Network Simulator Tutorial

About This Tutorial

- Based on
  - “NS Fundamentals” by Padmaparna Haldar and Xuan Chen, ISI, University of Southern California
  - “Network Simulator Tutorial” by Vacha Dave, University of Texas at Austin

What is Network Simulation

- Simulate the network behavior
  - From physical layer to application layer

- Mostly used to evaluate the performance of computer networks

Why Simulation?

- Experiments with real system
  - Availability
  - Scalability
  - Cost
  - Flexibility
- Simulations helps
  - Test new protocol
    - Explore the design space
  - Modify existing protocol
    - Performance tuning

Outline

- Introduction of NS2
- Using NS2
- Documentation
- Conclusion

Installation

- Download from http://www.isi.edu/nsnam/ns/ns-build.html
- Getting the pieces
  - tcl/tk, otcl, tclcl, ns-2, nam-1, Xgraph etc
- ns-allinone package

http://sourceforge.net/projects/nsnam/files/allinone/ns-allinone-2.35/

Installation

- Download from http://www.isi.edu/nsnam/ns/ns-build.html
- Getting the pieces
  - tcl/tk, otcl, tclcl, ns-2, nam-1, Xgraph etc
- ns-allinone package

http://sourceforge.net/projects/nsnam/files/allinone/ns-allinone-2.35/

https://www.howtoforge.com/tutorial/ns2-network-simulator-on-ubuntu-14.04/

Goals of NS2

- Support networking research and education
  - Protocol design, traffic studies, etc
  - Protocol comparison
- Provide a *collaborative* environment
  - Freely distributed, *open source*
    - Share code, protocols, models etc
  - *Allow easy comparison* of similar protocols
  - *Increase confidence* in results

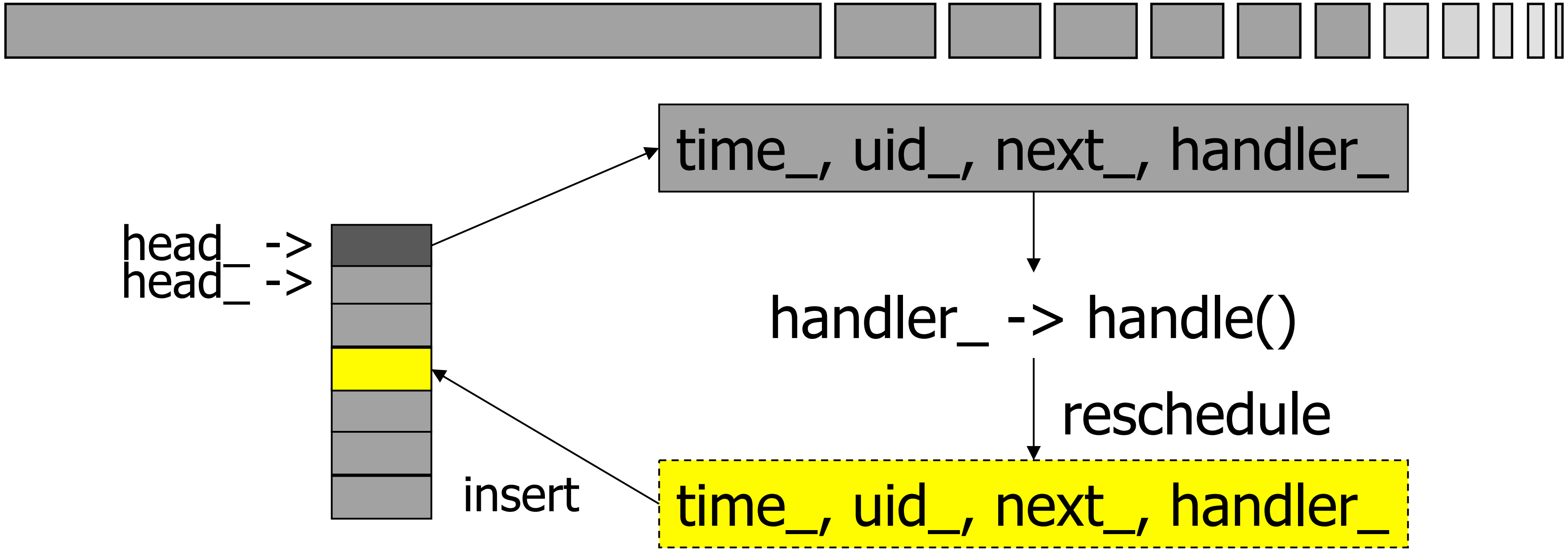
What NS2 can simulate?

