Wireless LANs
June – September 2009

NS-2:
Wireless Network Simulation

รศ. ดร. อนันต์ ผลเพิ่ม

Assoc. Prof. Anan Phonphoem, Ph.D.

anan.p@ku.ac.th

http://www.cpe.ku.ac.th/~anan

Computer Engineering Department

Kasetsart University, Bangkok, Thailand

Slide materials

Modified from :

- Lubo Song, Standard Ad Hoc Network Scenario In Ns-2 Simulation

Outline

- Basic Ad Hoc Network
- Ad Hoc Network with
  - Connection pattern file generator: cbrgen.tcl
  - Mobility file generator: setdest & calcdest

Ad Hoc Network

- Collection of wireless mobile nodes
- Without infrastructures
- Dynamic network topology
- Wireless channel: CSMA/CA
- Distributed algorithms

Simulation Scenario

- Network topology
- Traffic pattern
- Node configuration
- Trace file

Network Topology

- Number of nodes
- Moving range
- Initial positions
- Moving pattern
  - Direction
  - Velocity
  - Acceleration

Traffic Pattern

- Number of connections
- Traffic source/destination
- Connection type
  - TCP/UDP
- Packet size
- Packet rate

Protocols

- Transport layer
  - TCP connection: generate TCP traffic
  - UDP connection: generate CBR traffic
- Network layer
  - DSR: Dynamic Source Routing.
  - AODV: Ad Hoc On-Demand Distance Vector.
  - DSDV: Destination-Sequenced Distance Vector.
  - TORA: Temporally Ordered Routing Algorithm.
- Link layer
  - Queue/Delay control.
  - Fragmentation/Assembly control.
  - ARP: Address Resolution Protocol.

Protocols

- MAC layer (IEEE802.11)
  - DCF (Distributed Coordination Function) mode / Ad Hoc mode.
  - PCF (Point Coordination Function) mode / Infrastructure mode.
  - Partially implemented.
- Physical layer (IEEE802.11)
  - DSSS (Direct Sequence Spread Spectrum).
  - FHSS (Frequency-Hopping Spread Spectrum); not implemented.
  - IR (Infrared); not implemented.
- Radio Propagation Model
  - Friss-space model.
  - Two-ray model.
  - Shadowing model.
  - Omni directional antenna.

Simulation 1:

A simple wireless simulation

simple-wireless.tcl

Scenario

- 2mobile nodes
- moving within 500mX500m flat topology
- using DSDV ad hoc routing protocol
- TwoRayGround mobility model
- TCP and CBR traffic

Step 1

```
# Define options
#======================================================
set val(chan)           Channel/WirelessChannel     ;# channel type
set val(prop)           Propagation/TwoRayGround    ;# radio-propagation model
set val(netif)          Phy/WirelessPhy             ;# network interface type
set val(mac)            Mac/802_11                  ;# MAC type
set val(ifq)            Queue/DropTail/PriQueue     ;# interface queue type
set val(ll)             LL                          ;# link layer type
set val(ant)            Antenna/OmniAntenna         ;# antenna model
set val(ifqlen)         50                          ;# max packet in ifq
set val(nn)             2                           ;# number of mobilenodes
set val(rp)             DSDV                        ;# routing protocol
```

Step 2

```
# Initialize Global Variables
#
set ns_          [new Simulator]
set tracefd      [open simple.tr w]
$ns_ trace-all $tracefd

set namtrace [open simple-wireless-out.nam w]        ;# for nam tracing
$ns_ namtrace-all-wireless $namtrace  500 500
```

Step 3

```
# set up topography object
set topo        [new Topography]

$topo load_flatgrid 500 500
```

