Evaluation of AODV, DSR and DSDV Routing Protocols using NS2 Simulator

Shujun Bian Weidong Guo Yue Li Yuxiang Liu Chiyao Shen
{sbian3, wguo64, yueli, yliu858, cshen72}@gatech.edu

***Abstract* - A wireless adhoc network (WANET) is a particular type of computer network in which hosts are interconnected by routers using wireless communication links, which is quite important nowadays. So we try to receive the highest performance in wireless environments, for this we evaluate different routing protocols' performance and do the comparison between these protocols, especially the Adhoc On Demand Distance Vector (AODV) Routing, Dynamic Source Routing (DSR), and Destination Sequenced Distance Vector (DSDV) Routing by using NS2 simulator. The delay, throughput, control overhead and packet delivery ratio are the four most major part used to do the comparison.**

I. Introduction

A wireless ad hoc network (WANET) is a decentralized type of wireless network, which does not rely on a pre existing infrastructure, such as routers in wired networks or access points in managed (infrastructure) wireless networks. Due to the increasing trend of mobile devices usage in recent years, many applications emerged by taking use of the decentralized structure of wireless adhoc network. In our project, we plan to study the one of the wireless adhoc network application: mobile adhoc networks (MANET). We mainly focus on three MANET routing protocols: Ad hoc On Demand Distance Vector (AODV) Routing, Dynamic Source Routing (DSR), and Destination Sequenced Distance Vector (DSDV) Routing. Theses three protocols have some advantages and disadvantages under particular circumstances and we can analyze their performance by simulation. Specifically, we use NS2 to do the simulation and compare the difference between each protocol.

II. Motivation

In the process of the development of the Internet, PC Internet has become almost saturated and increase slowly while the mobile Internet is growing increasingly. With the more powerful mobile devices on the market, people start to do a lot of work on the mobile platform instead of PC platform. So we believe the future belongs to mobile devices and the mobile internet, which raises a new question that is current wireless network protocol strong enough to deal with the much more complex situation in mobile network today? Does current wireless network protocol still have good performance in today's mobile network? So these bunch of questions motivate us to do some deep look on the current wireless network protocol especially AODV, DSR, and DSDV.

We would repeat the experiment in *Performance analysis of AODV, DSR and DSDV Routing Protocols using NS2 Simulator*, while create some new scenario and use different test data to measure all 3 protocols. We would evaluate the performance of AODV, DSR, and DSDV, which help us to get a conclusion such that which protocol is better for today's mobile network, which protocol has a better performance in average PDR. Through this kind of conclusion, we might get some ideas about the design of the future mobile network protocol.

III. MANET Routing Protocols

A. Ad hoc On Demand Distance Vector

AODV is a reactive routing protocol, combining with some features of proactive routing. It's reactive because routes are found by an on demand approach that AODV only establishes and maintains a route when it's needed by a source. The advantage is that it's simple and traffic along the links remains low all the time. However, the disadvantage is that the time to establish a route on demand could be costly.
.

B. Dynamic Source Routing

DSR is also a reactive routing protocol, and is similar to AODV as it also uses on demand approach. The difference is DSR uses source routing the path from source to destination is stored in data packets. The advantage is similar to AODV. The disadvantage is that it's not scalable in the environment with high mobility.

C. Destination Sequenced Distance Vector

DSDV is also a proactive routing protocol. Each node has a routing table that contains the all destinations this node can reach, and "next hop" of each destination, ''number of hops'' to each destination and the sequence number. The route is based on the information in the route. The advantage

is similar to OLSR as some topology information is stored locally. The disadvantage is that it always needs resource to update the routing table, which also causes scalability problem.

IV. METHODOLOGY

1. Simulation Modeling

In communication and computer network research, network simulation is a technique where a program models the behavior of a network, which could help in analyzing the performance and behavior of complex networks before implementing it in today's real application. And through the simulation, we could set the different condition to run each protocols while evaluate their performance.