- Wired network
  - Applications and Traffic model (HTTP,FTP,CBR…)
  - Transport protocol (UDP, TCP…)
  - Routing (DV, LS…) and Queuing (RED, FIFO…)
  - QoS
  - LANs
- Wireless network
  - Ad hoc routing and mobile IP
  - Propagation model/Energy model
  - WLAN (802.11)

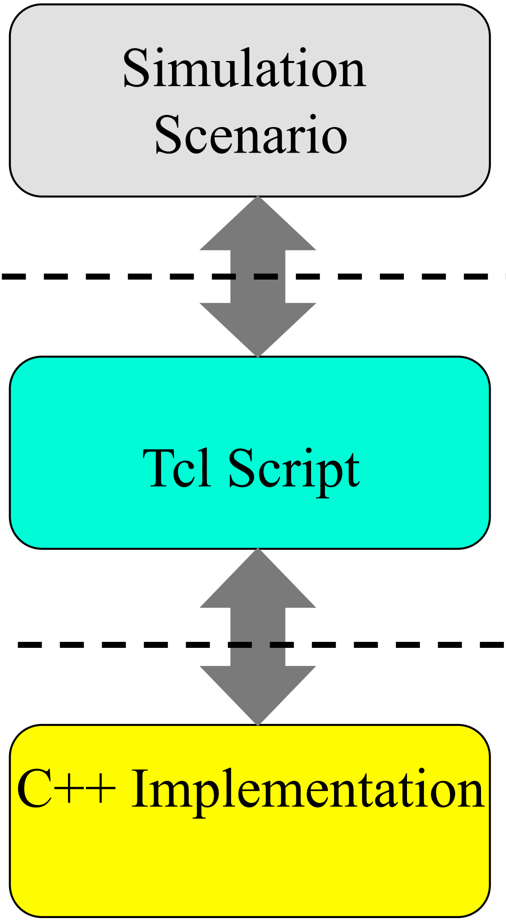
NS2 is a Discrete Event Simulator

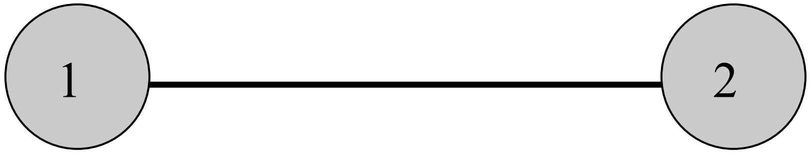
- Model world as *events*
  - Simulator has list of events
  - Scheduler: take next one, run it, until done
  - Each event happens in an instant of *virtual (simulated) time*, but takes an arbitrary amount of *real* time
- Ns uses simple model: single thread of control => no locking or race conditions to worry about

Discrete Event Scheduler



NS-2 Environment





```
set ns_ [new Simulator]
set node_(0) [$ns_ node]
set node_(1) [$ns_ node]
```

```
class MobileNode : public Node
{
        friend class PositionHandler;
public:
         MobileNode();
                  •
                  •
}
```

Why Two Languages? (Tcl & C++)

- “data” / control separation
  - Compiled vs interpreted

- C++ for “data”:
  - When run-time speed matters
  - Per packet processing, core of *ns*
  - Detailed protocol implementation