Step 4

God (General Operations Director)

```
# Create God
#
create-god $val(nn)
```

Step 5.1

```
#  Create the specified no. of mobile nodes [$val(nn)]
#  Configure node
      $ns_ node-config -adhocRouting $val(rp) \
                       -llType $val(ll) \
                       -macType $val(mac) \
                       -ifqType $val(ifq) \
                       -ifqLen $val(ifqlen) \
                       -antType $val(ant) \
                       -propType $val(prop) \
                       -phyType $val(netif) \
                       -channelType $val(chan) \
                       -topoInstance $topo \
                       -agentTrace ON \
                       -routerTrace ON \
                       -macTrace OFF \
                       -movementTrace OFF
```

Step 5.2

```
for {set i 0} {$i < $val(nn) } {incr i} {
        set node_($i) [$ns_ node]
        $node_($i) random-motion 0              ;# disable random motion
}
```

Step 6

```
# Provide initial (X,Y, for now Z=0) co-ordinates for mobilenodes
#
$node_(0) set X_ 5.0
$node_(0) set Y_ 2.0
$node_(0) set Z_ 0.0

$node_(1) set X_ 390.0
$node_(1) set Y_ 385.0
$node_(1) set Z_ 0.0
```

Step 7

```
# Now produce some simple node movements
# Node_(1) starts to move towards node_(0)
#
$ns_ at 50.0 "$node_(1) setdest 25.0 20.0 15.0"
$ns_ at 10.0 "$node_(0) setdest 20.0 18.0 1.0"

# Node_(1) then starts to move away from node_(0)
$ns_ at 100.0 "$node_(1) setdest 490.0 480.0 15.0"
```

Step 8

```
# Setup traffic flow between nodes
# TCP connections between node_(0) and node_(1)

set tcp [new Agent/TCP]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(0) $tcp
$ns_ attach-agent $node_(1) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 10.0 "$ftp start"
```

Step 9

```
# Tell nodes when the simulation ends
#
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at 150.0 "$node_($i) reset";
}
$ns_ at 150.0 "stop"
$ns_ at 150.01 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
}
```

Step 10

```
puts "Starting Simulation..."
$ns_ run
```

Outline

- Basic Ad Hoc Network
- Ad Hoc Network with
  - Connection pattern file generator: cbrgen.tcl
  - Mobility file generator: setdest & calcdest

Setdest
A Node-Movement Generator

- Generating idea

  Node moves randomly. (distribution of nodes: uniform)
- Location

  ~ns/indep-utils/cmu-scen-gen/setdest/setdest{.cc; .h}
- Command format

  setdest [-n ##] [-p ##] [-s ##] [-t ##] [-x ##] [-y ##]
- Option explanation

  n: number of nodes; p: pause time; s: maximum speed;

  t: simulation time; x: maximum x; y: maximum y
- Build your own node-movement generator

Cbrgen.tcl
A CBR Traffic Generator

- What is CBR?

  Constant Bit Rate.
- Generating idea

  Randomly pick up node pairs as sources and destinations.
- Location

  ~ns/indep-utils/cmu-scen-gen/cbrgen.tcl
- Command format

  ns cbrgen.tcl [-type ##] [-nn ##] [-seed ##] [-mc ##][-rate ##]

Cbrgen.tcl (cont.)

- Option explanation

  type: traffic/connection type. Must be tcp or cbr.

  nn: number of nodes.

  seed: seed for generating random number. It is used to generate the random starting time of the traffic.

  mc: maximum number of connections.

  rate: packet rate = 1 / packet interval

- Generate real random traffic

  cbrgen.tcl does not generate the real random traffic.