In this project, we decide to use the NS-2 Simulator as our experiment tool. NS2 is an open source event-driven simulator designed specifically for research in computer communication networks [2]. NS2 consists of two key languages: C++ and Object-oriented Tool Command Language (OTcl). While the C++ defines the internal mechanism (e.g. network protocol stack) of the simulation, the OTcl sets up simulation by assembling and configuring the objects as well as scheduling discrete events.

Hence, we have set bunch of parameters, for each run of each protocol, we would change some of the parameters in order to exam the performance of each protocols in different conditions. Here is the table of the parameters and their description.

Parameter	Description
Density(nodes/per unit area)	0.01,0.02,0.03,0.04,0.05
Packet Size(Bytes)	200,400,600,800,1000
Topology	Random
Channel	WirelessChannel
Model of radio propagation	TwoRayGround
MAC	IEEE 802.11
Antenna	OmniAntenna
Routing protocol	AODV/DSR/DSDV
-rxPower	0.3w
-txPower	0.6w
Connection Type	CBR/UDP

Table. 1 All parameters

2. Simulation Implementation and test cases

For each protocol, we implement it by inheriting a class called Agent which creates, transmits, receives, processes, and destroys routing packets. Basically, we create .cc and .h file for main definition of the protocol, packet header, route entry and routing table and the buffer to store data. Then, we will configure the network in OTcl scripts. Basically, we specify the number of nodes, how to deploy these nodes, protocols used in different layers, randomness, transmission and receiving power, antenna type, etc. In addition, we will turn on the 'trace' program to trace the packets in different layers. This 'trace' program will automatically generate data files in standard format and we will use 'gawk' to extract the useful information (e.g. packets drop rate, trace route RTT, throughput, etc.) from these files. Moreover, we will use 'Nam' (an animation tool) to view network simulation traces and real world packet traces.

We build the experiment environment by setting the following variables.

```
Agent/MessagePassing/Flooding instproc recv {source sport
size data} {
   $self instvar messages_seen node_
   global ns BROADCAST_ADDR

   # extract message ID from message
   set message_id [lindex [split $data ":"] 0]
   puts "\nNode [$node_ node-addr] got message
$message_id\n"

   if {[lsearch $messages_seen $message_id] == -1} {
    lappend messages_seen $message_id
     $ns trace-annotate "[$node_ node-addr] received
{$data} from $source"
     $ns trace-annotate "[$node_ node-addr] sending
message $message_id"
    $self sendto $size $data $BROADCAST_ADDR $sport
   } else {
     $ns trace-annotate "[$node_ node-addr] received
redundant message $message_id from $source"
   }
}

Agent/MessagePassing/Flooding instproc send_message {size
message_id data port} {
   $self instvar messages_seen node_
   global ns MESSAGE_PORT BROADCAST_ADDR

   lappend messages_seen $message_id
   $ns trace-annotate "[$node_ node-addr] sending message
$message_id"
   $self sendto $size "$message_id:$data" $BROADCAST_ADDR
$port
}
```

We set event that node 0 will send message to other nodes.

```
$ns at 0.2 "$a(0) send_message 800 1 {first
message}  $MESSAGE_PORT"
```