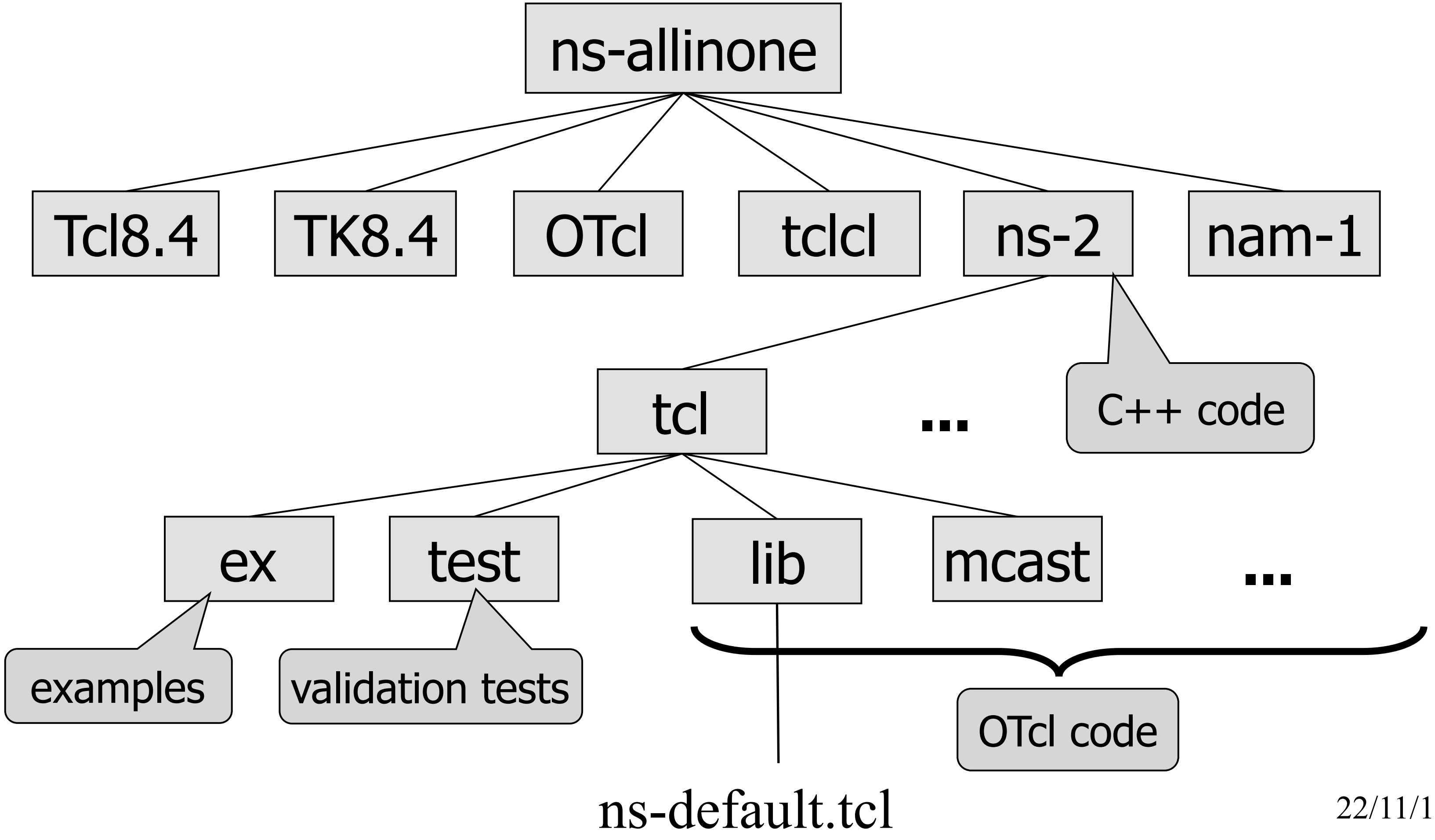
Why Two Languages? (Tcl & C++)

➢ OTcl for control:

- When turn-around time matters
- Simulation scenario configurations
- Manipulating existing C++ objects
- Fast to write and change

+ Running vs. writing speed
- Learning and debugging (two languages)

NS-2 Directory Structure



Outline

- Introduction of NS2
- Using NS2
- Documentation
- Conclusion

Hello world!

