Università degli Studi del Sannio
facoltà di Ingegneria
cdl Ing. Informatica

Simulazione di tecniche di scheduling di un router per QoS applicate al VoIP

Guadagno Potito
Pagliarulo Luana
Parente Cosimo
Zardo Pietro Giuseppe

a.a. 2006/2007

Obiettivi

Lo scopo del progetto è la simulazione via software del comportamento di una rete VoIP in presenza di traffico di diverso tipo proveniente da varie fonti.

- ✓ Utilizzo di tool per la simulazione software di una rete VoIP
  - ✓ NetworkSimulator "NS2" - Nscript
- ✓ Testing delle politiche di accodamento e di scheduling
  - ✓ DropTail – RandomEarlyDetection
  - ✓ RoundRobin – WeightedRoundRobin – Priority
- ✓ Monitoring ed analisi dei risultati
  - ✓ Pacchetti persi – Pacchetti scartati – Pacchetti ricevuti
  - ✓ Ritardo VoIP
  - ✓ Throughput
  - ✓ Perdita dei pacchetti nel tempo in funzione del flusso

NS2
Network Simulator

NS è un simulatore di rete IP-based open-source, implementato dall'Università di Berkeley nel 1989.

- ✓ Utilizza come linguaggio di programmazione il *Tcl* (Tool Command Language), per descrivere tutte le operazioni necessarie alla definizione di una simulazione. È possibile impostare lo scenario di rete, oggetto dello studio di simulazione e la sua evoluzione temporale in uno script *OTcl*
- ✓ Il programma simula il funzionamento della rete descritta nel file di configurazione della simulazione e produce l'insieme dei file di risultati.
- ✓ Attraverso il modulo *NAM* (Network Animator Module) è possibile ricostruire la simulazione attraverso una simulazione ed avere un riscontro visivo del funzionamento della rete e delle interazioni tra i componenti.

NS2
Network Simulator

Per effettuare una simulazione con NS occorre:

✓ *definire un oggetto di simulazione*

```
#Create a 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 $nf
     exec nam out.nam &
     exit 0 }
```

✓ *descrivere lo scenario simulativo* (nodi, link, sorgenti di traffico, ecc.)

```
#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
```

✓ *eseguire la simulazione*

```
#Run the simulation
$ns run
```

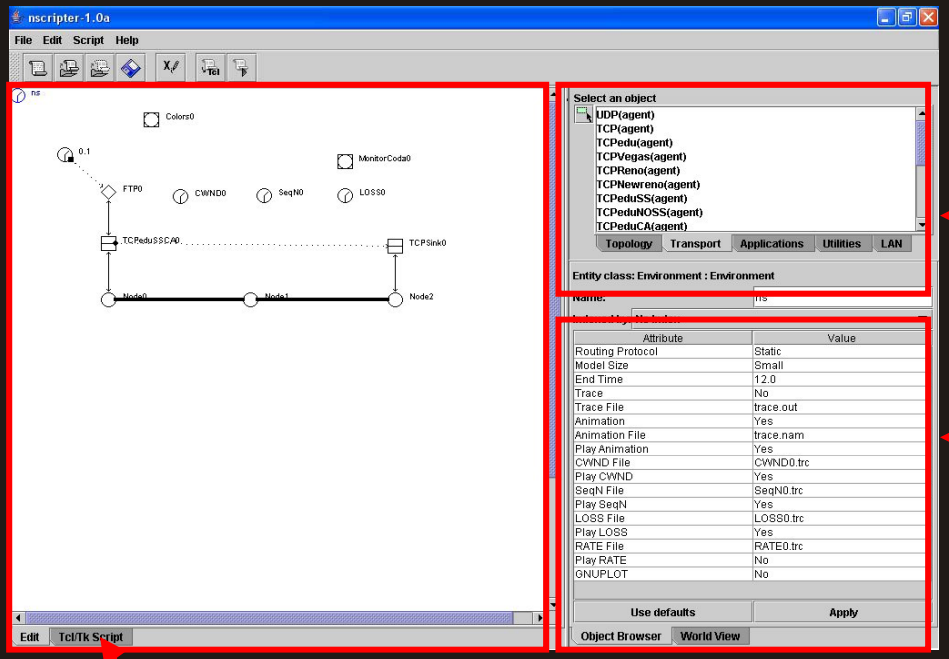
✓ *visualizzare i risultati*

- ✓ File di traccia e grafici - Tool grafico TraceGraph

Nscript

- ✓ Nscript è un'interfaccia grafica costruita interamente con il linguaggio di programmazione Java ed è un'interfaccia di supporto all'uso di NS2, in quanto serve per tradurre in codice TCL i grafici di topologie di rete.

