

EEC 687/787 Mobile Computing

Ns-2 Laboratory #8

Prof. Chansu Yu

<http://academic.csuohio.edu/yuc/>
c.yu91@csuohio.edu

MANET Simulation (in-class lab)

- ❑ This lab aims at making the students familiar with routing protocols used in ad hoc networks and see the performance of DSDV protocol.
- ❑ DSDV is table-driven routing scheme for ad hoc mobile wireless networks based on the Bellman-Ford algorithm. Routing information is distributed between nodes via sending routing tables.
- ❑ Performance Metrics used
 - Packet delivery fraction : packets delivered / packets generated
 - Packet delay: sum of delays / number of packets delivered
 - Routing load : routing packets / packets delivered.

ns2 Instructions

- ❑ Get, read, and run the tcl script (dsv-test.tcl)
 - The topology - 3 nodes over 500x400 m² area (where are they?)
 - The mobility model – pre-defined (what is it?)
- ❑ Check the mobility using manet.nam
 - Pay attention the traffic changes around 55, 65 and 130 seconds
- ❑ Determine the performance using the trace file dsv-test.tr
 - PDR ?
 - Throughput vs time (fil_tcp.awk ?)
 - awk -f fil_tcp.awk dsv-test.tr > dsv.xg
 - xgraph dsv.xg
 - Average packet delay vs time => dsv-d.xg (?)

3

c.yu91@csuohio.edu

ns2 Instructions

- ❑ Edit dsv-test.tcl to use AODV instead of DSDV and run the simulation
- ❑ Check the mobility using manet.nam
 - When does the traffic change?
- ❑ Determine the performance using the trace file dsv-test.tr
 - Throughput vs time
 - awk -f fil_tcp.awk dsv-test.tr > aodv.xg
 - xgraph dsv.xg aodv.xg
 - Average packet delay vs time => aodv-d.xg
 - xgraph dsv-d.xg aodv-d.xg
- ❑ What're different?

4

c.yu91@csuohio.edu

ns2 Instructions

- How to check routing control packets?
 - `cat dsdv-test.tr | grep RTR | grep ^s | grep AODV | more`
 - RREQ, RREP, RERR packets (REQUEST, REPLY, ERROR)
 - `cat dsdv-test.tr | grep RTR | grep ^s | grep AODV | grep REQUEST | wc -l`
 - `cat dsdv-test.tr | grep RTR | grep ^s | grep AODV | grep REPLY | wc -l`
 - `cat dsdv-test.tr | grep RTR | grep ^s | grep AODV | grep ERROR | wc -l`
- How to check routing control overhead in DSR?
- How to check routing control overhead in DSDV?

5

c.yu91@csuohio.edu

How To Obtain #RREQ/RREP

- DSR Trace %d [%d %d] [%d %d %d %d->%d] [%d %d %d %d->%d]
- Number Of Nodes Traversed
- Routing Request Flag, Route Request Sequence Number
- Routing Reply Flag, Route Request Sequence Number, Reply Length
- Source Of Source Routing -> Destination Of Source Routing
- Error Report Flag (?), Number Of Errors, Report To Whom
- Link Error From -> Link Error To

6

http://nslam.isi.edu/nslam/index.php/NS-2_Trace_Formats

c.yu91@csuohio.edu

DSR traces using DSR code

❑ Trace file analysis

➤ Using grep and awk to get #RREQ

```
$ cat file.tr | grep DSR | awk '$19=="[1" {print $2}' | wc -l
```

➤ Using grep and awk to get #RREP:

```
$ cat file.tr | grep DSR | awk '$21=="[1" {print $2}' | wc -l
```

➤ Using grep and awk to get #RERR:

```
$ cat file.tr | grep DSR | awk '$25=="[1" {print $2}' | wc -l
```

** RERR can be piggybacked on RREQ, which is called gratuitous RERR and can be enabled by "dsragent_propagate_last_error"*

c.yu91@csuohio.edu

Reactive MANET Routing Protocols – DSR and AODV (Lab report)

❑ Compare the performance of DSR and AODV protocols with CBR traffic.

❑ Performance Metrics used

- Packet delivery fraction : packets delivered / packets generated
- Routing load : routing packets / packets delivered
- Average packet delay

NS2 Instructions

- Get the tcl script compare.tcl
- This script takes 4 command line arguments - scenario file, traffic file, output trace file and routing protocol (1 = DSR and 2 = AODV). Script usage is :
 \$ ns compare.tcl -scen {scen} -tfc {tfc} -tr {tr} -rpr {rpr}

- Use setdest to create five mobility files
 \$ setdest -v 1 -n 25 -p 0 -M 20 -t 100 -x 500 -y 500 > scen-25-0
 \$ setdest -v 1 -n 25 -p 10 -M 20 -t 100 -x 500 -y 500 > scen-25-10
 \$ setdest -v 1 -n 25 -p 20 -M 20 -t 100 -x 500 -y 500 > scen-25-20
 \$ setdest -v 1 -n 25 -p 40 -M 20 -t 100 -x 500 -y 500 > scen-25-40
 \$ setdest -v 1 -n 25 -p 100 -M 20 -t 100 -x 500 -y 500 > scen-25-100

NS2 Instructions

- Use cbrgen.tcl to create four traffic files

```
ns cbrgen.tcl -type cbr -nn 25 -seed 1 -mc 5 -rate 8.0 > cbr-25-5
ns cbrgen.tcl -type cbr -nn 25 -seed 1 -mc 10 -rate 8.0 > cbr-25-10
ns cbrgen.tcl -type cbr -nn 25 -seed 1 -mc 15 -rate 8.0 > cbr-25-15
ns cbrgen.tcl -type cbr -nn 25 -seed 1 -mc 20 -rate 8.0 > cbr-25-20
```

NS2 Instructions

- ❑ With five mobility and four traffic files, we have 20 simulation scenarios.
- ❑ For each of DSR and AODV, we will use a script file (run-dsr2.sh and run-aodv2.sh) to run them all. You can run these scripts as follows.
 - \$ chmod +x run-aodv2.sh
 - \$ chmod +x run-dsr2.sh
 - \$./run-aodv2.sh
 - \$./run-dsr2.sh
- ❑ The above scripts will create 3 files each.
 - aodv_sent, aodv_recv, and aodv_route_pkts
 - dsr_sent, dsr_recv, and dsr_route_pkts
 - Each line in the file is "Pause Time", "CBR Load", "Count".

11

c.yu91@csuohio.edu

NS2 Instructions

- ❑ Handin
 - Plot each performance metric for both DSR and AODV versus pause-times, for each CBR Load. (3-D graph?)
 - State if any peculiar behavior is observed. Give a brief report as to the interpretation of the graphs.
- ❑ What else can you do?
 - More simulations with different seeds?
 - Different scenarios (e.g., higher traffic)?
 - Compare DSDV too?

12

c.yu91@csuohio.edu