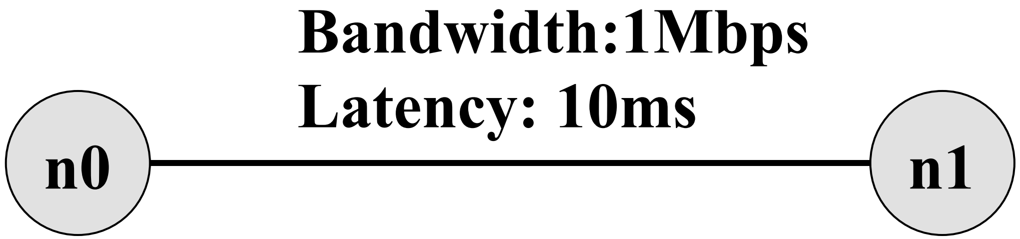
```
#Create scheduler
set ns [new Simulator]

#Schedule event
$ns at 1 “puts \“Hello World!\””
$ns at 1.5 “exit”

#Start scheduler
$ns run
```

(save as hello.tcl, then run “ns hello.tcl”)

Basics of using NS2

- Create Network topology
- Define connections (e.g. TCP or UDP)
- Add traffic load (e.g. CBR)
- Run the simulation
- Observe network behavior
  - Post-processing (output is in form of trace files, or it can be visualized by using NAM).

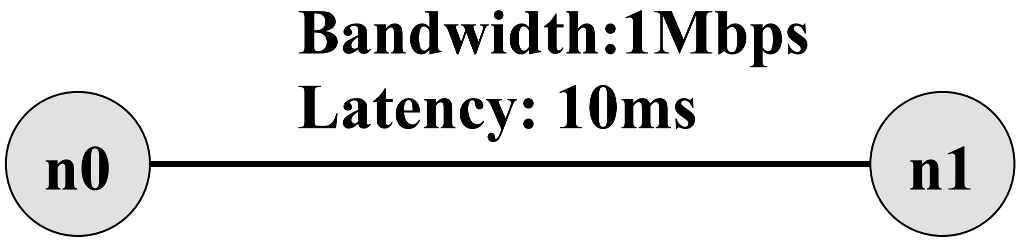
A Simple Example



Creating the topology

- Nodes
  - Set properties like queue length, location
  - Protocols, routing algorithms
- Links
  - Set types of link – Simplex, duplex, wireless, satellite
  - Set bandwidth, latency etc.
- Done through tcl Scripts

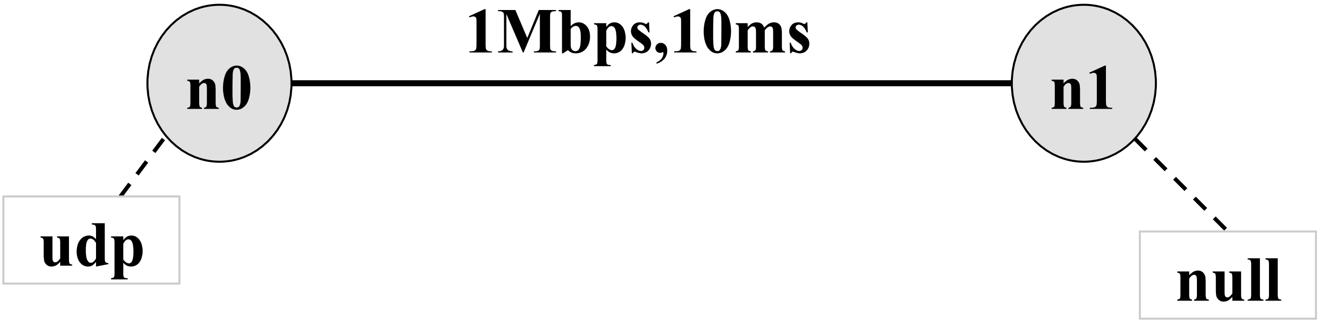
Creating the topology



Bandwidth:1Mbps
Latency: 10ms

```
#create two nodes
set n0 [$ns node]
set n1 [$ns node]

#create a duplex link between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
```