1
Finestra dove si assembla graficamente la rete



2
Finestra dei menù (link, nodi, TCP, ..)

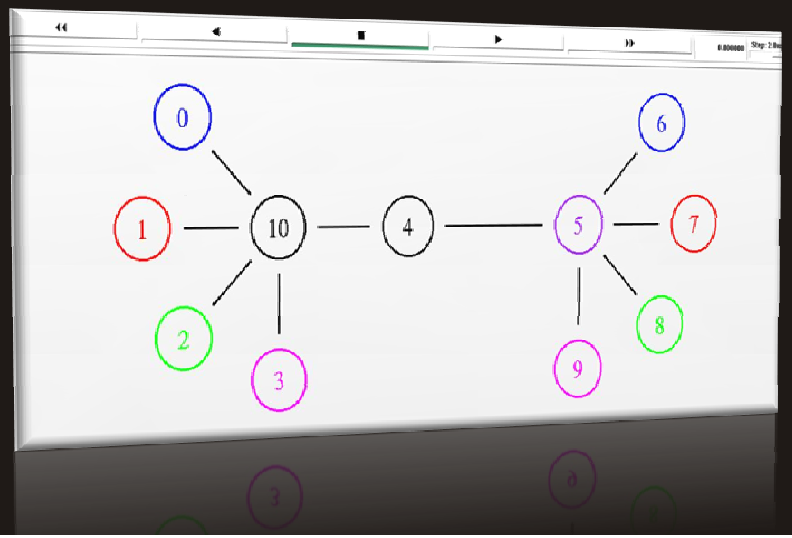
3
Finestra di impostazione dei parametri

Cliccando su "Tcl/tk script" si apre la pagina dello script.tcl

Definizione del progetto

- ✓ Architettura scelta :
  - ✓ modello *Differentiated Services*
- ✓ La topologia della rete oggetto della simulazione è caratterizzata da 4 sorgenti di traffico (nodi 0,1,2,3) , 4 destinatari (nodi 6,7,8,9) e 1 router.

Sorgente 0 :
Generatore di traffico HTTP su protocollo TCP – Pareto Distribution
Sorgente 1 :
Generatore di traffico VOIP su protocollo UDP – 218 byte – 20ms
Sorgente 2 :
Generatore di traffico ICMP – 38 byte – 50ms
Sorgente 3 :
Generatore di traffico PeerToPeer – 1500 byte – 10ms



Specifiche del progetto

Simulazione 1

- ✓ Nella prima simulazione abbiamo utilizzato una politica di accodamento FIFO di tipo "DropTail" che scarta un pacchetto in arrivo quando questo trova la coda piena. Tale politica se da un lato è facile da implementare, dall'altro ha diversi difetti:
  - ✓ le sorgenti perdono pacchetti quando la coda è piena determinando un aumento del ritardo end-to-end e del jitter con effetti negativi sulle applicazioni multimediali e in particolare su quelle interattive;
  - ✓ tra due flussi aventi lo stesso bitrate ma lunghezza media del burst differente, viene penalizzato quello con lunghezza media del burst più alta;
  - ✓ tutti i flussi TCP che mandano dati ad una coda piena subiscono perdita di pacchetti e contemporaneamente abbassano il bitrate per poi rialzarlo; questa sincronizzazione nel comportamento delle sorgenti abbassa l'efficienza nell'utilizzazione del bottleneck.

```
$ns simplex-link $edge $node4 100Mb 10ms DropTail
$ns simplex-link $node4 $edge 100Mb 10ms DropTail
# bottleneck link
$ns simplex-link $node4 $node5 1.2Mb 20ms DropTail
$ns simplex-link $node5 $node4 1.2Mb 20ms DropTail
$ns queue-limit $node4 $node5 25
```

Specifiche del progetto

Simulazione 2

- ✓ Per le altre simulazioni abbiamo testato le politiche di scheduling **RR**, **WRR** e Priority, con politica di accodamento **RED**. Con il comando addPolicyEntry configuriamo "l'edge router" per marcare i pacchetti che devono essere assegnati alle varie code.

- ✓ Simulazione 2.1 - RoundRobin

```
set qe2 [[$ns link $edge $node2] queue]
$qe2 addPolicyEntry [$node0 id] [$node4 id] Null 11
$qe2 addPolicerEntry Null 11

set qe2 [[$ns link $edge $node2] queue]
#$qe2 addPHBEntry $etichetta $queueNum $virtualQueueNum
$qe2 addPHBEntry 11 1 0
```

