A Novel Approach to Source Routing for Multi-hop Ad Hoc Networks

E.Papapetrou, F.-N.Pavlidou

***Abstract*— The popularity of wireless ad-hoc networks has significantly increased over the past years. As a result, a great scientific effort has been made to develop and implement efficient routing protocols, able to cope with the stochastically varying topology of such networks. In this paper we will present an efficient routing protocol that combines source routing, caching of routes and the use of sequence numbers to alleviate the routing load and at the same time increase the successful delivery of data packets.**

***Index Terms*— ad hoc networks, source routing, sequence numbers**

I. Introduction

Wireless mobile networks that operate without the need of a fixed infrastructure are widely known as ad-hoc networks. Due to recent technology advances, their penetration to markets worldwide has significantly increased over the last years. Ad hoc networks consist of mobile hosts that move randomly in and out of each others communication range. As a result, connections between nodes are prone to sudden failures and the graph of the formatted network varies stochastically. It is clear that the aforementioned context encumbers routing. Therefore, the choice of a suitable routing technique is deeply affected.

Traditional table-driven routing protocols [1] cannot perform efficiently in such an environment [2]. The reason is that they waste the limited system resources to discover routes that are not needed. On the other hand, on-demand routing protocols [1],[3]-[4] have been proposed as an effective solution to the problem. Their main advantage is that a route discovery is performed only when there is a request for communication between two network nodes. Thus, the bandwidth needed for the protocol operation is minimized.

One of the most representative on-demand protocols is the Dynamic Source Routing (DSR) protocol [3]. It is based on source routing techniques known from IEEE 802 LANs implementation. Routes are discovered only when needed using a route discovery procedure. This means that either a user request or a route break down may cause a new route discovery. In this way, routing packets are minimized since only the topology changes of interest are considered. Although DSR outperforms table driven protocols, for high host mobility, frequent link failures degrade the network performance and increase the routing load [2].

Another well-known on-demand protocol is AODV [4]. AODV discovers paths with a procedure similar to this of DSR but without using source routing. AODV maintains tables instead of caching routes. To avoid loop formation AODV makes use of sequence numbers that represent the freshness of routing information. In this way, AODV manages to deliver more successfully data packets and at the same time reduce significantly the routing load involved in a route discovery phase. On the other hand AODV fails to reduce the number of route discoveries because it does not make full use of routing information. As a result the overall overhead increases as demonstrated in [2] and [5].

Summarizing, it is clear that the advantage of DSR, allowing it to reduce routing load is the use of extensive routing information stored in each node cache. On the other hand AODV manages to avoid using stale routing information by means of sequence numbers and therefore increase delivery ratio. In this paper we will propose a new routing protocol that uses cached routes combined with sequence numbers to enhance network performance in terms of both delivery ratio and routing load.

The rest of the paper is structured as follows. In Section II the proposed protocol is presented in detail. In Section III we present the results of a simulation study in which the new protocol is compared to DSR. Finally, useful conclusions are drawn in Section IV.

II. Proposed Algorithm

Sequence number Aided Source Routing (SASR) is an on-demand protocol. Like most of the algorithms of this category consists of three mechanisms: i) *route request*, ii) *route set-up* and iii) *route maintenance*. Each node i maintains a request number (rn_i), a sequence number (sn_i), a table where it stores the last known request number for each node j ($rn_i[j]$), a table where it stores the last known sequence number for each destination ($sn_i[j]$) and a cache memory where it stores all known routes. SASR does not implement source routing in discovering a route to a specific destination. Instead it uses sequence numbers, as AODV does, to avoid loop formation and use only fresh routing information. However the implementation of sequence numbers must undergo certain changes because SASR uses cached routes rather than routing tables. The use of cached routes is necessitated by the need to exploit all routing information gathered during a route discovery. To this effect SASR utilizes source routing in the setting up of routes as well as in packet forwarding. In the following, the three basic mechanisms that constitute SASR will be described in detail.

