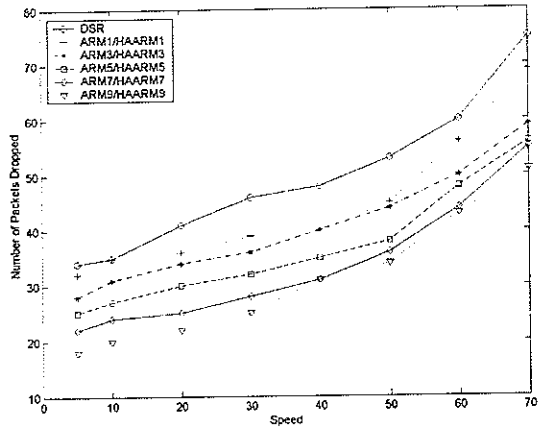

Fig. 4.2. Dropped packets (with T=1, 3, 5, 7 and 9) for HAARM/ ARM and DSR with 10 mobile nodes.

We now calculate the number of packet dropping nodes. The number of packet dropping nodes for HAARM, ARM and DSR are as shown in Fig. 4.3 with 50 nodes and Fig. 4.4 with 10 nodes, by varying the node speed and T. One can observe that the larger is the number of mobile nodes, the smaller is the ratio of packet dropping nodes relative to the total number of nodes, for all the protocols. For example, with HAARM / ARM when the number of nodes N=10 at speed 70 km/h and T=1, we see that 9 out of 10 are dropping nodes, i.e., 90% of the nodes have dropped at least one packet, while 10 out of 10 (100%) are packet dropping nodes with DSR. By changing N to 50 nodes with the same configuration as above, all protocols show improvement in dropping nodes number, DSR shows 34 out of 50 (66%) while HAARM / ARM shows 33 out of 50 (68%). But if N=50 and T=9, with the same speed, the HAARM / ARM has only 11 out of 50 (22%) are packet dropping nodes. Sowith the same configuration, HAARM/ ARM reduces the packet dropping nodes by more than 300% relative to DSR for larger values of T. Also, we observe that the difference between the curves becomes smaller as the number of nodes in the network decreases, i.e., the separation between the DSR curve and the HAARM/ ARM curves reduces and they even intersect.

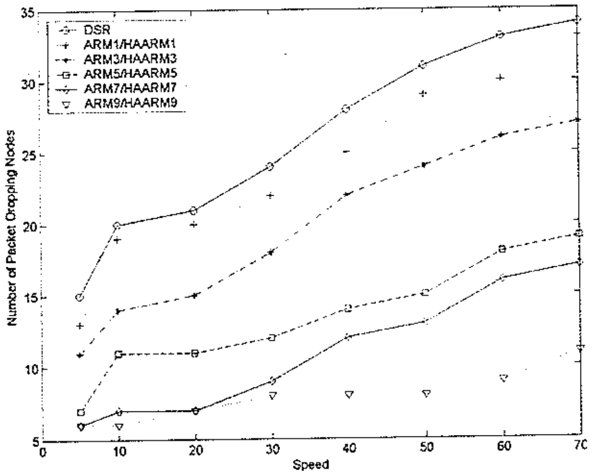

Fig. 4.3. Number of packet dropping nodes (with T=1, 3, 5, 7 and 9) for HAARM/ ARM and DSR with 50 mobile nodes.

In the second part of the simulation, we examine the effect of HAARM/ARM on the average path length in hops between the sender and receiver nodes. As discussed before, the expand function lengthens the path between the sender and receiver by adding more hops, while the shrink function tries to shorten the route by trimming the redundant hops. Fig.4.5 and Fig. 4.6 depict the average path length Vs node speeds with 50 nodes and 10 nodes, respectively.

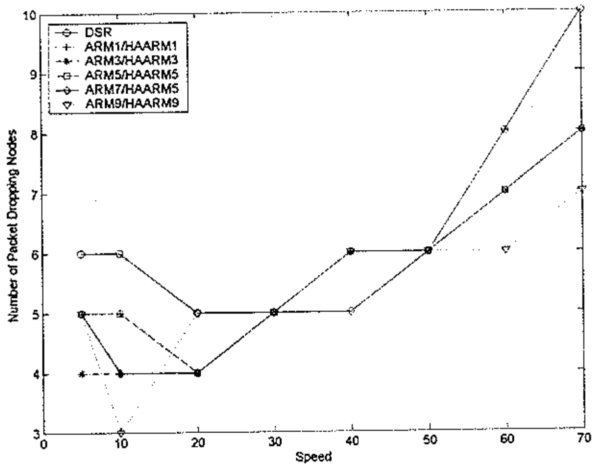

Fig. 4.4. Number of packet dropping nodes (with T=1, 3, 5, 7 and 9) for HAARM/ ARM and DSR with 10 mobile nodes.

