SDDSR: SEQUENCE DRIVEN DYNAMIC SOURCE ROUTING FOR AD HOC NETWORKS

RAMZI A. HARATY AND WAEL KDOUH
LEBANESE AMERICAN UNIVERSITY
DIVISION OF MATHEMATICS AND COMPUTER SCIENCE
BEIRUT, LEBANON
EMAIL: {rharaty, wael.kdouh}@lau.edu.lb

ABSTRACT

Mobile ad-hoc networks are becoming more popular as the use of mobile computers is increasing. The biggest challenge facing such networks is continuous and random change in the topology. Table driven routing protocols where not designed for such networks. For this reason new routing protocols that can handle continuous change of topology where created. Two popular protocols are Dynamic Source Routing (DSR), and Ad-hoc On-demand Distance Vector (AODV). DSR has the advantage of making heavy use of routing information to reduce the routing load, where as AODV has the advantage of using sequence numbers, which guarantees that at all time we are using non-stale routing entries. In this paper, we present the implementation of Sequence Driven Dynamic Source Routing (SDDSR). SDDSR is an on-demand routing protocol, which builds upon DSR and AODV. We use the NS-2 simulator to show the experimental results of the new protocol. The results showed better packet delivery as well as less routing load.

KEYWORDS: Ad-hoc On-demand Distance Vector, Dynamic Source Routing, Mobile Networks, and Sequence Driven Dynamic Source Routing.

1. INTRODUCTION

Wireless networks have been gaining popularity in recent years for many reasons. Perhaps the most important of which is the innovative approach concerning the design of the wireless processors. Nowadays, we can see them integrated within compact electronic equipments like Pocket PCs. Another reason is the availability of those chips at cheap prices. An example is Intel's CMOS system, which is currently being manufactured and supports 802.11a, b and g, as well as the next generation 802.11n wireless networks, and hence this is an indication that wireless networks will keep on gaining even more popularity in the near future. Another promising technology is Intel's WIMAX, a technology which will boost the popular VOIP. This popularity of wireless technology yields to that of wireless mobile ad-hoc networks.

A mobile ad-hoc network is a peer-to-peer wireless network where there is no centralized access point which regulates the flow of data between hosts; rather each host participating in the network will act as a router. In such networks, hosts usually move randomly from each other's range. Knowing this we can deduce that a lot of processing will have to be handled by each host. It is also obvious that there is no static graph of the overall network; rather this graph is being updated continuously. As a result, an appropriate routing protocol is needed due to the limited resources of such hosts as well as the random nature of such networks which may end up increasing routing load.

Traditional table-driven routing protocols such as RIP, DSDV, CGSR, and WRP lack the capability to cope with the random change of topologies in mobile ad-hoc networks [7]. The

aforementioned problem results from the fact that such protocols waste limited resources to discover routes that may never be needed. On the other hand, on-demand routing protocols have been introduced as solution of this problem [9]. Such protocols only attempt to initiate route discovery only when there is a request of communication between two hosts. Thus, less processing will be handled by each host since less route discoveries will be initiated.

There are many existing on-demand routing protocols such as AODV, DSR, TORA, ABR, SSR, and WRP [10]. The best existing on-demand routing protocols are DSR and AODV. DSR uses source routing and new routes are discovered only when needed. In other words, route discovery is only initiated when route request occurs, or route break down takes place. Although DSR surpasses table driven protocols, when high mobility occurs routing load tends to increase due to frequent link failures.

AODV, on the other hand, discovers routes in a similar procedure but without the use of source routing. AODV maintains tables instead of caching routes. A feature introduced in AODV is the use of sequence numbers to represent the freshness of routing information. Thus, AODV succeeds in delivering more packets and at the same time reduces the routing load. But AODV has its own problem, which is the number of route discovery request since it does not make full use of routing information. As a result the overall overhead increases.

Looking at the two protocols we can see that the advantage of DSR is the heavy use of routing information stored in each node's cache. Where as the advantage of AODV lies in the fact that it uses sequence numbers to avoid stale information and thus increase delivery ratio.

The new proposed protocol, Sequence Driven Dynamic Source Routing, uses the advantages of the two protocols. In other words it uses source routing combined with sequence numbers to enhance the overall performance.

The rest of the paper is organized as follows. Section 2 defines the specifications and goals of the new protocol. Section 3 addresses the design and implementation of SDDSR. Section 4 reports the experimental results. And section 5 presents the conclusion.