A. Route Discovery

As mentioned before, SASR is an on-demand protocol. This means that a node initiates a route discovery only when it is in need of a route to a destination. Suppose that a node with address s wants to send a data packet to a destination node with address d. In the beginning, node s checks its cache for a route to destination. If such a route doesn't exist node s increases counter rn_s and then a route request packet is broadcasted. The route request packet has the structure $\langle s, pn, rn_s, sn_{max_d}, d\rangle$ where pn is the address of the node from which the packet was received (initially $pn = s$), rn_s the request number of the source and sn_{max_d} the maximum sequence number which is recorded by the packet during its travel (initially $sn_{max_d} = sn_s[d]$, with $sn_s[d]$ being the sequence number of node d known at node s). Each node i receiving the route request packet first checks the numbers $\langle s, rn_s\rangle$. If $rn_s < rn_i[s]$ then the packet is discarded, otherwise after updating $rn_i[s]$, node i performs a series of actions: i) if number $sn_i[d] > sn_{max_d}$ then the packet is updated with the value $sn_i[d]$, ii) adds in its cache a route entry that contains the data $\langle pn, s, rn_s\rangle$, which represents the reversed route from which the packet was received. This route entry is used in setting-up the discovered route, iii) the packet is finally broadcasted. After a certain number of retransmissions, the request packet will reach its destination or a node having a "valid route" to destination. In both cases if the receiving node hasn't previously heard the packet, increases its sequence number and proceeds to route set-up phase. The term "valid" will be explained in the following.

B. Route set-up

The route set-up procedure is initiated by the destination node or a node possessing a "valid" cached route to the destination. The route set-up procedure involves sending back to the source of the request, a reply packet containing the discovered route. Let us denote the address of the replying node by r, by $\langle x_{r,j,d}\rangle$ the vector which contains the addresses of the nodes consisting the $j-th$ route to d existing in the route cache of node s and by $\langle sn_{r,j,d}\rangle$ the vector of the corresponding sequence numbers. The reply packet contains both vectors which can be written in detail:

$$\langle x_{r,j,d}\rangle = \langle r, i_1, i_2, \ldots, i_{k-1}, d\rangle \tag{1}$$

$$\langle sn_{r,j,d}\rangle = \langle sn_r, sn_{r_j}[i_1], \ldots, sn_{r_j}[i_{k-1}], sn_{r_j}[d]\rangle \tag{2}$$

where $sn_{r_j}[i]$ the sequence number of node i at the time that route j was formed. It is clear that in the case that $r = d$ the aforementioned vectors are simplified to $\langle d\rangle$ and $\langle sn_d\rangle$. Node r sends to node pn a reply packet that contains the following information $\langle s, rn_s, \langle x_{r,j,d}\rangle, \langle sn_{r,j,d}\rangle\rangle$. When node pn receives the packet, adds the discovered route carried in the packet along with the sequence numbers, to its cache. Furthermore, node pn updates the numbers $sn_{pn}[i]\ ,\ \forall\, i \in \langle x_{r,j,d}\rangle$. Then, increases its sequence number sn_{pn} and adds it along with its address to the packet. Finally, using the numbers $\langle s, rn_s\rangle$, recalls from its case the next hop to node s and transmits the packet. The route entry corresponding to the path towards node s is deleted after a period of t_{exp} seconds rather than immediately. This is done to allow multiple replies to reach the originator of the request. Time t_{exp} may be relatively high without affecting the performance of the network in a negative manner.