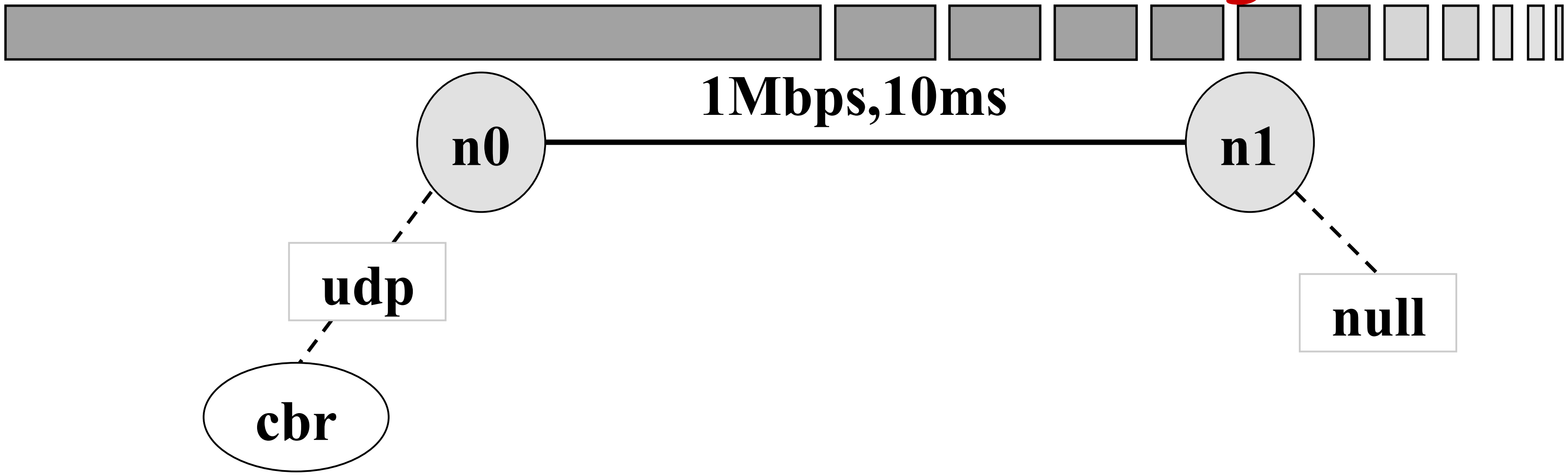
Adding Connection



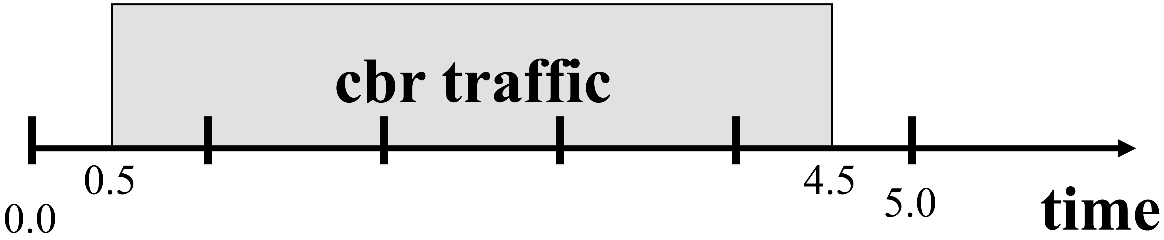
Adding Connection

```
#create a udp agent and attach it to node n0
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

#create a Null agent(a destination) and attach it to node n1
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0

#Connect the source to the destination
$ns connect $udp0 $null0
```

Adding Traffic



Packet Size: 500 bytes
interval: 0.02s



Adding Traffic

```
#Create a CBR traffic source and attach
it to udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.02
$cbr0 attach-agent $udp0

#Schedule events for CBR traffic
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
```

