

ENSC 835: HIGH-PERFORMANCE NETWORKS

CMPT 885: SPECIAL TOPICS: HIGH-PERFORMANCE NETWORKS

Scalability and Robustness of the Gnutella Protocol

Final project presentation

Spring 2006

Eman Elghoneimy
www.sfu.ca/~eelghone
eelghone@sfu.ca

Roadmap

- Peer-to-peer networks and protocols
- Gnutella Protocol
- Implementation
- Simulations and results
- Future work and conclusions

Peer-to-peer networks

- ❑ The user is the client and the server at the same time.
- ❑ These decentralized systems are used for file sharing applications over the internet. Each user uploads and downloads files to and from the network at the same time with no centralized control.
- ❑ “The network is the computer”, p2p nodes relate to each other side-by-side within a global computing arena [3].

P2p protocols

- ❑ The original **Napster** protocol is now non-operational. It featured a centralized directory of users' files, which acted as a single point of failure in the system and caused a copyright violation problem.
- ❑ **KazAa** features group leader nodes which authorize users – also shut down.
- ❑ **Gnutella** protocol [1],[2] is a fully distributed decentralized network. Gnutella 'clients' can be purchased from the bearshare website [4].
- ❑ Other p2p protocols are, **Kademlia** [5] protocol which is used for the free 'client' e-mule [6], and **BitTorrent** [7].

Gnutella Protocol

- ❑ Gnutella is a p2p protocol for distributed search.
- ❑ Each node in the network is called a “**servent**” and acts as a client and server in the same time.
- ❑ A servent connects to the network by establishing a connection to another node currently on the network.
- ❑ HTTP protocol is used to download files from one servent to another directly (not through the Gnutella network).

Descriptors [1]

- Descriptors are used for communicating data between servants.

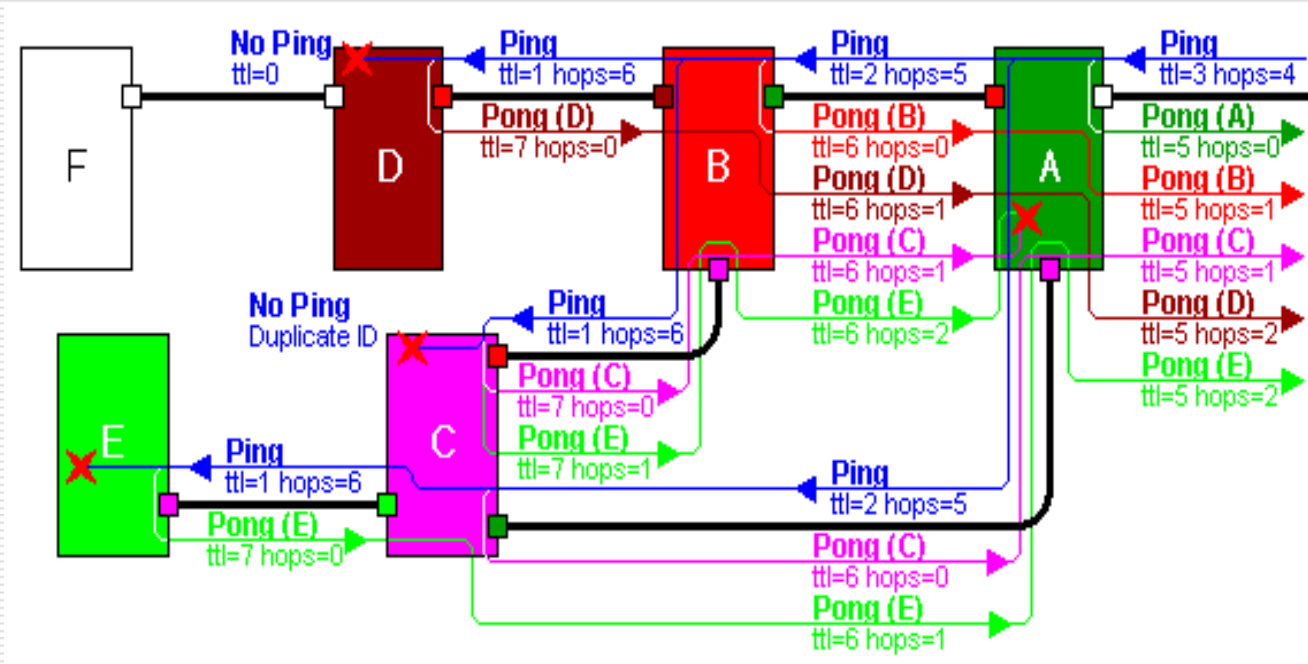
Ping	Used to actively discover hosts on the network. A servant receiving a Ping descriptor is expected to respond with one or more Pong descriptors.
Pong	The response to a Ping. Includes the address of a connected Gnutella servant and information regarding the amount of data it is making available to the network.
Query	The primary mechanism for searching the distributed network. A servant receiving a Query descriptor will respond with a QueryHit if a match is found against its local data set.
QueryHits	The response to a Query. This descriptor provides the recipient with enough information to acquire the data matching the corresponding Query.
Push	A mechanism that allows a firewalled servant to contribute file-based data to the network.