C. Replying from Cache

Contrary to DSR not all of nodes having a route to a destination can reply to a request for this destination. As described, a route entry contains not only the addresses of the nodes that must be traversed to reach the destination but also their sequence numbers at the time the route was created. Suppose that a node r receives a packet originated from node s, requesting a route to node d. It searches its cache for a "valid" route to node d. A route to a destination is considered "valid" only if:

$$sn_r[d] > sn_{max_d} \tag{3}$$

where $sn_{max,d}$ is the sequence number recorder in the packet header. In this way not only more up-to-date routes are used but also the formation of loops is precluded. The latter can be proved by the following rationale. It is clear that:

$$sn_{i_k}[d] \geq sn_r[d]\ ,\ \forall\, i_k \in \langle x_{r,j,d}\rangle \tag{4}$$

since the route reply packet that contained in its header the combined vector $(\langle x_{r,j,d}\rangle, \langle sn_{r,j,d}\rangle)$ has already traverse all the nodes included in this vector. Let us suppose that Eq. 3 is valid and node r replies to the request packet using route $\langle x_{r,j,d}\rangle$. Let us also suppose that the request packet had already traversed one of the nodes $i_k \in \langle x_{r,j,d}\rangle$ (loop formation). Then it is clear that the inequality $sn_{max_d} \geq sn_r[d]$ should be valid, which conflicts with Eq. 3.

D. Packet Forwarding and Route Maintenance

SASR uses source routing in forwarding data packets. This means that the route to the destination is written on the packet header. Accordingly the identification of broken links is made in the same manner as in DSR. That is that if a transmitting host times out (after retransmitting the packet for a number of times), it then sends an error packet to the originator of the original packet. The error packet contains the addresses of the two nodes constituting the broken link. Upon receiving an error packet, a host removes from its cache the parts of the routes that contain the broken link. Furthermore, by using source routing, SASR is able to take advantage of all optimization procedures proposed for DSR [7] such as salvaging, gratuitous route repair and promiscuous listening.

III. Simulation Results

To evaluate the performance of SASR and DSR, we developed the code in C++ programming language. At the link layer we simulated the discrete coordination function (DCF) of the IEEE 802.11 Wireless LAN standard [6]. For DSR all the optimization functions described in [7], were implemented. We simulated a scenario in which 50 nodes move randomly in an area of $1000x1000m^2$. The nominal communication range of each host was considered equal to $250m$. The total time for each simulation was $900s$. A total of 40 connections each one emerging from a different host, was used. The packet generation rate of each host was set to 1 packet/sec and