Per default le code sono gestite con politica di scheduling RoundRobin

Specifiche del progetto

Simulazione 2.2

- ✓ Per servire una coda con priorità maggiore rispetto ad un'altra occorre usare il metodo Weighted Round Robin (WRR) che assegna pesi diversi alle varie code. Come si vede nel frammento di codice seguente, dove la coda 0 (HTTP) verrà servita il 20% del tempo, quella 1 (VoIP) il 30%, quella 2 il 40% (ICMP) e la coda 3 (P2P) il restante 10%:

```
set q45 [[$ns link $node4 $node5] queue]
$q45 set numQueues_ 4
$q45 setNumPrec 2
$q45 setSchedularMode WRR
$q45 addQueueWeights 0 2
$q45 addQueueWeights 1 3
$q45 addQueueWeights 2 4
$q45 addQueueWeights 3 1
```

Specifiche del progetto

Simulazione 2.3

- ✓ L'ultima politica testata è la Priority che assegna la priorità in base all'ordine (dalla coda 0 alla coda 3) e che permette di assegnare ai vari flussi di dati una determinata banda massima, superata la quale il router scarterà i pacchetti provenienti da quella sorgente. Nel codice sottostante assegnamo, su un totale di 1.2 Mb di banda disponibile, un massimo di 50Kb per la coda 0 (ICMP), 100Kb per la 1 (VoIP), 350Kb per la coda 2 (HTTP), 700Kb per la coda 3 (P2P):

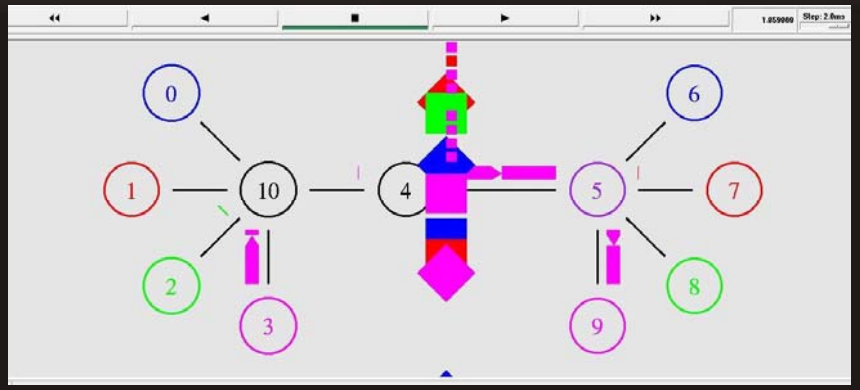
```
set q45 [[$ns link $node4 $node5] queue]
$q45 meanPktSize 400
$q45 set numQueues_ 4
$q45 setNumPrec 2
$q45 setSchedularMode PRI
# riga per il PRI
$q45 addQueueRate 2 350Kb
$q45 addQueueRate 1 100Kb
$q45 addQueueRate 0 50Kb
$q45 addQueueRate 3 700Kb
```

Specifiche del progetto

Esecuzione

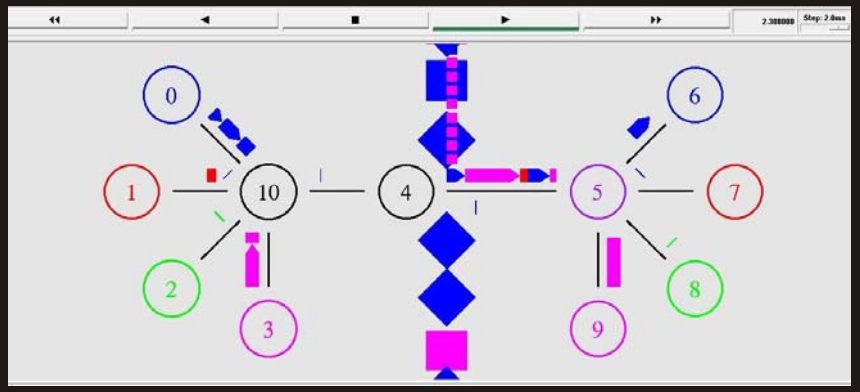
- ✓ All'istante "0.05" facciamo partire la sorgente "cbr2" ICMP
- ✓ All'istante "0.1" facciamo partire la sorgente "cbr1" VoIP
- ✓ All'istante "0.2" facciamo partire la sorgente "cbr3" P2P
- ✓ All'istante "0.2" facciamo partire la sorgente “Expoo_traffic” HTTP
- ✓ La simulazione dura complessivamente 10 secondi.
- ✓ Allo start i link sono liberi; man mano che le applicazioni cominciano a generare traffico, i link cominciano a saturarsi; il traffico “CBR3” P2P , è stato impostato in modo da saturare il link “bottleneck” (nodo 4 -> nodo 5) per verificare cosa succede in caso di carico eccessivo della rete. A questo punto si possono analizzare i comportamenti delle varie politiche di scheduling e come i pacchetti vengono scartati in base ai flussi e al riempimento delle code del router core.
- ✓ Le simulazioni sono state analizzate con il software Tracegraph, che ha permesso di ottenere dei grafici dettagliati a partire dai file di traccia generati da NS.