Descriptor format & Pong payload [1]

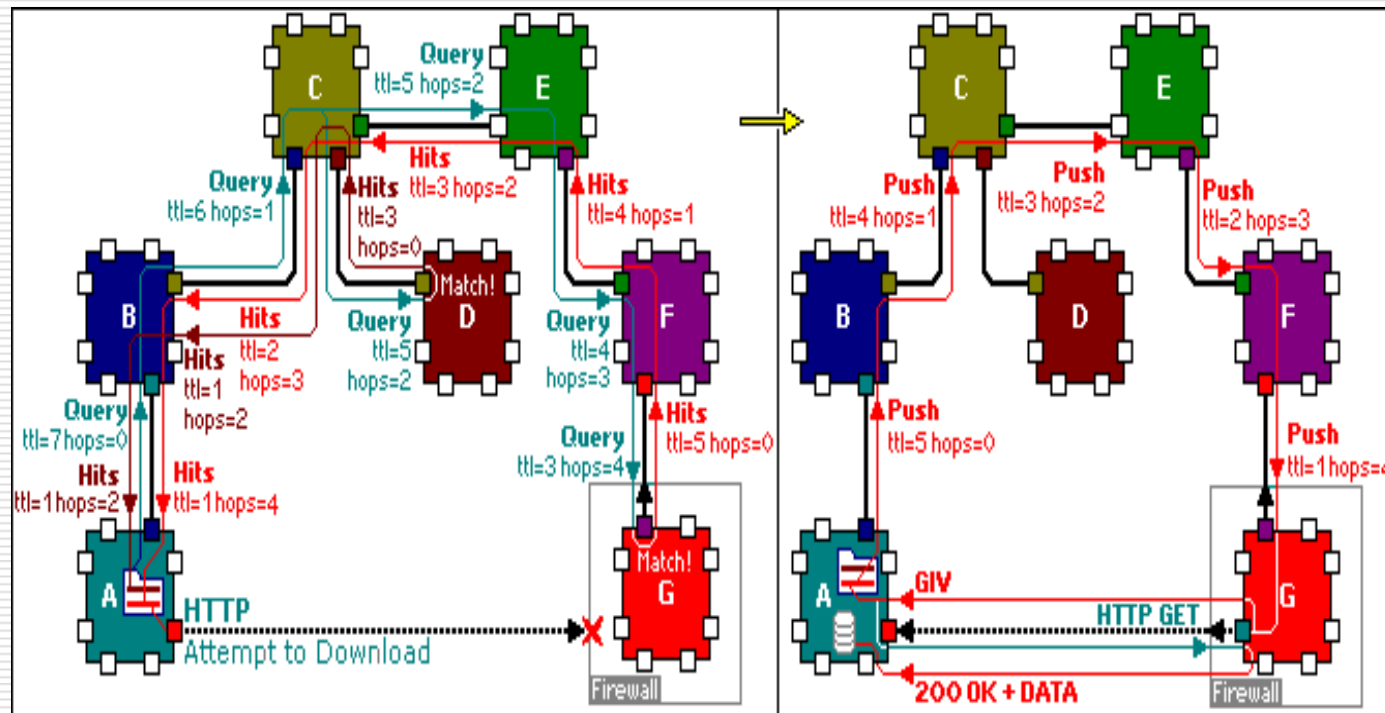
Fields	Descriptor ID	Payload Descriptor	TTL	Hops	Payload Length
Byte offset	0...15	16	17	18	19...22

Fields	Port	IP Address	Number of Files Shared	Number of Kilobytes Shared	<i>Optional Pong Data</i>
Byte offset	0...1	2...5	6...9	10...13	<i>14...L-1</i>

Ping-Pong message routing [1]



Query-QueryHit-Push messages [1]