In evaluation part, we use AWK to retrieve information from the trace file that is generated after Tcl code is done. The sample AWK code to calculate the packet drop rate of the whole network is shown bellow.

```
# calculate the packet drop rate of the whole network
BEGIN {
```

```
sum=0;
for(i=0;i<50;i++)
    node[i]=0;
}
{
action = $1;
if((action=="r")||(action=="s"))
{
nodeid = $9;
if(node[nodeid]==0)
{node[nodeid]=1;
sum++;
}
if(action == "r")
    rcv[nodeid]++;
else if(action == "s")
    snd[nodeid]++;
}
}
END {
for(i=0;i<sum;i++)
{   printf("id %d, rcv %d, snd %d\n",i, rcv[i], snd[i]);
    rcvsum+=rcv[i];
    sndsum+=snd[i];
}
printf("avg rcv %.8f, avg snd %.8f\n", rcvsum/sum,
sndsum/sum);
printf("drop_rate:%.8f\n", 1-rcvsum/(sndsum*(sum-1)))

# calculate the end-to-end delay metric
BEGIN {
for(i=0;i<50;i++)
{
    energy[i][0]=-1.0;
    time[i][0]=-1.0
}
isHEAD = 1; #switch[1.for head node : 2.for normal node]
}
{
nodeid = $5;
if($1 == "N")
{
if(energy[nodeid][0]==-1.0)
{
    energy[nodeid][0]=$7;
    time[nodeid][0] = $3;
    sum++;
}
energy[nodeid][1]=$7;
time[nodeid][1]=$3;
}
}
END {
#node number, duration
for(i=0;i<sum;i++)
{
    rate[i]=(energy[i][1]-energy[i][0])/(time[i][1]-
time[i][0]);
    if(i!=0)
        avgrate += rate[nodeid]
}
if(isHEAD == 1)
    printf("%d %.8f\n",sum,rate[0]);
else
{

    printf("%d %.8f\n", sum, avgrate/(sum-1));
}
}
```

3. Screenshot of the test case scenarios

Here we proved or screenshot of test case scenarios, which are AODV, DSR and DSDV.

AODV:

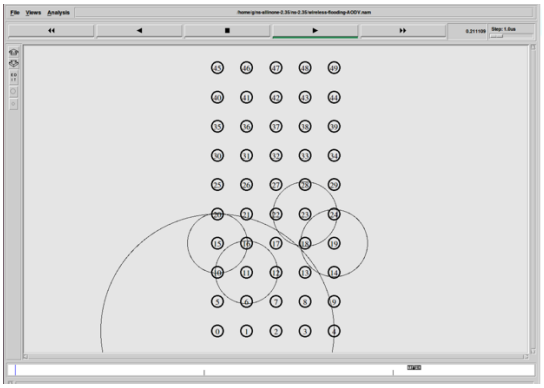

Fig. 1 AODV test case

DSR:

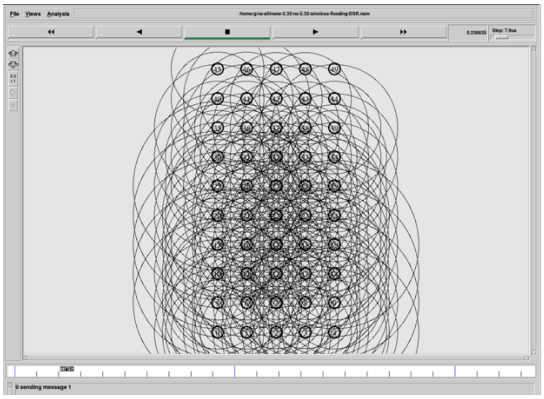

Fig. 2 DSR test case

DSDV:

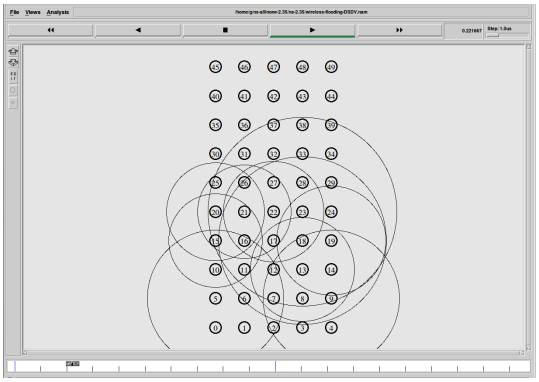

Fig. 3 DSDV test case

V. Performance Analysis

We run our test cases and we get the following results:

1. End to End Delay(seconds)

First evaluation is on End-to-End Delay. The results are shown bellow

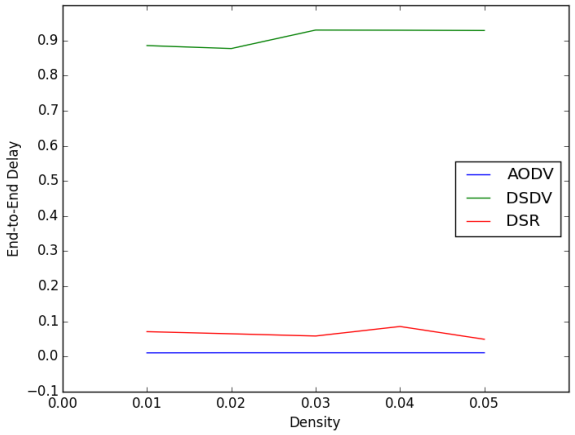

Fig. 4 Different Density End-to-End Delay(second)

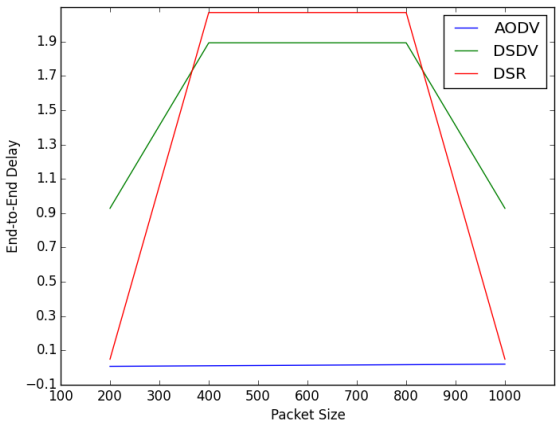

Fig. 5 Different Packet Size End-to-End Delay(second).

As we can see, AODV performance is the best considering its ability to maintain connection by periodic exchange of data's.

2. Throughput(Bytes/s)

Then we complete experiment regarding to the throughput, the results are shown bellow.

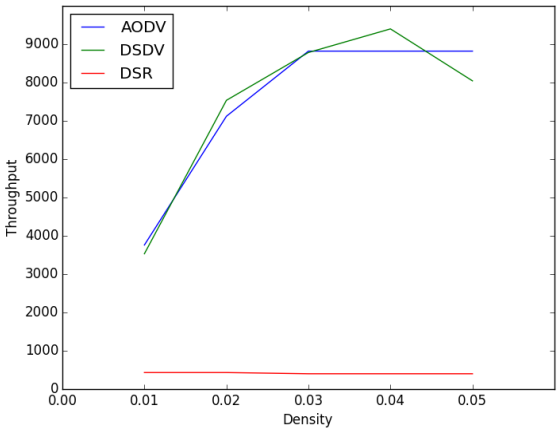

Fig. 6 Different Density Throughput(Bytes/s)

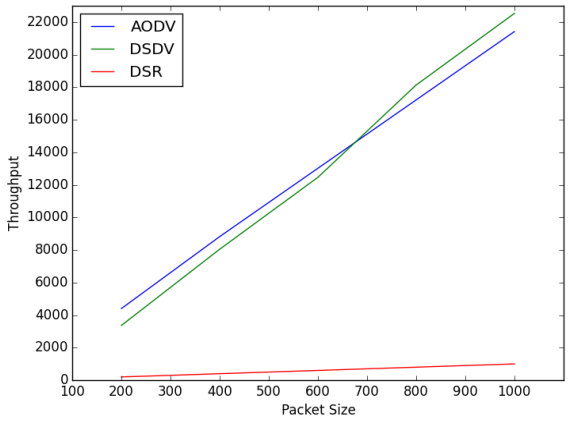

Fig. 7 Different Packet Size Throughput(Bytes/s)

DSDV has huge control overhead because its periodic routing table updates in the network. Then AODV is slightly lower than the DSDV and DSR have lower control overhead then two other routing protocols.