Politiche di accodamento 1



Drop-Tail

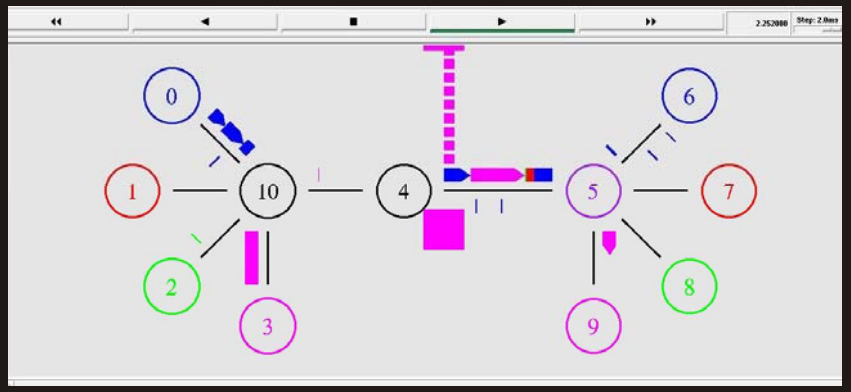
Scarta qualsiasi tipo di pacchetto quando la coda è piena.



RandomEarlyDetection – RoundRobin

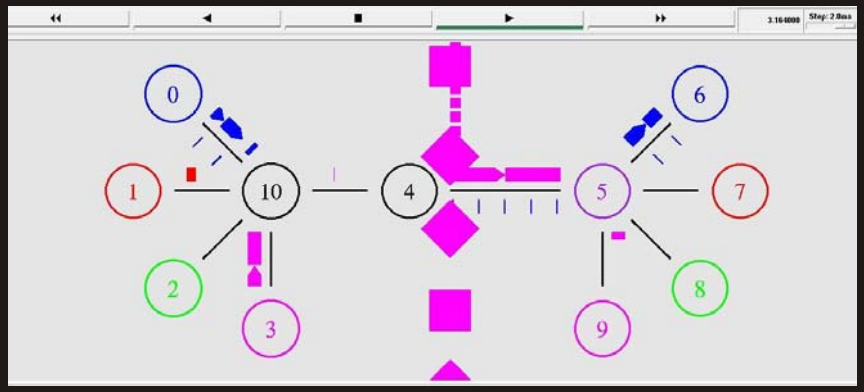
Inoltrando i pacchetti in maniera circolare, scarta quelli che provengono dalle sorgenti che inviano di più.

Politiche di accodamento 2



RandomEarlyDetection – WeightedRoundRobin

Servendo le code in base alla priorità assegnata, vengono scartati i pacchetti provenienti dalle sorgenti con tempo di utilizzo del canale inferiore.

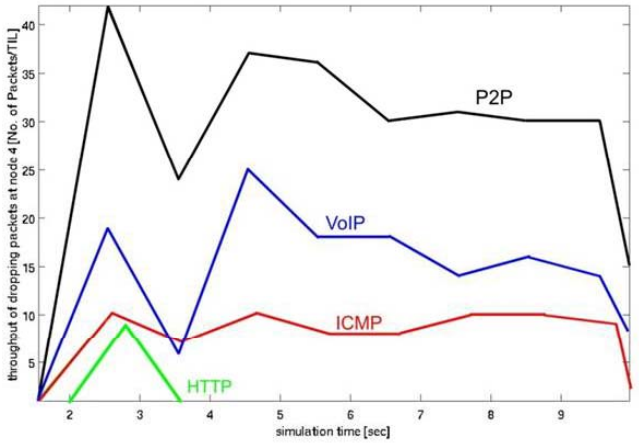


RandomEarlyDetection – Priority

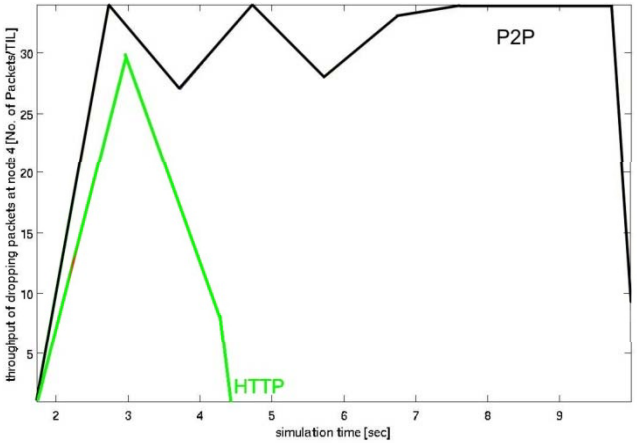
Servendo le code in base all'ordine ed alla larghezza di banda massima, vengono scartati i pacchetti provenienti dalle sorgenti con priorità più bassa.