2. SPECIFICATIONS AND GOALS

Like all on-demand routing protocols, Sequence Driven Dynamic Source Routing consists of three phases: (a) route discovery, (b) route setup, (c) route maintenance. SDDSR aims at to improve packet delivery ratio, as well as reducing routing load. Packet delivery ratio can be improved by the use of sequence numbers, which guarantees at all times the use of fresh routes. As for the routing load, it is solved through two stages. The first stage is the heavy use of information stored in each node's cache, which results in less route requests. Where as the second stage ensures that the amount of information carried during route discovery is always minimized.

2.1 SDDSR Phases

Like any on-demand routing protocol SDDSR includes three phases:

2.1.1 Route Discovery

Route discovery allows any host in the ad-hoc network to dynamically discover a route to any other host in the ad hoc network, whether directly reachable within wireless transmission range or

reachable through one or more intermediate network hops through other hosts. A host initiating a route discovery broadcasts a route request packet, which may be received by those hosts within wireless transmission range of it. The route request packet identifies the host for which the route is requested. If the route discovery is successful, the initiating host receives a route reply packet listing a sequence of hops through which the target is reachable. The route request propagates through the network until it reaches the destination or an intermediate node possessing a route to the destination. This node is responsible for sending the route reply.

2.1.2 Route Setup

This phase is also known as route reply, it involves sending back a reply to the source of the request. This reply will eventually inform the source about the route to destination. The replying host could be the destination itself or any intermediate host depending on the validity of the entry.

2.1.3 Route Maintenance

This phase is responsible for identifying any broken links in the network that may be caused by a host going out of transmission range, or by a sudden halt of a host. Post to failure identification, a route request is initiated triggering the whole three phases to take place again.

2.2 SDDSR Performance Analysis

The new protocol is evaluated based on two key metrics:

(i) Packet Delivery Ratio - the ratio of the packets delivered to the destination to those generated by the CBR (Constant Bit Rate) sources.
(ii) Routing Load - the number of routing packets "transmitted" per data packet "delivered" at the destination.

Other important parameters to evaluate the protocol performance are shown in Table 1.

Performance parameters	**AODV**	**DSR**	**SDDSR**
Time complexity (initialization)	O(2d)	O(2d)	O(2d)
Time complexity (postfailure)	O(2d)	O(2d)	O(2d)
Communication complexity (postfailure)	O(2n)	O(2n)	O(2n)
Loop-free	Yes	Yes	Yes
Routes maintained in	Route table	Route cache	Route cache
Route reconfiguration methodology	Erase route; notify source	Erase route; notify source	Erase route; notify source
Routing metric	Freshest and shortest path	Shortest path	Freshest and shortest path

Table 1. Performance parameters of three on-demand routing protocols.

*d = Diameter of the network
*n = Number of nodes

Since SDDSR is based on both DSR and AODV, it is expected to show similar results as those shown in table 1.

3. SDDSR

Destination Sequenced Dynamic Source Routing is an on-demand routing protocol. Like most of the protocols of its category, it consists of three phases:

i) Route discovery,
ii) Route set-up, and
iii) Route maintenance.

3.1 The Algorithm

Every host **i** has a request number ($\mathbf{rn_i}$), a sequence number ($\mathbf{sn_i}$), and two tables for storing the last known request number for each host **j** ($\mathbf{rn_i[j]}$) and the last known sequence number for each destination ($\mathbf{sn_i[j]}$). The request number $\mathbf{rn_i}$ along with address will be used by the receiving nodes to identify the request of node **i**. As for the sequence number $\mathbf{sn_i}$, it will be used to make sure that only fresh routes to node **i**, are going to be used, and hence improve the delivery ratio.

SDDSR uses caching in stead of routing tables, since it is going to utilize source routing in the route setup. To overcome the problems of DSR [10, 12, 17, 25], where the addresses of all nodes had to be carried during both, the route discovery as well as the route reply, SDDSR does not use source routing in discovering a route to the destination. Instead it uses the same mechanism as AODV, where every node along the path will point to the previous node from which it received the request. This helps in reducing the routing load. So by the time the request will reach the destination, or any other node acquiring a valid route to the destination, it will know the way back to the source. After the route to the destination is discovered, a reply containing the addresses of all nodes (source routing) has to be built during the route reply. As the reply propagates back towards the source of the request, each node will add its address as well as its latest sequence number. The addresses will eventually be used to build the source route where as the sequence numbers will be used to update the tables of all nodes on the way back to make that all time they contain the latest sequence numbers of all the nodes along the active path. The use of sequence numbers also ensures that the network is loop free. However, the use of cached routes imposes some changes on the way sequence numbers are used. The use of cached routes is important to make use of all routing information gathered in the route discovery phase, which will lead us to solve the problem of AODV, since AODV uses sequence numbers without exploiting all routing information.