Simulation 2:
Wireless simulation with pattern file

wireless1.tcl

Scenario

- 3 mobile nodes
- moving within 670mX670m flat topology
- using DSR ad hoc routing protocol
- TwoRayGround mobility model
- TCP and CBR traffic

Step 1

```
# Define options
    set val(chan)     Channel/WirelessChannel
    set val(prop)     Propagation/TwoRayGround
    set val(netif)    Phy/WirelessPhy
    set val(mac)      Mac/802_11
    set val(ifq)      Queue/DropTail/PriQueue
    set val(ll)       LL
    set val(ant)      Antenna/OmniAntenna
    set val(x)        670                ;# X dimension of the topography
    set val(y)        670                ;# Y dimension of the topography
    set val(ifqlen)   50                 ;# max packet in ifq
    set val(seed)     0.0
    set val(adhocRouting)   DSR
    set val(nn)       3                  ;# how many nodes are simulated
    set val(cp)       "../mobility/scene/cbr-3-test"
    set val(sc)       "../mobility/scene/scen-3-test"
    set val(stop)     400.0              ;# simulation time
```

Step 2

```
# create simulator instance

    set ns_                         [new Simulator]

    # setup topography object

    set topo            [new Topography]

    # create trace object for ns and nam

    set tracefd         [open wireless1-out.tr w]
    set namtrace    [open wireless1-out.nam w]

    $ns_ trace-all $tracefd
    $ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
```

Step 3

```
# define topology
    $topo load_flatgrid $val(x) $val(y)

    #
    # Create God
    #
    set god_ [create-god $val(nn)]
```

Step 4

```
#global node setting

    $ns_ node-config -adhocRouting $val(adhocRouting) \
                     -llType $val(ll) \
                     -macType $val(mac) \
                     -ifqType $val(ifq) \
                     -ifqLen $val(ifqlen) \
                     -antType $val(ant) \
                     -propType $val(prop) \
                     -phyType $val(netif) \
                     -channelType $val(chan) \
                          -topoInstance $topo \
                          -agentTrace ON \
                     -routerTrace OFF \
                     -macTrace OFF
```

Step 5

```
#  Create the specified number of nodes [$val(nn)] and "attach" them
   #  to the channel.

   for {set i 0} {$i < $val(nn) } {incr i} {
          set node_($i) [$ns_ node]
          $node_($i) random-motion 0      ;# disable random motion
   }
```

Step 6

```
# Define node movement model
    #
    puts "Loading connection pattern..."
    source $val(cp)

    #
    # Define traffic model
    #
    puts "Loading scenario file..."
    source $val(sc)
```

Step 7

```
# Define node initial position in nam

    for {set i 0} {$i < $val(nn)} {incr i} {

        # 20 defines the node size in nam, must adjust it according to your scenario
        # The function must be called after mobility model is defined

        $ns_ initial_node_pos $node_($i) 20
    }
```