3. PDR

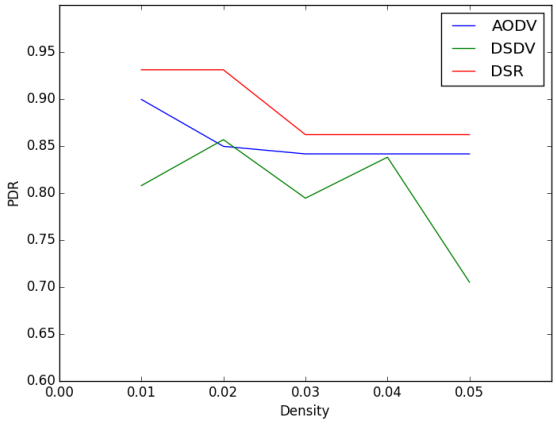

Fig. 8 Different Density PDR

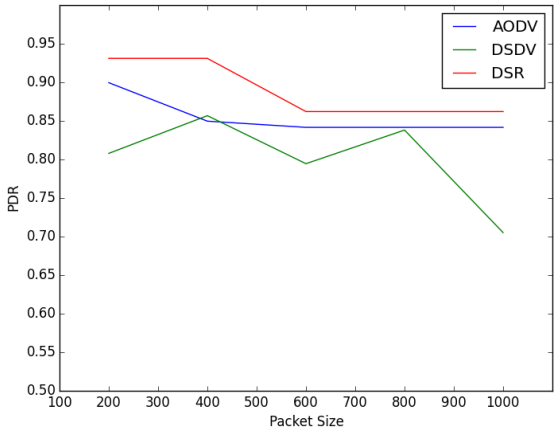

Fig. 9 Different Packet Size PDR

As it can be seen from the above results, Table driven routing protocol(DSDV) has lower PDR than reactive

protocols (AODV, DSR). Among these three protocols DSR has better PDR than AODV and DSDV.

In short, as far as throughput is concerned, AODV and DSR perform better than the DSDV even when the network has a large number of nodes. Overall, our simulation work shows that AODV performs better in a network with a larger number of nodes whereas DSR performs better when the number of nodes is slight.

Through our experiment so far, we could conclude some advantages and disadvantages of AODV, DSR, and DSDV protocols.

For AODV, the advantage is that routes are established on demand and destination sequence numbers are used to find the latest route to destination. The connection setup delay is less. While its disadvantage is that intermediate nodes can lead to inconsistent routes if the source sequence number is very old and the intermediate nodes have a higher but not the latest destination sequence number, thereby having stale entries. Also multiple Route Request packets in response to a single Route Re quest packet can lead to heavy control overhead.

For DSR, the advantage is that it uses a reactive approach which eliminates the need to periodic ally flood the network with table update messages which are in table-driven approach. The intermediate nodes also utilize the route cache information efficiently to reduce the control overhead. Meanwhile, its disadvantage is that its route maintenance mechanism does not locally repair a broken link. Stale route cache information could also result in inconsistencies during the route reconstruction phase.

For DSDV, the advantage is that it is simple and easy to implement and there is no latency caused by route discovery. However, no sleeping nodes and high overhead especially in the big scale network is its disadvantage.

VI. Future Work

In our current experiment, the nodes in different scenarios are stationary, and we will extend our experiment by removing that restriction in future work. We will consider the scenarios where nodes are subject to random movements, or constrained by various trace functions.

In addition, the second figure in the end-to-end delay section, which shows the delay with regards to different packet size, is hard to explain though. We plan to scale the parameters in future work, to find the potential cause of such consequence.

VII. Acknowledgment

First and foremost, we would like to express our sincere gratitude to Professor Ammar and all the TAs who have always been dedicated to this course. The course schedule is generally satisfactory, and we have acquired lots of knowledge both theoretical and practical during the last four months.

Furthermore, we want to thank all the members in our group and anyone who has offered us assistance. The efforts we have made, along with the spirits of collaboration we have witnessed during the project shall be cherished.

VIII. References

[1] Mohapatra, S., and P. Kanungo. "Performance analysis of AODV, DSR, OLSR and DSDV routing protocols using NS2 Simulator." Procedia Engineering 30 (2012): 69-76.

[2] Teerawat Issariyakul and Ekram Hossain. 2012. Introduction to Network Simulator Ns2 (2nd ed.). Springer Publishing Company, Incorporated.