3.1.1 Route Discovery

Since SDDSR is an on-demand protocol, it implies that a host initiates a route discovery only when it needs a route to the destination. Suppose there is a host **s** that needs to send a data packet to a destination host **d**. First, host **s** checks whether it has a route to the destination, and in case it does not, it will increase its counter $\mathbf{rn_s}$, before initiating a route request which has the following structure: $\mathbf{<s, pn, rn_s, sn_{maxd}, d>}$ where **pn** denotes the address of the host from which the packet was received (originally **pn=s**), $\mathbf{rn_s}$ the request number of the source host, and $\mathbf{sn_{maxd}}$ the maximum sequence number recorded during the packet's travel (initially $\mathbf{sn_{maxd}= sn_s[d]}$, where $\mathbf{sn_s[d]}$ is the sequence number of host **d** known at host **s**). Each host **i** that receives the RREQ, checks the numbers $\mathbf{< s, rn_s >}$. If $\mathbf{rn_s < rn_i}$[s] then the RREQ is discarded, otherwise $\mathbf{rn_i}$[s] gets

updated, and host **i** performs a series of actions: i) if number $\mathbf{sn_i}[d] > \mathbf{sn_{maxd}}$, then the packet is updated with the value $\mathbf{sn_i}[d]$, ii) adds in its cache a route entry that contains the data $< \mathbf{pn, s, rn_s} >$, which will be used later as a reverse route to reach the host 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 period of time, the RREQ packet will reach either the destination or an intermediate node possessing a "valid" route to the destination. In either case, if the receiving host has not processed the RREQ before, it increases its sequence number and starts the route setup phase.

3.1.2 Route setup

After the RREQ packet reaches the destination or an intermediate node possessing a "valid" route to the destination, the route setup is started by that node. It involves replying to the requesting node by sending back the discovered route. Let us consider the address of the replying host is **r**, and $\mathbf{<x_{r,j,d}>}$ the vector containing the addresses of the hosts consisting the j-th route to **d** existing in the route cache of host **r,** and $\mathbf{<sn_{r,j,d}>}$ the vector of the corresponding sequence numbers. The reply packet includes the following vectors:

$$\mathbf{<x_{r,j,d}> = <r, i_1, i_2, \ldots, i_{k-1}, d>} \quad (1)$$

$$\mathbf{<sn_{r,j,} > = <sn_r, sn_{rj}\,[i_1], \ldots, sn_{rj}\,[i_k\text{-}_1], sn_{rj}\,[d]>} \quad (2)$$

$\mathbf{sn_{rj}\,[i]}$ is the sequence number of host **i** at the time **j** route was formed. It is clear that in the case that r=d the aforementioned vectors are simplified to [**d**] and $[\mathbf{sn_d}]$.

Host **pn** will receive a reply packet from **r** that contains the following information$\mathbf{<s, rn_s, x_{r,j,d,} sn_{r,j,d}>}$. After receiving the RREP packet, **pn** caches the new route and updates the numbers $\mathbf{sn_{pn}[i]}$, for any **i** belonging to $\mathbf{<x_{r,j,d}>}$. In addition it updates its sequence number $\mathbf{sn_{pn}}$, before adding it along with its address to the RREP packet. Finally, using the numbers $\mathbf{<s, rn_s>}$, it recalls the next hop to host **s** in order to transmit the packet**.** The reverse path pointing towards the source node **s** is deleted after a period of $\mathbf{t_{exp}}$ seconds. This is done to allow multiple replies to reach the originator of the request.

A favorable feature of SDDSR is the control of cache reply. Contrary to DSR not all of hosts having a route to a destination can reply to a request for this destination. If, for example, a host **r** receives a packet originated from host **s**, requesting a route to host **d**, even if host **r** has a route in its cache to the destination, it can not use it before making sure that this route is valid. A route is only considered valid when:

$$\mathbf{sn_r[d] > sn_{maxd}} \quad (3)$$

where $\mathbf{sn_{maxd}}$ is the sequence number recorder in the packet header. In this way not only more up to data routes are used but it also the formation of loops is excluded.