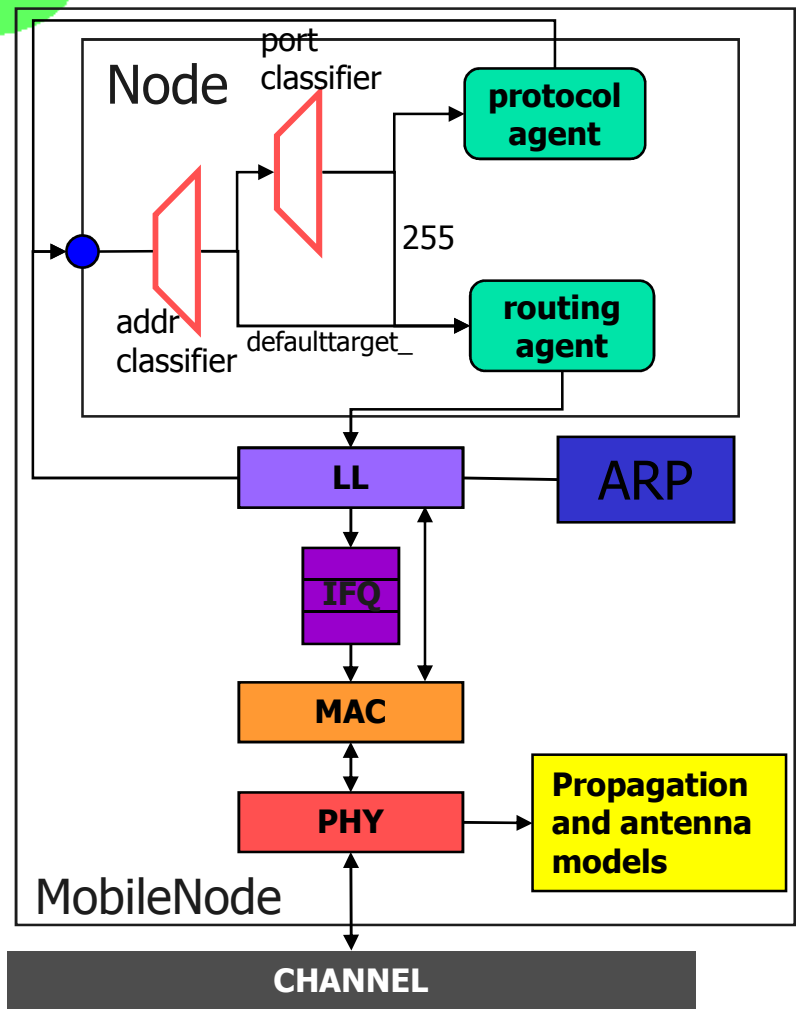
Step 8

```
# Tell nodes when the simulation ends
    #
    for {set i 0} {$i < $val(nn) } {incr i} {
        $ns_ at $val(stop).0 "$node_($i) reset";
    }

    $ns_ at  $val(stop).0002 "puts \"NS EXITING...\" ; $ns_ halt"

    puts $tracefd "M 0.0 nn $val(nn) x $val(x) y $val(y) rp $val(adhocRouting)"
    puts $tracefd "M 0.0 sc $val(sc) cp $val(cp) seed $val(seed)"
    puts $tracefd "M 0.0 prop $val(prop) ant $val(ant)"

    puts "Starting Simulation..."
    $ns_ run
```

Mobile Node Abstraction

- Location
  - Coordinates (x,y,z)
- Movement
  - Speed, direction, starting/ending location, time ...

Portrait of A Mobile Node



Classifier: Forwarding

Agent: Protocol Entity

Node Entry

LL

LL: Link layer object

IFQ

IFQ: Interface queue

MAC

MAC: Mac object

PHY

PHY: Net interface

Mobile Node: Components

- Link Layer
  - Same as LAN, but with a separate ARP module
- Interface queue
  - Give priority to routing protocol packets
- Mac Layer
  - IEEE 802.11
  - RTS/CTS/DATA/ACK for all unicast packets
  - DATA for all broadcast packets