Pacchetti Scartati 1

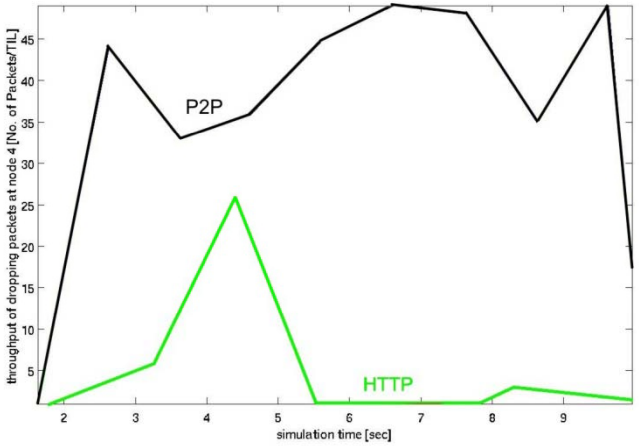
DropTail



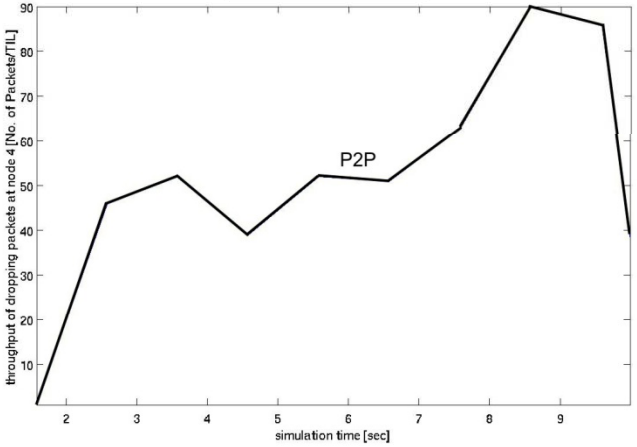
RoundRobin



WeightedRoundRobin

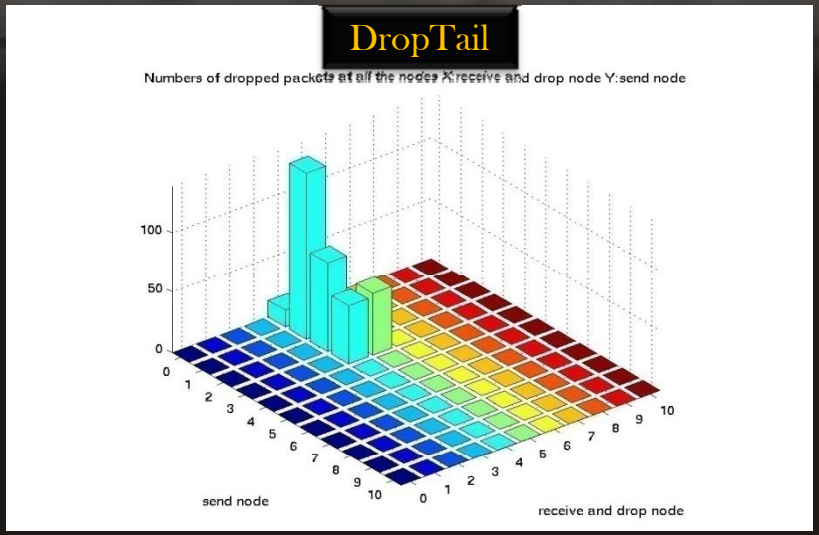


Priority

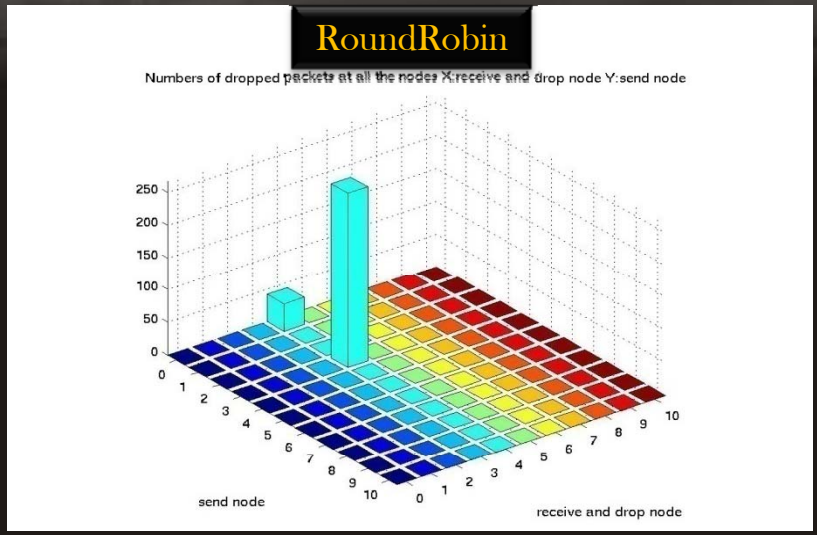