3.1.3 Route Maintenance and Packet Forwarding

SDDSR uses source routing in forwarding data packets. As for broken links it identifies them in the same way as DSR. That is, if a host tries to send a packet for several times and does not get any reply, it will send RERR to the source of the request. When the host receives the RERR, it will remove all the routes containing the hosts forming the broken link, and a new route discovery process will be initiated by the source, if the route is still in need.

4. EXPERIMENTAL RESULTS

4.1 The Simulator

SDDSR was evaluated under NS 2 (Network Simulator version 2). NS is built to run under UNIX, so we used CYGWIN, which emulates UNIX platform under windows in order to compile it [4]. NS is written in C++ and OTcl (Tcl script language with Object-oriented extensions) [6]. NS is an event-driven network simulator that simulates variety of IP networks. It implements network protocols such as TCP and UPD, traffic source behavior such as FTP, Telnet, Web and CBR, router queue management mechanism such as Drop Tail and RED, routing algorithms such as Dijkstra.

NS is an object-oriented Tcl (OTcl) script interpreter that has a simulation event scheduler and network component object libraries, and network setup (plumbing) module libraries (actually, plumbing modules are implemented as member functions of the base simulator object).

To setup and run a simulation network, an OTcl scripts must be written to initiate an event scheduler, setup the network topology using the network objects and the plumbing functions in the library, and tell the traffic sources when to start and stop transmitting packets through the event scheduler.

The term "plumbing" is used for a network setup, because setting up a network is plumbing possible data paths among network objects by setting the "neighbor" pointer of an object to the address of an appropriate object. The power of NS comes from this plumbing.

MobileNode is the basic node object with added functionalities like movement, ability to transmit and receive on a channel that allows it to be used to create mobile, wireless simulation environments. The class MobileNode is derived from the base class Node. The mobility features including node movement, periodic position updates, maintaining topology boundary, etc. are implemented in C++ while plumbing of network components within MobileNode itself (like classifiers, dmux, LL, Mac, Channel, etc.) have been implemented in Otcl.

4.2 Simulation Results

In order to asses the performance of SDDSR, we created the code under C++ as well as OTcl. The simulated scenario consisted of ten nodes moving randomly within a boundary of 500x500m^2. The simulation runs for 400 *seconds.* We used the Random Waypoint Mobility to control the node movement as well as transmission range [2, 22].

After running the simulation, the results showed that SDDSR outperforms DSR in terms of both, routing load as well as packet delivery ratio.

Figure 1 shows that the packet delivery ratio of SDDSR is much higher than that of the DSR protocol [Note: we did not compare SDDSR with AODV since studies have shown that DSR outperforms AODV [5]. The reason for the high packet delivery ratio for SDDSR is that it never uses stale routes. If it does not have a route for a destination it finds one through “Route Discovery” and it also uses “Route Maintenance” to ensure that only valid routes are stored at the nodes. The graph represents the result of 10 nodes moving at different speeds. As shown in the graph, the more mobility we tend to have in the network, the better SDDSR tend to perform. We

can see that at the speed of 20 m/sec SDDSR performs 87% delivery ratio where as DSR performs 78%.

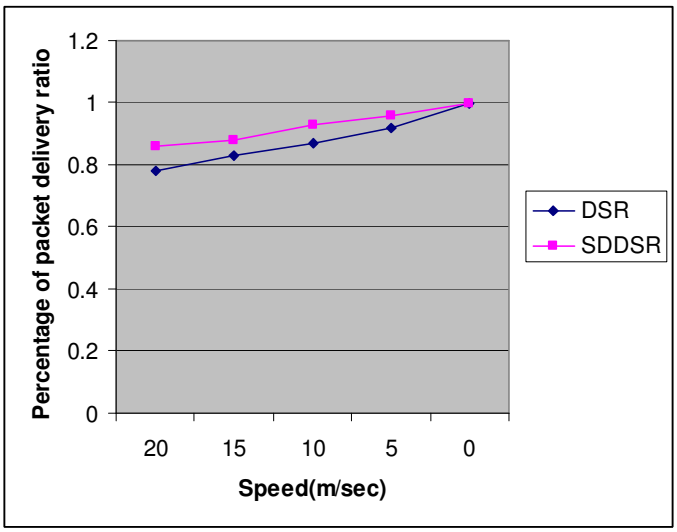

Figure 1. Delivery ratio for different speeds.

Figure 2 shows that the performance of SDDSR is again better. The better results under SDDSR are due to the use of fresh routing information. The routing overhead is the highest for low pause times. As the movement of the nodes becomes less frequent the routes persist for larger durations of time and hence the number of "Route Discovery" packets decreases. This explains the decrease in the routing overhead as the pause time increases.