Putting it together..

```
#create a new simulator object
set ns [new Simulator]

#open the nam trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#define a 'finish' procedure
proc finish {} {
    global ns nf
    $ns flush-trace
    #close the trace file
    close $nf
    #execute nam on the trace file
    exec nam out.nam &
    exit 0
}
```

```
#create two nodes
set n0 [$ns node]
set n1 [$ns node]

#create a duplex link between the nodes
$ns duplex-link $n0 $n1 1Mb 10ms DropTail

#create a udp agent and attach it to node n0
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0

#create a Null agent(a traffic sink) and attach it
to node n1
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0

#Connect the traffic source to the sink
$ns connect $udp0 $null0
```

```
#Create a CBR traffic source and attach it to udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.02
$cbr0 attach-agent $udp0

#Schedule events for CBR traffic
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"

#call the finish procedure after 5 secs of
simulated time
$ns at 5.0 "finish"

#run the simulation
$ns run
```

Demo

Observing Network Behavior

- Observe behavior by tracing “events”

  e.g., packet received, packet drop etc.

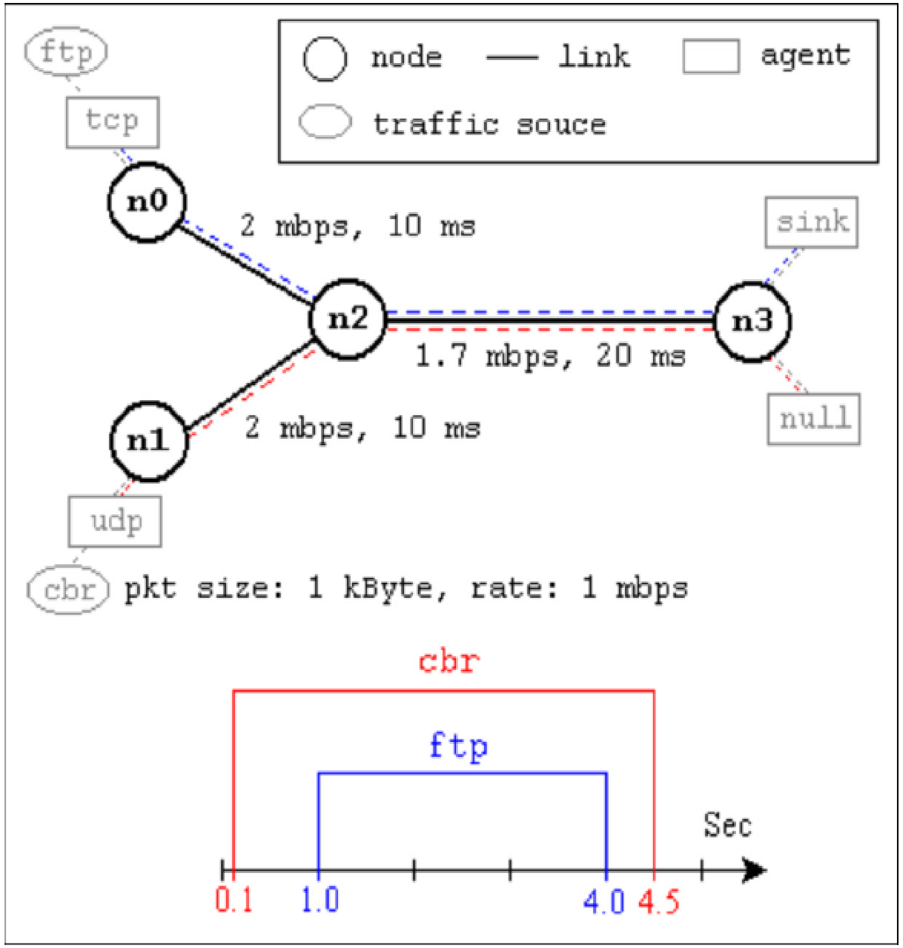
time

```
+ 0.84 0 1 cbr 500 ------- 0 0.0 1.0 17 17
- 0.86 0 1 cbr 500 ------- 0 0.0 1.0 9 9
+ 0.86 0 1 cbr 500 ------- 0 0.0 1.0 18 18
r 0.87 0 1 cbr 500 ------- 0 0.0 1.0 8 8
+ 0.88 0 1 cbr 500 ------- 0 0.0 1.0 19 19
d 0.88 0 1 cbr 500 ------- 0 0.0 1.0 19 19
- 0.9 0 1 cbr 500 ------- 0 0.0 1.0 10 10
```