Pacchetti Scartati 2

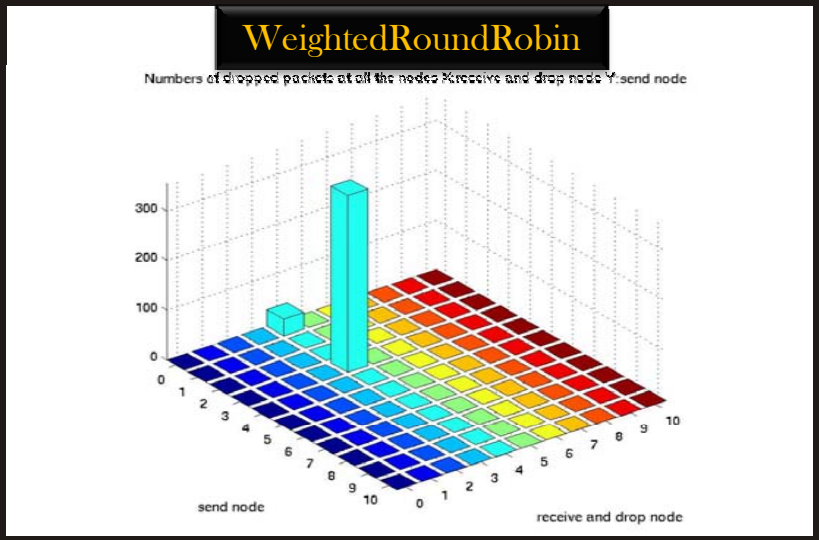
DropTail



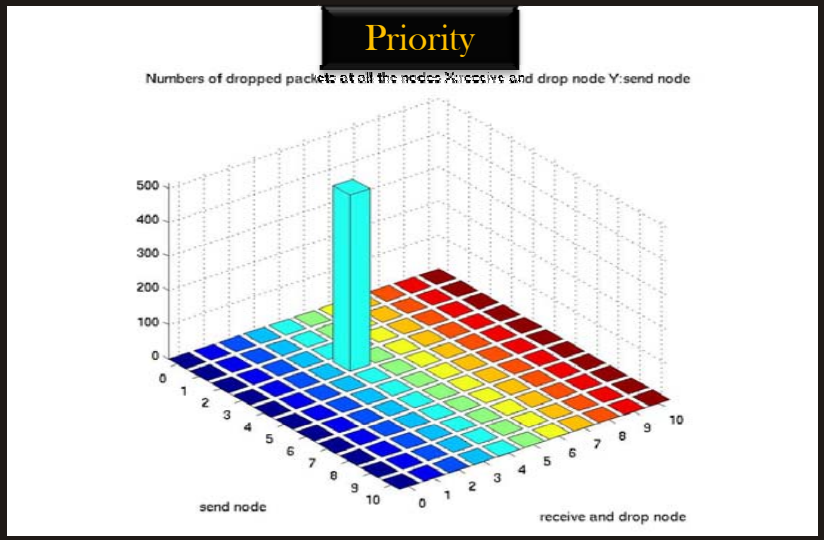
RoundRobin



WeightedRoundRobin

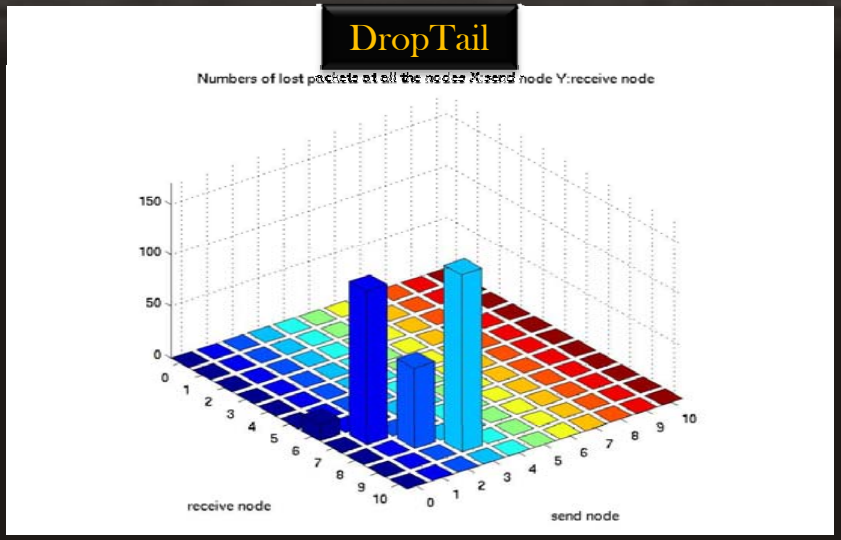


Priority