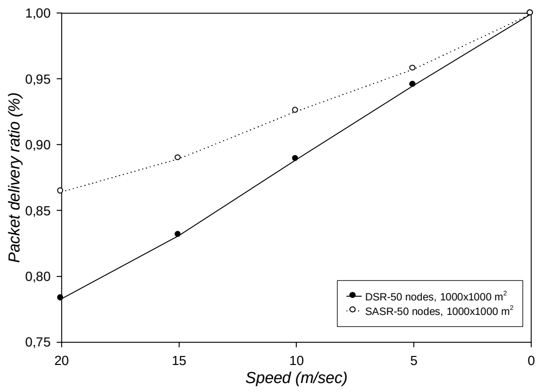

Fig. 1. Packet delivery ratio for various speeds

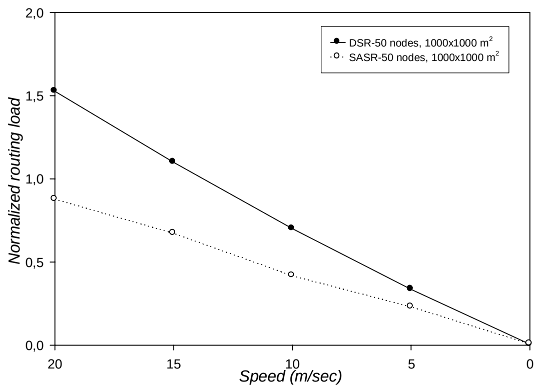

Fig. 2. Normalized routing load for various speeds

the packets where generated using the well-known Poisson distribution. The channel capacity was set to 2 Mbits/s, while the packet size is 512 bytes. The time that a packet is allowed to wait for a route in a buffer was set to $500ms$. The timeout for repeating a route request was set to $1s$ and for a non-propagating search to $30ms$. The size of packet's header is determined as follows: 4 bytes for each node address and 4 bytes for every other information carried, such as request or sequence numbers. The hosts move according to the Random Waypoint algorithm [2]. The speed of each node is uniformly chosen in the range of 0 to $v_{max} m/sec$ and different values of v_{max} (0,5,10,15 and 20 m/sec), were tested. In Fig. 1, the packet delivery ratio (i.e. the ratio of packets successfully delivered to packets generated) for different speeds is depicted. It is clear that SASR outperforms DSR over the whole mobility range. Moreover, the performance of SASR degrades slower than this of DSR, which indicates its durability in quickly varying environments. Fig. 2 depicts the normalized routing load (i.e. the number of routing packets per data packet successfully delivered) is illustrated. Again SASR performs better than DSR for all speeds. In fact, SASR scales better than DSR, which is a very useful property for a routing protocol. The performance of SASR is owned to the fact that it uses more fresh routing information. Finally, in Fig. 3, the average delay encountered by each data packet is illustrated. SASR

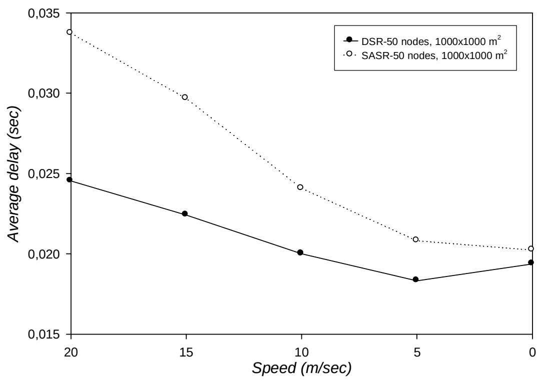

Fig. 3. Average delay for various speeds

presents higher delays as a result of two reasons. The first is the longer queues formed in the network as a result of the greater delivery ratio. The second one relates to the fact that in the case of DSR the calculation of average delay is biased in favor of packets that travel a small number of hops and therefore encounter smaller delays. This happens because for packets travelling more hops, there is a greater probability to be dropped.

IV. Conclusion

In this paper a novel routing algorithm, suitable for multihop ad-hoc networks has been presented and evaluated. The new algorithm combines source routing with the use of route caching and sequence numbers to lessen the routing load and at the same time achieve greater delivery ratio. The evaluation of the algorithm proved that SASR presents a performance superior to that of DSR in terms of both delivery ratio and routing load. Moreover the new algorithm does not use source routing in the route discovery phase, avoiding in this way the waste of useful network resources.

References

[1] Elizabeth M. Royer, Chai-Keong Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks", IEEE Personal Communications, Vol. 6, No. 2, pp. 46-55, April 1999.

[2] Josh Brosh, D.A. Maltz., D.B. Johnson, Y.C. Hu and J. Jetcheva, "A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols" in Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'98), October 25-30, 1998, Dallas, Texas, USA.

[3] D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," Mobile Computing, E. Imielinski and H. Korth, Eds. Norwell, MA: Kluwer, 1996.

[4] Charles E. Perkins, Elizabeth M. Royer, Samir R. Das, "Ad Hoc On-demand Distance Vector Routing", October 2001 IETF Draft, available at http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-08.txt.

[5] S.R.Das, C.E.Perkins, E.M.Royer, "Performance Comparison of two On-demand Routing Protocols for Ad Hoc Networks", in Proc. INFOCOM 2000, pp.3-12.

[6] IEEE Computer Society LAN MAN Standards Committee. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std 802.11-1997. The Institute of Electrical and Electronics Engineers, New York, New York, 1997.

[7] David Maltz, Josh Broch, Jorjeta Jetcheva, and David Johnson. "The effects of on-demand behavior in routing protocols for multihop wireless ad hoc networks". IEEE Journal on Selected Areas in Communication, Vol. 17, No.8, August 1999, pp. 1439-1453.