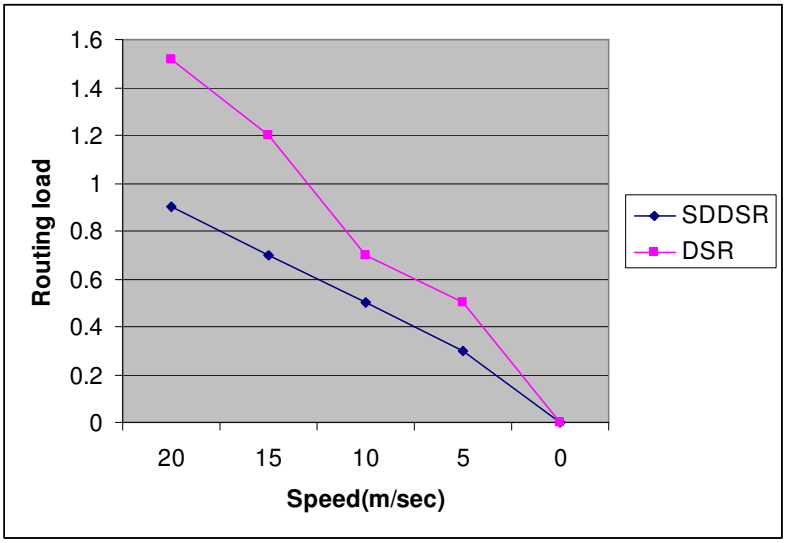

Figure 2. Routing load for different speeds.

5. CONCLUSION AND FURTHER WORK

In this work we developed a new routing protocol – SDDSR - for mobile ad-hoc networks. The new protocol uses source routing, caching, and sequence numbers, to

decrease routing load and improve delivery ratio. The results showed that SDDSR outperforms DSR in terms of both delivery ratio and routing load. Moreover, SDDSR does not use source routing during route discovery in order to avoid the waste of useful network resources, a problem which existed in DSR and caused the increase the routing load.

We plan to continue with our simulations for DSR and SDDSR. We would like to study the effects of changing other parameters (e.g., the number of mobile nodes, the dimensions of simulation space and the speed of movement of the nodes) and collect other metrics, if possible. We can also repeat each test with a large number of scenarios to remove the randomness, if any, from the results.

6. REFERENCES

[1] C. Bettstetter, "Topology Properties of Ad-Hoc Networks with Random Waypoint Mobility," ACM SIGMOBILE Mobile Computing and Communications Review, Vol. 7, Issue 3, pages 50-52, July 2003.

[2] J. Broch, D. Johnson, and D. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad-hoc Networks," http://www.ietf.org/internet-drafts/ draft-ietf-manet-dsrOl.txt, Dec 1998. IETF Internet Draft.

[3] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research," Wireless Communication & Mobile Computing (WCMC): Special Issue on Mobile Ad Hoc Networking: Research, Trends and Applications, Vol.2, No.5, pp.483-502, 2002.

[4] J. Chung and M. Claypool, "NS by Example". Available at http://www.wpi.edu. 2005.

[5] S. R. Das, R. Castaneda, J. Yan, and R. Sengupta, "Comparative Performance Evaluation of Routing Protocols for Mobile, Ad-Hoc Networks," Seventh International Comference on Computer Communications and Networks, pp. 153-161. October, 1998.

[6] K. Fall and K. Varadhan . "NS Notes and Documentation," 1999. Available from http://www-mash.cs.berkeley.edu/ns/.

[7] D. Johnson and I. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," On Computer Communications Review - Proceedings of SIGCOMM '96, Aug.1996.

[8] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad hoc Wireless Networks.," In T. Imielinski and H. Korth, editors, Mobile computing, chapter 5. Kluwer Academic, 1996.

[9] D. Maltz, J. Broch, J. Jetcheva, and D. Johnson, "The Effects of On-demand Behavior in Routing Protocols for Multihop Wireless Ad hoc Networks," IEEE Journal on Selected Areas in Communication, 1999.

[10] S. Murthy and 1. J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks," ACM Mobile Networks and App. J., Special Issue on Routing in Mobile Communication Networks, Oct. 1996, pp. 183-97.

[11] Y. Zhong and D. Yuan, "Dynamic Source Routing Protocol for Wireless Ad Hoc Networks in Special Scenario using Location Information," Communication Technology Proceedings, ICCT, International Conference, Vol.2, pages 1287-1290, April 2003.