Src Dst
(Link Layer)

Src Dst
Address, Port
(Network
Layer)

A Second Scenario *



* Taken from Mark Greis NS2 tutorial

```
#Create a simulator object
set ns [new Simulator]

#Define different colors for data flows (for NAM)
$ns color 1 Blue
$ns color 2 Red

#Open the NAM trace file
set nf [open out.nam w]
$ns namtrace-all $nf

#Define a 'finish' procedure
proc finish {} {
        global ns nf
        $ns flush-trace
        #Close the NAM trace file
        close $nf
        #Execute NAM on the trace file
        exec nam out.nam &
        exit 0
}
```

```
#Create four nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#Create links between the nodes
$ns duplex-link $n0 $n2 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 1.7Mb 20ms DropTail

#Set Queue Size of link (n2-n3) to 10
$ns queue-limit $n2 $n3 10

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right
```

```
#Monitor the queue for link (n2-n3)(for NAM)
$ns duplex-link-op $n2 $n3 queuePos 0.5

#Setup a TCP connection
set tcp [new Agent/TCP]
$tcp set class_ 2
$ns attach-agent $n0 $tcp
set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink
$ns connect $tcp $sink
$tcp set fid_ 1

#Setup a FTP over TCP connection
set ftp [new Application/FTP]
$ftp attach-agent $tcp
```

```
#Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set null [new Agent/Null]
$ns attach-agent $n3 $null
$ns connect $udp $null
$udp set fid_ 2

#Setup a CBR over UDP connection
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set packet_size_ 1000
$cbr set rate_ 1mb
```

```
#Schedule events for the CBR and FTP agents
$ns at 0.1 "$cbr start"
$ns at 1.0 "$ftp start"
$ns at 4.0 "$ftp stop"
$ns at 4.5 "$cbr stop"

#Call the finish procedure
#after 5 seconds of simulation time
$ns at 5.0 "finish"


#Run the simulation
$ns run
```

Demo

Outline

- Introduction of NS2
- Using NS2
- Documentation
- Conclusion

Documentation – NS2 Documentation

- NS2 Manual
  - Information about Otcl interpreter, C++ class hierarchy, parameters for various protocols
  - http://www.isi.edu/nsnam/ns/doc/index.html
  - Very detailed, useful when looking for something specific, like:
    - What are the shadowing models available for wireless? How do I select them?
    - How do I make my routing strategy to be Distance Vector routing?

Documentation - NS2 documentation

- NS2 Tutorial by Marc Greis

  http://www.isi.edu/nsnam/ns/tutorial/index.html

  - From simple tcl examples to how to add a protocol in NS2
  - Wireless simulations

Documentation – NS2 Documentation

- NS2 for beginners
  - http://www-sop.inria.fr/members/Eitan.Altman/COURS-NS/n3.pdf
  - More detailed than Marc Greis' Tutorial
  - Examples:
    - » What does each line of a tcl script do?
    - » Most common examples of trace formats that are useful

Documentation – Tcl Documentation

- Tcl Tutorial
  - http://www.tcl.tk/man/tcl8.5/tutorial/tcltutorial.html
- Tcl Manual
  - All commands and their explanation
  - http://www.tcl.tk/man/tcl8.6/TclCmd/contents.htm

When things go wrong..

