DKII: Simulation and Analysis of MANET Routing Protocols

1 The Purpose

The purpose of this assignment is to gain familiarity with a network simulation tool and to get experience in basic data analysis. The simulator that will be used is called ns-2 [1] and is common within the networking research community. You will run the simulation using a simple network scenario and investigate the behaviors of two different mobile ad hoc network (MANET) routing protocols. For detailed information on ns-2, please refer to the ns-2 manual.

2 Simulator Overview

Ns-2 is a discrete event simulator, which means that events (e.g., packets to send, timeouts, etc.) are scheduled in a global event queue according to their time of execution. When a simulation is run the simulator removes events from the head of the queue, moves the simulator time to that of the currently removed event and executes it. When done, it continues to the next event and so forth.

Each simulation is defined by a scenario, that contains a number of predefined events that define the scenario. Ns-2 scenarios are implemented in TCL-scripts that contain the commands to initialize the simulator and to create the nodes[1] and their configuration. Each simulation run generates a trace file containing all the data packets that are sent between the nodes during the course of the scenario. By analyzing this file it is possible to determine the performance effect of parameter variations, different routing protocols and more. A simulation can be very useful because it is possible to scale the networks easily and therefore to eliminate the need for time consuming and costly real world experiments. While the simulator is a powerful tool, it is important to remember that the ability to do predictions about the performance in the real world is dependent on the accuracy of the models in the simulator.

[1] Each simulated computer or device is called a node.

2.1 Simulation Scenario

The assignment focuses on a simple ad hoc network scenario consisting of four nodes that communicate using wireless network interfaces (WiFi). The scenario is illustrated in Figure 1 and is called *Roaming node*. The distinctivess of this type of simulation compared to simulations of static, wired networks is that the communication is wireless, the nodes are mobile and the connectivity therefore intermittent.

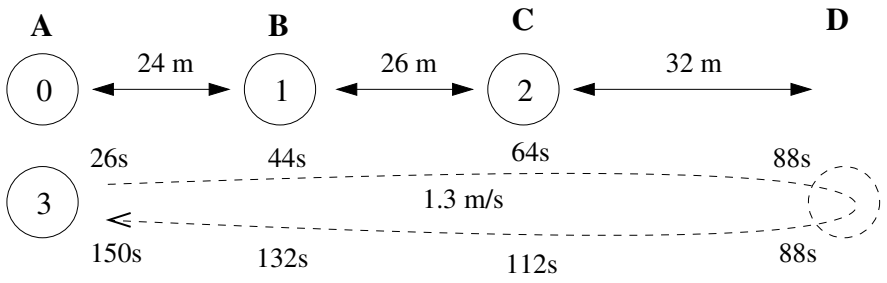

Figure 1: Logical overview of the *Roaming node* scenario in the ns-2 simulator. The dotted line indicates movement and is accompanied by the times of when node 3 passes each of the waypoints labeled A, B, C and D.

The Roaming node scenario comprises four nodes out of which three are stationary (nodes 0, 1, 2) and one node is mobile (node 3). Node 3 moves at normal walking speed (1.3 m/s) and during the course of the scenario it communicates with node 0. The movement will force node 3 to sometimes use nodes 1 and 2 to relay its data packets. It is the task of the routing protocol to automatically configure the required routes. In the scenario there is always a potential path between the source and destination nodes over either one, two or three hops. Therefore, packet loss should be low. You will run the AODV [3] and OLSR [2] routing protocols and evaluate their performance and explain their distinctive behavior in this particular scenario.

The data traffic consists of a Constant Bit Rate (CBR) UDP flow from node 3 to node 0, meaning that packets are sent from the source with a constant time inbetween them for the duration of the scenario. Node 3 will send 20 packets per second during 124 seconds (2480 packets total), starting 26 seconds from the start of the scenario. The ns-2 scenario file that implements this scenario is called `roaming-node-DK2.tcl` and is found in the software package accompanying the assignment.

2.2 Running the Simulation

Ns-2 is run from a UNIX terminal window. To edit the scripts used in this assignment you need a text editor, e.g., Emacs, Vi or similar.