One can observe that increase in number of nodes increases the path length dramatically. This is because as the number of nodes increases, the expand function will find many nodes available within the transmission range of the upper stream node to get inserted in the path with the lower stream node. So, more nodes are likely to be pushed into the route to avoid the service failure. On the contrary, with smaller number of nodes, the upper stream node is unlikely to find available nodes within its transmission range to fix the route and hence the path length may not be affected (but lead to higher probability of link failures). Another observation is that the speed increase of the mobile nodes is causing the increase of the average path length. This is because the probability for the link TTF to exceed the time to recover is high, and so the expand function is likely to push bridge nodes continuously.

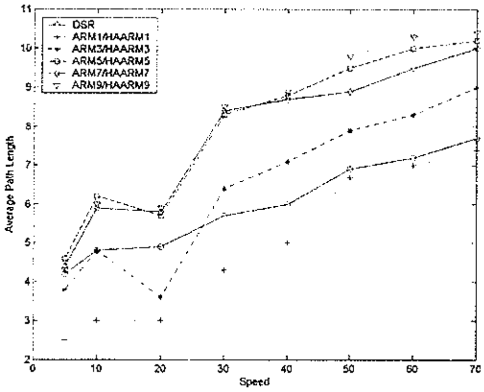

Fig. 4.5. Average path length (with T=1, 3, 5, 7 and 9) for HAARM/ ARM and DSR with 50 mobile nodes.

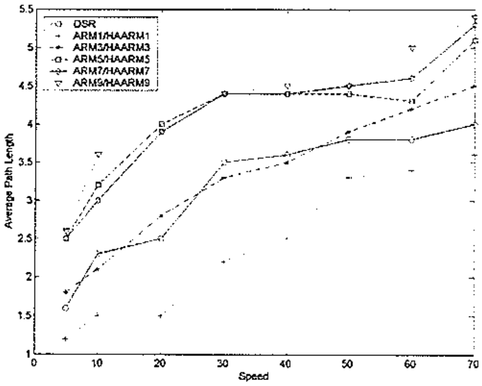

Fig. 4.6. Average path length (with T=1, 3, 5, 7 and 9) for HAARM/ ARM and DSR with 10 mobile nodes.

Fig. 4.7 compares the control overhead in the network using HAARM, ARM and DSR protocols. We make this comparison for an optimal value of T=9. Obviously, the ARM introduces more control overhead in the network. Clearly, the DSR is the winner and HAARM stands in between. Thus, HAARM reduces the control overhead of ARM, keeping the other performance metrics intact.

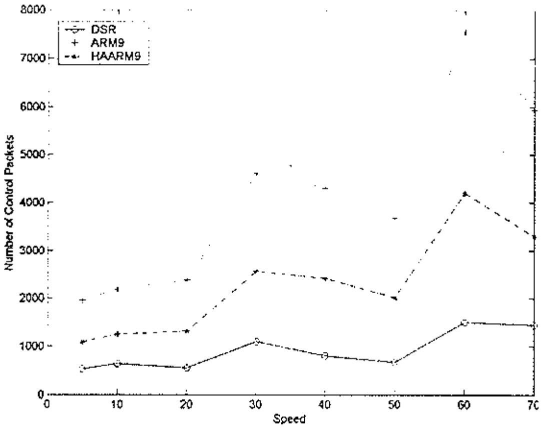

Fig. 4.7. Control overhead (with T= 9) for HAARM, ARM and DSR with 50 mobile nodes.