- Googling for the problem!
  - Extensive NS2 mailing lists
  - Chances that other people have had the same problem are very high
  - Responsive forums

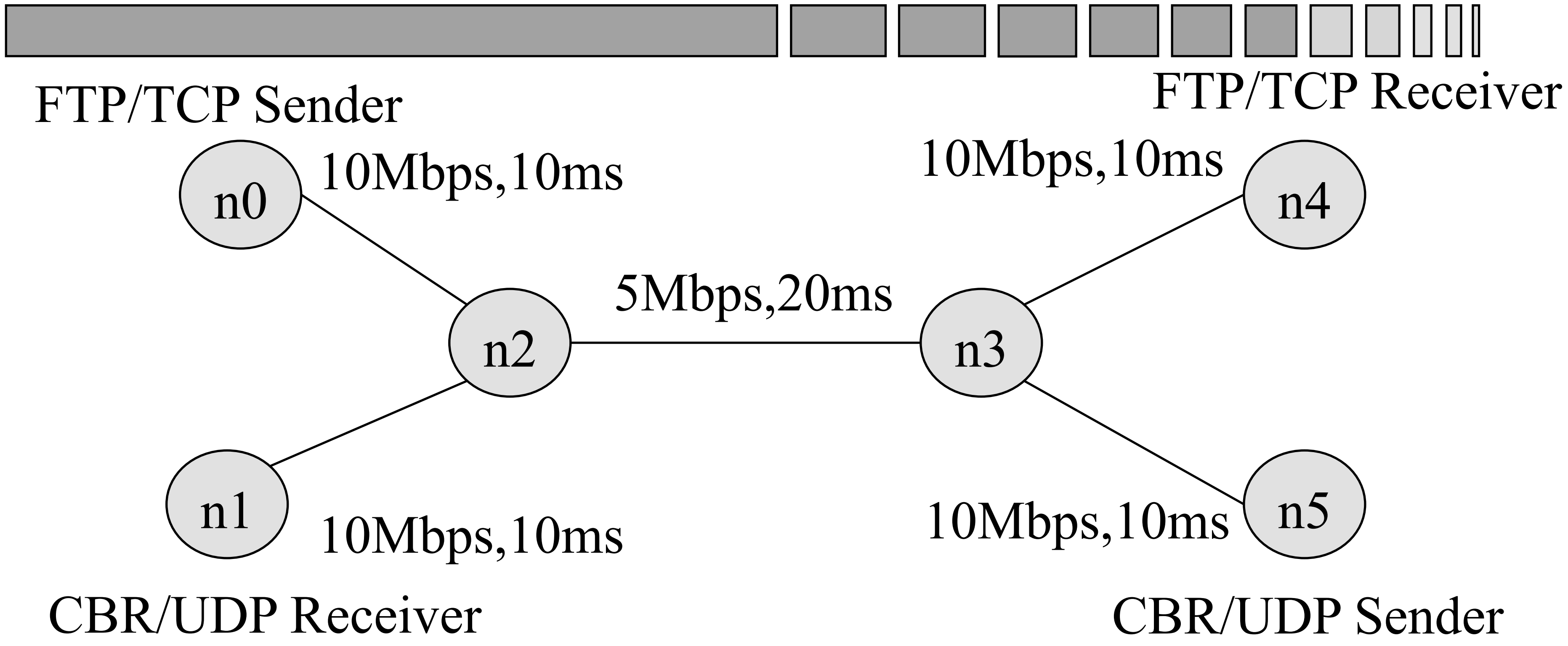
Other simulators

- QualNet
- OPNET
- GloMoSim (wireless only)
- SSFNET
- PDNS (parallel/distributed ns)
- JavaSim
- OMNET++

Conclusion

- Simulation is an abstraction of real world
- Network simulator is essential for research and education in networking area
- Can help the deep understanding of network protocols
- Mostly used for evaluating the performance of new protocols

Lab Specifications



- FTP start time 0.5 end time 4.0, CBR start time 1.0 end time 4.5 [rate 5Mbps], Total simulation time 5.0

Lab Specifications

- Submit your TCL file Plus
  - The graph of TCP and UDP flow throughput vs time
  - Total packet losses for TCP and UDP