The roaming node scenario can be run using AODV as the default routing protocol by passing the scenario file as input to ns-2. The optional *-prot* argument selects either AODV or OLSR routing (AODV is default).

```
$ ns roaming-node-DK2.tcl -prot [aodv|olsr]
```

Running this command in a terminal window will produce two output files. The first is the `.tr` file, which is the main *trace* file and the other is a `.nam` file, which is used to visualize the scenario using the *nam* animation tool. If you want to view an animation of the node movements you can run nam, e.g.:

```
$ nam roaming-node-udp-aodv.nam
```

However, the `.tr` trace file is the important file for this assignment and is used for the analysis.

2.3 The Trace file

Figure 2 shows an example excerpt from a trace file.

```
s 66.050000000 _3_ AGT  --- 1061 cbr 1370 [0 0 0 0] ------- [3:0 0:0 32 0] [801] 0 0
r 66.050000000 _3_ RTR  --- 1061 cbr 1370 [0 0 0 0] ------- [3:0 0:0 32 0] [801] 0 0
s 66.050000000 _3_ RTR  --- 1061 cbr 1370 [0 0 0 0] ------- [3:0 0:0 32 1] [801] 0 0
r 66.051688825 _1_ RTR  --- 1061 cbr 1370 [13a 1 3 800] ------- [3:0 0:0 32 1] [801] 1 0
f 66.051688825 _1_ RTR  --- 1061 cbr 1370 [13a 1 3 800] ------- [3:0 0:0 32 0] [801] 1 0
r 66.053481632 _0_ RTR  --- 1061 cbr 1370 [13a 0 1 800] ------- [3:0 0:0 32 0] [801] 2 0
r 66.053481632 _0_ AGT  --- 1061 cbr 1350 [13a 0 1 800] ------- [3:0 0:0 32 0] [801] 2 0
```

Figure 2: Example output from an ns-2 trace file.

Each line of the trace file represents one event traced at some level of the simulator, loosely corresponding to the layers of the TCP/IP network stack. Note that several lines can refer to the same packet, only at different levels of the simulator or at different nodes. The simulator does not use regular MAC or IP addresses, but instead uses simple node numbers for all types of addresses.

The first column of the trace file describes the type of event that the line represents. It can be either send (`s`), receive (`r`), forward (`f`) or drop (`d`). The second column is the time of the event, the third the node number and the fourth the type of the simulator entity/level responsible for the tracing (RTR=router, AGT=agent). Next follows an optional reason for the event, an event/packet ID, packet type and packet size, respectively. The first set of square brackets contains MAC level information: Time to send data, destination MAC address, source MAC address, and protocol type (800=IP). The second set of square brackets contains IP level data: Source IP node number and source port (separated by a colon), destination IP node number and destination port followed by the TTL

value and next hop address (if any). The data that follows is protocol specific and varies with type.

✍ **TIP!**: There are a number of good resources on the web with more details on the trace formats.

3 Analysis

The main task of this assignment is to analyze the trace files that are generated in the simulation. You are provided with a template Perl script (`cbr-analysis.pl`) to help you get started. You can run the script by piping the trace file into the script on "stdin".

```
$ ./cbr-analysis.pl < roaming-node-udp-[aodv|olsr].tr
```

Without any modifications the script prints one line for each generated CBR packet with some extracted information. Use this script template as a starting point for your analysis.

3.1 Task #1: Packet Delivery

At a glance:

1. Modify analysis script to print total number of packets received at the destination.
2. Answer the questions in the questionnaire.

The first task in the analysis is to calculate the number of packets that are received for AODV and OLSR. You need to modify the analysis script to check for logged CBR packets at the receiving node (AGT level) and that the source and destination IP match the source and destination nodes of the CBR flow.

3.2 Task #2: Visualization

At a glance:

1. Modify analysis script to print the number of packets received over time.
2. Use gnuplot to visualize the routing protocol performance.
3. Study the graph and answer the questions in the questionnaire.