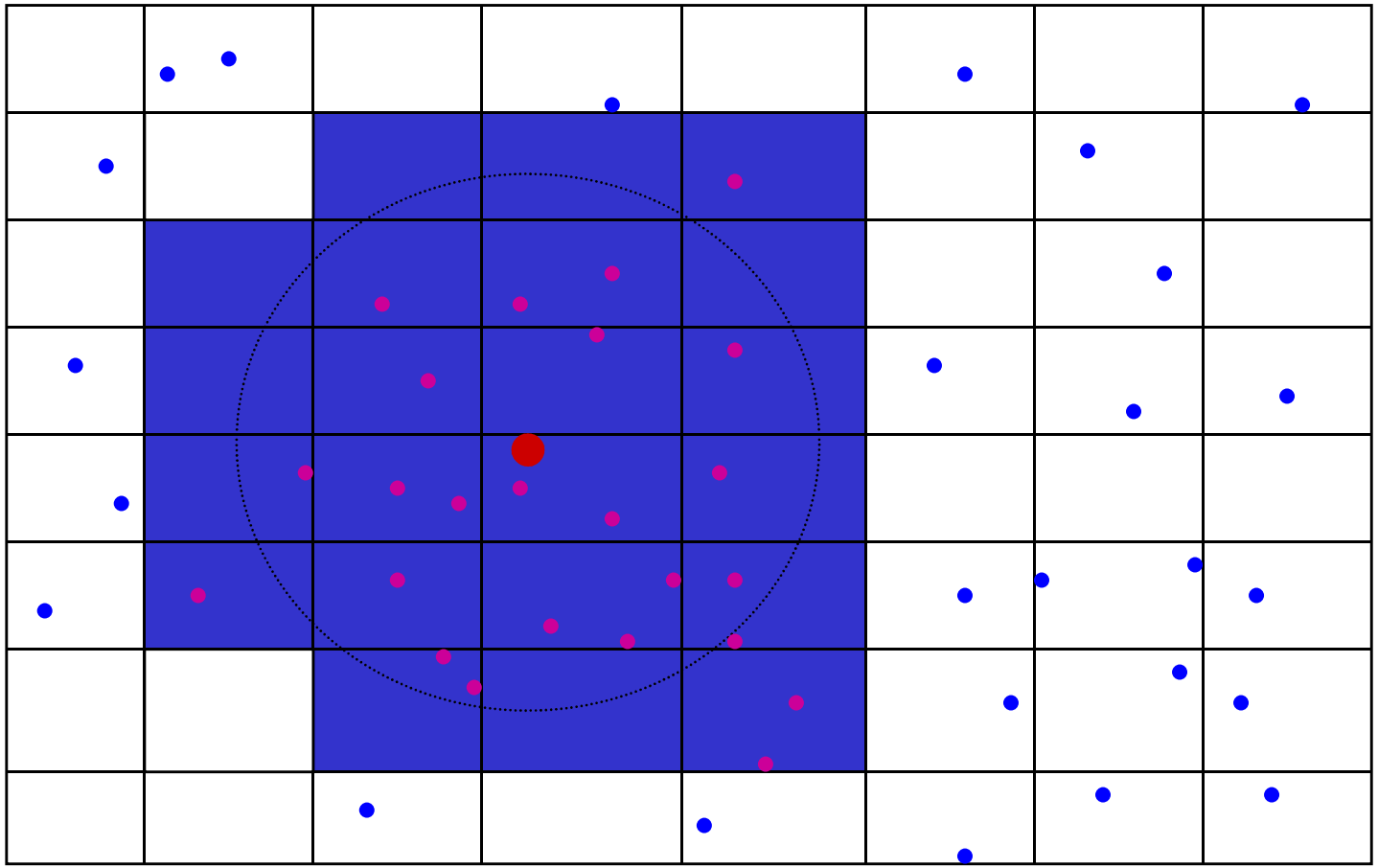
Mobile Node: Components

- Network interface (PHY)
  - Parameters based on Direct Sequence Spread Spectrum (WaveLan)
  - Interface with: antenna and propagation models
  - Update energy: transmission and reception
- Radio Propagation Model
  - Friss-space attenuation($1/r^2$) at near distance
  - Two-ray Ground ($1/r^4$) at far distance
- Antenna
  - Omni-directional, unity-gain

Wireless Channel

- Duplicate packets to all mobile nodes attached to the channel except the sender
- It is the receiver's responsibility to decide if it will accept the packet
  - Collision is handled at individual receiver
  - $O(N^2)$ messages → grid keeper

Grid-keeper: An Optimization



Trace File

r 100.381997477 _1_ AGT --- 82 tcp 1060 [13a 1 0 800] ------- [0:0 1:0 32 1] [32 0] 1 0

r: receive event;
100.381997477: time stamps;
_1_ : nod:e 1;
AGT: trace generated by agent;
82: event(pkt) id;
tcp: tcp packet;
1060: packet size;
13a:: expected duration of pkt transmission (not working);
1 : sender mac id;
0: transmitter mac id;
800: pkt type IP;
0:0: sender address:port#;
1:0: receiver address:port#;
32: TTL;
1: next hop address;
[32 0] : TCP sequence #, ack #.