5. CONCLUSIONS

In this paper, we investigate methods to reduce the control overhead introduced by the ARM protocol in ad hoc wireless networks. We achieve this by implementing the history-aware multi-path routing, which uses the session history as one of the route selection metrics. The resulting new protocol is called as HAARM. We compare the performance of HAARM with ARM and show that the former is with less control overhead. We also show how HAARM improves the functionality of DSR by preventing the links in the route from breaking. The effect is more visible at high mobility speeds rather than at low speeds, and at heavy data traffic load rather than at light load. One weak point of HAMR is that if nodes in the network do not have enough session history information, then the history-aware routing may become ineffective.

Future works include development of routing techniques to compensate for the weaknesses and to further reduce the control overheads in the network.

6. REFERENCES

[1] M. Al-Shurman, S. Yoo, and S. Park, "A Performance Simulation for Route Maintenance in Wireless Ad Hoc Networks," *in Proc. 42nd annual Southeast Regional Conference*, ACM Press, pp. 25-30, 2004.

[2] S. Kim and Sunshin, "A History-Aware Multi-Path Routing in Mobile Ad-Hoc Networks," *in Proc. ICOIN*, pp. 662-671, 2003.

[3] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networking," in *Mobile Computing*, T. Imielinski and Korth Eds. Kluwer Academic Publishers, 1996, pp. 153-181.

[4] S. Park and B. Van Voorst, "Anticipated Route Maintenance (ARM) in Location-aided Mobile Ad Hoc Networks," *in Proc. IEEE GlobeCom*, pp.2809-2813, Nov. 2001.

[5] Y. Ko and N. H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," *in Proc. 4th Annual ACM/IEEE Int. Conf. Mobile Computing and Networking*, pp. 66-75, 1998.

[6] Y. Ko and N. H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," *Wirel. Netw.*, Vol. 6 , pp. 307-321, April 2000.

[7] S. R. Das, R. Castaneda, and J. Yan, "Simulation-Based Performance Evaluation of Routing Protocols for Mobile Ad Hoc Networks," *Mob. Netw. Appl.*, Vol. 5, pp. 179-189, March 2000.

[8] G.S. Malkin and M.E. Steenstrup, *Distance-Vector Routing in Communications Networks*, London: Prentice Hall Intl. (UK) Ltd., 1995.

[9] V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," *in Proc. Sixteenth Annual Joint Conf. IEEE Computer and Communications Societies*, pp. 1405-1413, 1997.

[10] C. E. Perkins, S. R. Das, and E.M. Royer, "Ad Hoc On-Demand Distance Vector (AODV) Routing," Internet-Draft, *IETF*, 2000.

[11] C. E. Perkins and E.M. Royer, "Ad-Hoc On-Demand Distance Vector Routing," *in Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pp. 90-100, 1999.

[12] D. Johnson, "Routing In Ad Hoc Networks of Mobile Hosts," *in Proc. IEEE Workshop on Mobile Computing Systems and Applications*, pp. 158-163, 1994.

[13] A. Boukerche, S. K. Das, and A. Fabbri, "Analysis of A Randomized Congestion Control Scheme with DSDV Routing in Ad Hoc Wireless Networks," *J. Parallel Distrib. Comput.*, Vol 61, pp. 967-995, July 2001.

[14] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing for Mobile Computers," *in Proc. Conf. on Communications Architectures, Protocols and Applications*, ACM Press, pp. 234-244, 1994.

[15] S. Murthy and J. J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks," *Mob. Netw. Appl.*, Vol. 1, pp. 183-197, Feb. 1996.

[16] M. S. Corson and A. Ephremides, "A Distributed Routing Algorithm for Mobile Wireless Networks," *Wirel. Netw.*, Vol. 1, pp. 61-81, Jan. 1995.

[17] J. Broch, D.B. Johnson, and D.A. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks," Internet-Draft, *IETF*, Oct. 1999.

[18] http://www.isi.edu/nsnam/ns

Authors' Biography :

M.V. Sivannarayana received the B,Tech degree from Nagarjuna University in 2002 and M.Tech degree from NIT Calicut in 2005. Currently he is with Verizon Data Services, Chennai. His research interests include routing and security issues in MANETs.

M. P. Sebastian received the B.Tech degree from Kerala University in 1984 and the ME and PhD degrees from the Indian Institute of Science, Bangalore in 1990 and 1997, respectively. Currently he is the Professor & Head of the Department of Computer Engineering at National Institute of Technology Calicut. He is a referee of two international journals. His research interests include mobile networks, cryptography, and network security.