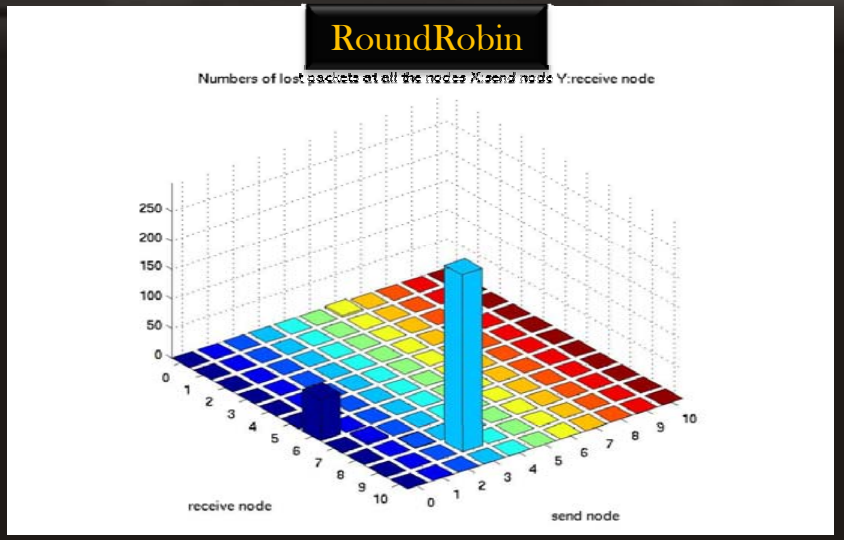


Pacchetti persi

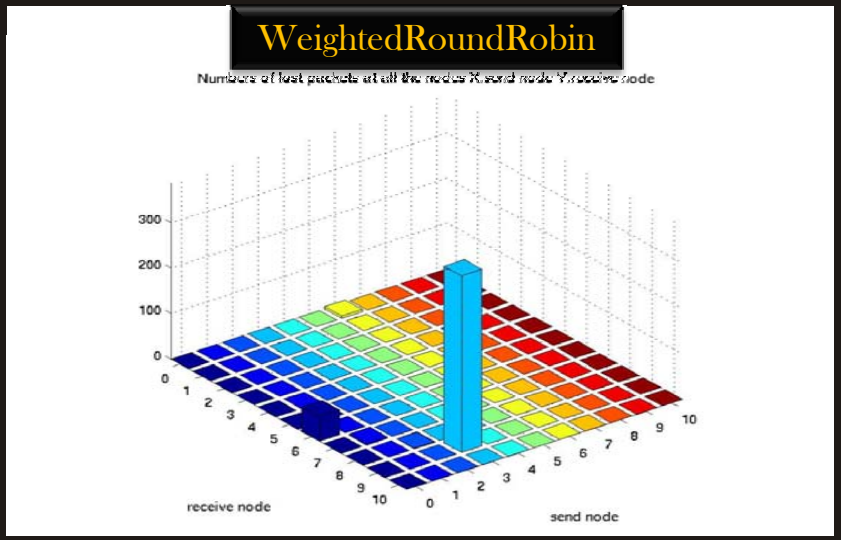
DropTail



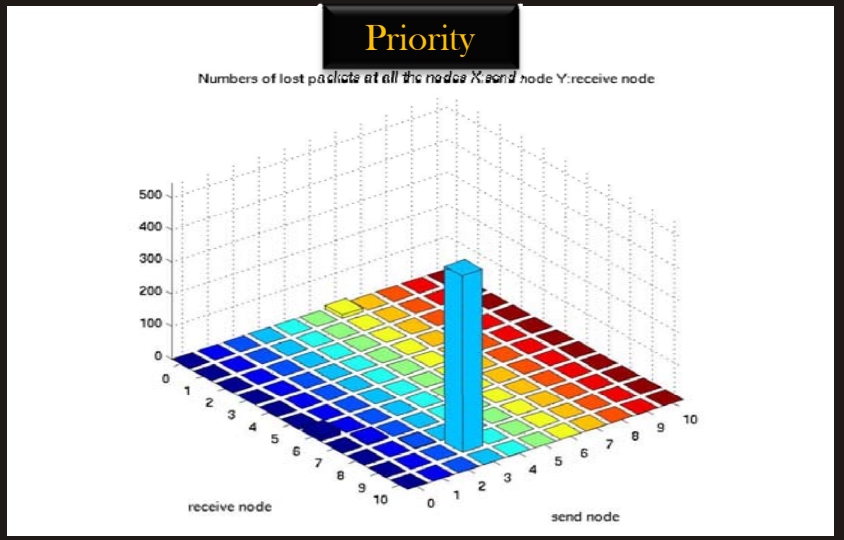
RoundRobin



WeightedRoundRobin

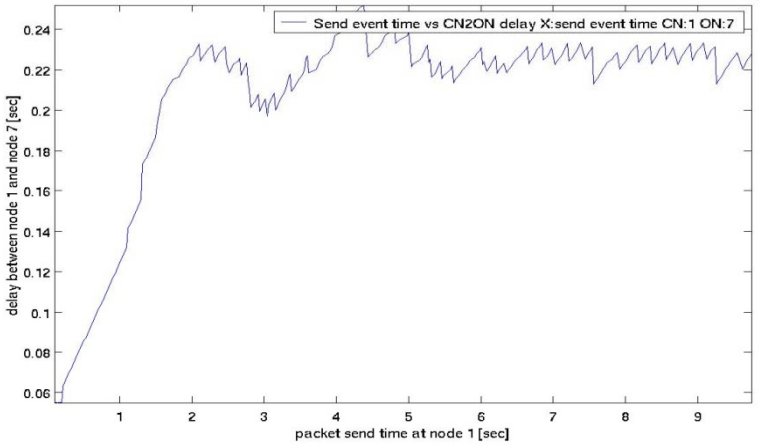


