NS-2 Tutorial (Part 3)
Mobile Ad Hoc Networking in NS-2

Patrick Lee

October 9, 2001

Objectives

- Discuss the simulation of a 3-node ad hoc networking scenario
- Study the trace file

Please refer to wireless1.tcl in Marc Greis' Tutorial Chapter 9 [1].

What is Mobile Ad Hoc Networking?

- A mobile ad hoc network is a collection of mobile nodes whose communications do not rely on any infrastructure.
- Each node acts as a host and a router.

Start Simulation

- Create a simulator instance

```
set ns_ [new Simulator]
```

- Create trace objects in a 500x1000 area

```
set tracefd [open wireless1-out.tr w]
set nametrace [open wireless1-out.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace 500 1000
```

- Create an topology object

```
set topo [new Topology]
```

- Define topology in a 500x1000 area

```
$topo load_flatgrid 500 1000
```

God Object

- The God (General Operations Director) is a centralized object that stores the following information:
  - Minimum number of hops between nodes
  - Next hop

  Normally, the information is stored in each node and changed from time to time. The use of God is to save the burden in computing these information during simulation runs.

- Create the God object:

  ```
  create-god 3
  ```

Node Configuration

Use the command **ns_ node** to create mobile nodes and configure nodes as follows:

```
$ns_ node-config  -adhocRouting   DSR, TORA, AODV or DSDV  \
                  -llType         LL                        \
                  -macType        Mac/802_11                \
                  -ifqType        Queue/DropTail/PriQueue   \
                  -ifqLen         50                        \
                  -antType        Antenna/OmniAntenna       \
                  -propType       Propagation/TwoRayGround  \
                  -phyType        Phy/WirelessPhy           \
                  -channelType    Channel/WirelessChannel   \
                  -topoInstance   $topo                     \
                  -agentTrace     ON or OFF                 \
                  -routerTrace    ON or OFF                 \
                  -macTrace       ON or OFF                 \
                  -movementTrace  ON or OFF
```

Movement Pattern

- The scenario file specifies the locations of mobile nodes.
- To initialize the x, y, z coordinates of a node:

```
$node_(0) set Z_ 0.000000000000
$node_(0) set Y_ 239.438009831261
$node_(0) set X_ 83.364418416244
```

- God information is stored inside the scenario file:

```
$god_ set-dist 0 2 1
```

It means the shortest distance between node 0 and 2 is 1 hop.

- A node-movement generator `setdest` is available to generate the movement pattern. It is stored under *~ns/indep-utils/cmu-scen-gen/setdest* directory. To run the generator, go to the above directory and run `setdest` with the following arguments:

  `./setdest [-n num_of_nodes] [-p pausetime] [-s maxspeed] [-t simtime] [-x maxX] [-y maxY] > [outdir/movement-file]`

- Example:

  `./setdest -n 3 -p 3.0 -s 10.0 -t 200 -x 670 -y 670 > scen-3-testa`

Connection Pattern

- The connection file specifies the traffic flow and how the nodes are connected.
- A generator cbrgen.tcl is available under $\sim$*ns/indep-utils/cmu-scen-gen* directory. It can generate either TCP or CBR connections.

- To generate the traffic pattern, go to the above directory and run the command with the following arguments:

  `ns cbrgen.tcl [-type cbr|tcp] [-nn nodes] [-seed seed] [-mc connections] [-rate rate]`

- Example:

  `ns cbrgen.tcl -type cbr -nn 10 -seed 1.0 -mc 8 -rate 4.0 > cbr-10-test`

Trace file

- The trace file (*.tr) can be used to verify the correctness of an algorithm in the wireless scenario.
- The trace is handled by $\sim$*ns/cmutrace.cc* and $\sim$*ns/trace.cc*.
- Trace format:

  ```
  [Trace type] [time] [node id] [layer] [reason] [pkt id] [pkt
  type] [pkt size] [duration mac_id_dest_node mac_id_send_node
  ethertype_ip] ------ [send_ip:port dest_ip:port TTL hop_cnt]
  [seq_no ack_no] num_fwd opt_num_fwd
  ```

- Note:
  - Trace type: s(send) r(receive) f(forward) d(drop)
  - layer: AGT(agent), RTR(routing), MAC
- Example of the trace of a pakcet:

```
s 150.392740763 _1_ MAC --- 130 tcp 1080 [a2 2 1 800] ------- [1:0
2:1 32 2] [64 0] 0 2
r 150.397061339 _2_ MAC --- 130 tcp 1028 [a2 2 1 800] ------- [1:0
2:1 32 2] [64 0] 1 2
```

Log Movement

- The code node-config -movementTrace ON is supposed to log movement in the trace file, but it fails.
- Add the log-movement procedure to wireless1.tcl and call it:

```
proc log-movement {} {
    global logtimer ns_ ns
    set ns $ns_
    source timer.tcl
    Class LogTimer -superclass Timer
    LogTimer instproc timeout {} {
        global val node_;
        for {set i 0} {$i < $val(nn)} {incr i} {
            $node_($i) log-movement
        }
        $self sched 0.1
    }
    set logtimer [new LogTimer]
    $logtimer sched 0.1     # run every 0.1 second
}
```

- Note:
  - You should copy timer.tcl from $\sim$*ns/tcl/mobility* to your working directory, or simply specify the path in the procedure.
  - The procedure can be called after the ns_ is instantiated.

Node Mobility in Nam

- To view the results in Nam, you have to:
  - Download the latest daily snapshot of Nam (the Nam in all-in-one version cannot show node movement)
  - In ∼*ns/mobilenode.cc*, change -v to -V and compile (run make).

- You may turn the routerTrace, agentTrace and macTrace ON to see the traffic flow.

  Note: If you turn the macTrace ON, you will see many ripples originating from each node. It is because each packet transmission is preceded by an RTS/CTS exchange, which is done in the MAC layer.

References

1. Marc Greis' Tutorial Chapter 9 and 11
   (http://www.isi.edu/nsnam/ns/tutorial/index.html)

2. The ns Manual – Chapter 15 Mobile Networking in Ns
   (http://www.isi.edu/nsnam/ns/doc-stable/node139.html)