The second task is to visualize the performance of each protocol to better understand how the protocols differ. For example, you want to know whether packet loss is evenly spread over time or concentrated to specific periods of time. A good approach is to plot the number of packets receieved over time. Modify the analysis script to instead print the total number of packets over time, i.e., each time a new packet is received. For the plotting program (Gnuplot) to understand the data you generate it should be in the format `[time #packets]`, e.g.:

```
26.251883275 6
26.301803275 7
26.351883275 8
26.401743275 9
```

With the updated script you can use Gnuplot to generate a graph by writing the data to a file called "sim.dat":

```
$ ./cbr-analysis.pl < roaming-node-udp-[aodv|olsr].tr > sim.dat
```

Then run gnuplot using the gnuplot command script bundled with the assignment:

```
$ gnuplot simplot.cmd
```

This will generate a postscript file called "sim-plot.eps" in the current directory. View the graph using, for example, *Ghostview*:

```
$ gv sim-plot.eps
```

Generate plots for both routing protocols and compare the results. Study the differences between the protocols. The differences may be small, but still visible. Explain the results based on your knowledge of the routing protocols and the scenario. Fill in the answers to the questions in the questionnaire.

3.3 Task #3: Latency

At a glance:

1. Modify analysis script to print the latency for each packet.
2. Use gnuplot to visualize the latency over time.
3. Study the resulting graphs and answer the questions.

Since the simulation uses a global clock for each node it is possible to accurately calculate the packet latency between two nodes. The latency is the time between a packet is sent from a source node and received by the destination node.

Modify your script to calculate the latency for each packet. Print the latency in a third column, next to the received packet number. The output should look similarly to Figure 3 below.

```
26.452 10 0.00165
26.502 11 0.00179
26.551 12 0.00147
26.601 13 0.00137
```

Figure 3: Analysis script output with number of packets received and packet latency.

To calculate the latency you need to keep track of sent packets so that you can match the send time against the receive time. Use the packet ID number to accomplish this. The send time of packets need to be saved so that the latency can be computed once the packet is received by the destination. Use an array or hash for this purpose.

Once you have successfully modified the script you should generate a new graph with the latency of each packet. You have to modifiy the `gnuplot` command (`simplot.cmd`) file to use column three instead of column two for the input data. Also do not forget to update the axis-labels if necessary.

Study the latency graph for each protocol. How do they differ and why? Why are they scaled differently? Fill in the questionnaire based on your analysis.

4 Hand-in Instructions

To pass the assignment follow the hand-in instructions below. *You only need to do one hand-in for each group.*

1. Complete the tasks required for your course and fill in the answers to the questionnaire at the end of this document.

2. You should provide a tarball with 1.) the produced graphs from your analysis and 2.) the modified analysis script. Create the tarball in the same form as the one provided for the assignment. Name the tarball "sim-{username}.tar.gz". Example:

   ```
   > tar zcvf sim-krmo5621.tar.gz sim-lab/
   ```

3. Put the tarball somewhere in your home directory on the institution's SUN system.

4. Fill in the form at the back of this compendium. Do not forget to provide a path to the tarball in your home directory. *Also make sure that the tarball has the right permissions, i.e., it must be readable by others.* Also make sure it is clear which people are part of the group.

5. Hand in the form and the questionnaire with your answers at the location instructed by your examiner.

If you do not follow these instructions your assignment will be automatically handed back to you uncorrected.

References

[1] The network simulator - ns-2. `http://www.isi.edu/nsnam/ns/`.

[2] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, and a. Qayyum et L. Viennot. Optimized link state routing protocol. In *IEEE National Multi-Topic Conference (INMIC 2001), December 2001.*

[3] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (AODV) routing, July 2003. IETF Internet RFC 3561.

DKII: Simulation and Analysis of MANET Routing Protocols

Name(s): _____

Login name: _____

Email: _____

Educational program: _____

Path to tarball: _____

(e.g., /home/gujo8932/sim-gujo8932.tar.gz)

Date: _____

✍Signature: _____

Questionnaire

Answers to Task #1

Received packets AODV:

Received packets OLSR:

Answers to Task #2

Explain, using your knowledge of the different routing protocols, the differences in the graphs that you have produced:

Answers to Task #3

Explain the appearances of the latency graphs for both protocols. Are there any differences between the protocols? In that case, explain the cause of the differences.