Priority

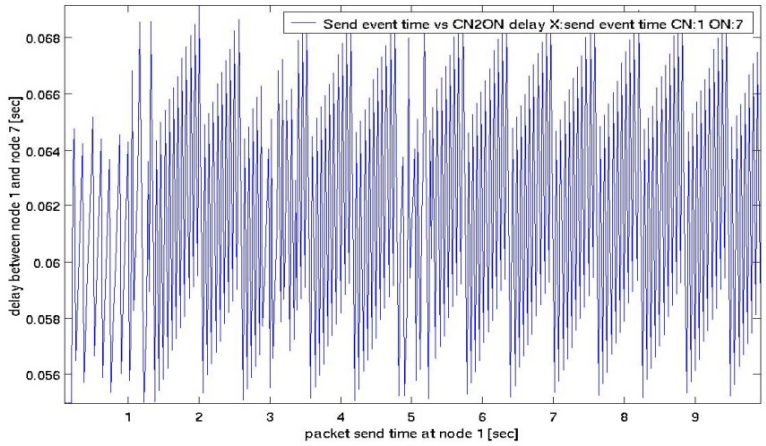


Ritardo VoIP

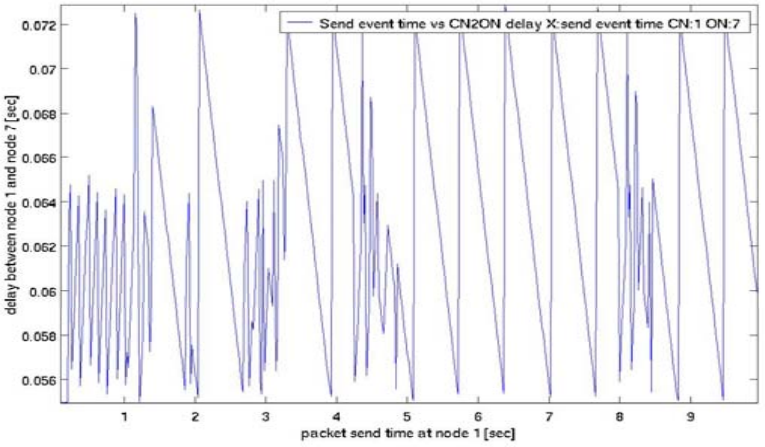
DropTail



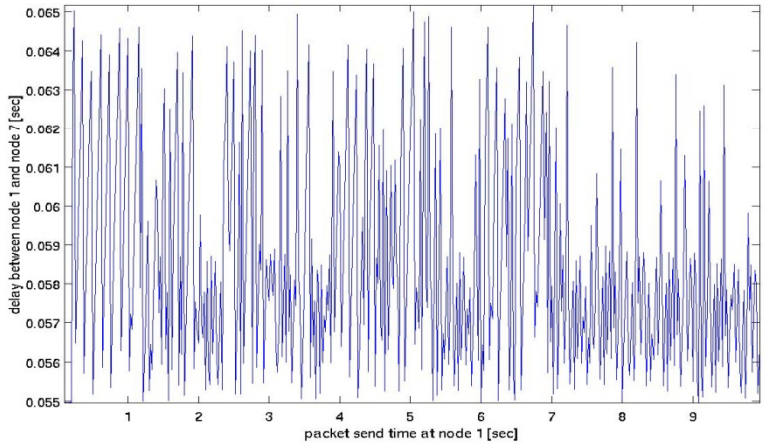
RoundRobin



WeightedRoundRobin



Priority



GRAZIE PER L'ATTENZIONE