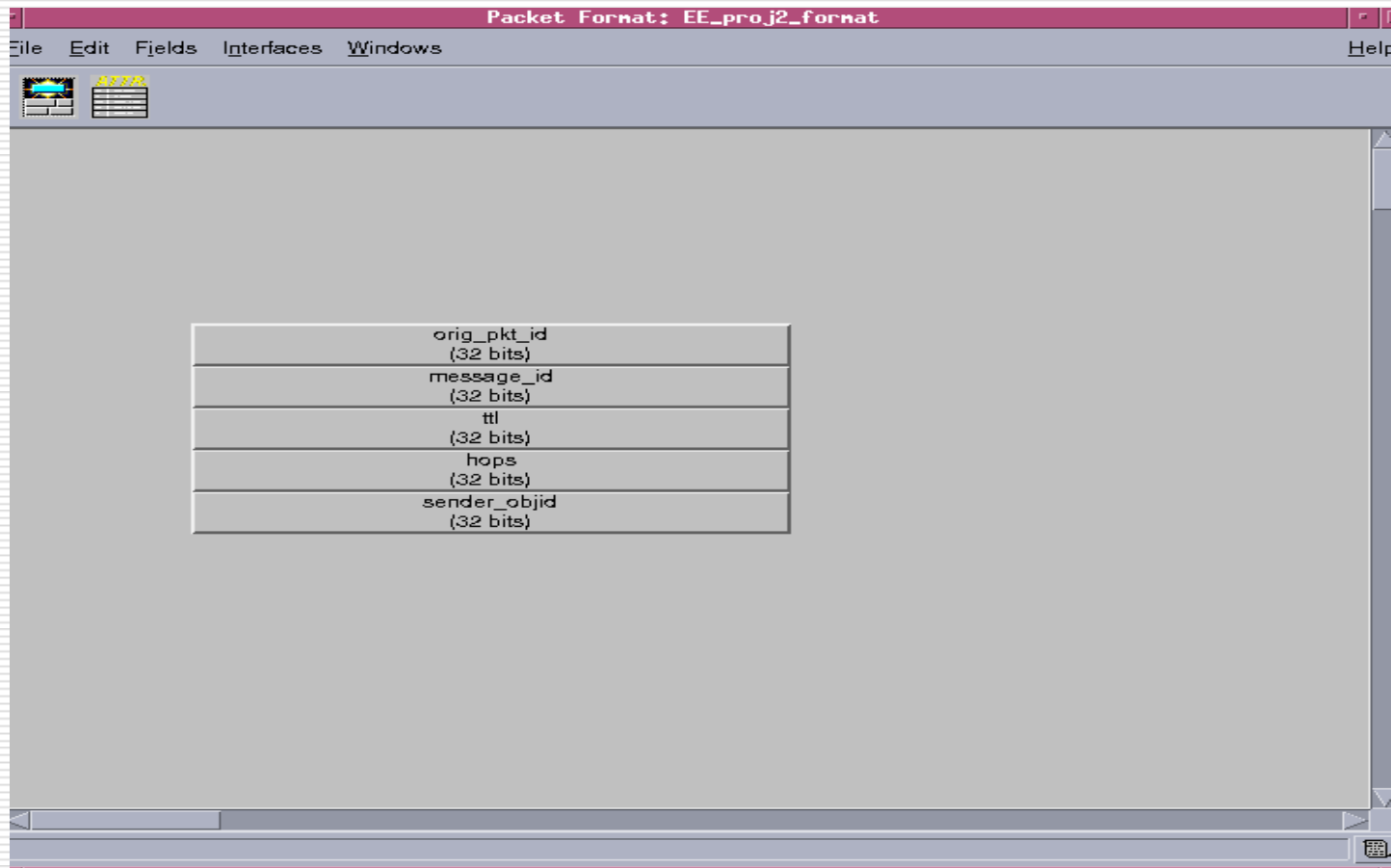
Gnutella v0.6

- ❑ Current stable version is v0.4. In Gnutella v0.4, servers with slow internet connection had problems in the Gnutella net due to the volume of messages nodes were required to forward to other nodes. More structure is added to the network in Gnutella v0.6 [2]. Nodes are of two types: **ultrapeer** and **leaf**.
- ❑ Ultrapeers are connected to each other. Each ultrapeer forwards to its leaf node only the queries that it can handle.
- ❑ A leaf node can turn into an ultrapeer dynamically if it satisfies several conditions such as bandwidth, uptime and operating system. There must be a need for an ultrapeer.
- ❑ A handshake protocol is used when a node wants to join the network as well.
- ❑ Pong messages are cached at the server. There is a maximum number of pong messages that each server can send.
- ❑ Descriptors are the same as those of v0.4.

Implementation - OPNET

- ❑ Gnutella ping-pong message routing protocol
- ❑ Using OPNET network modeling software package, Education version 11.0 under UNIX.

Packet format



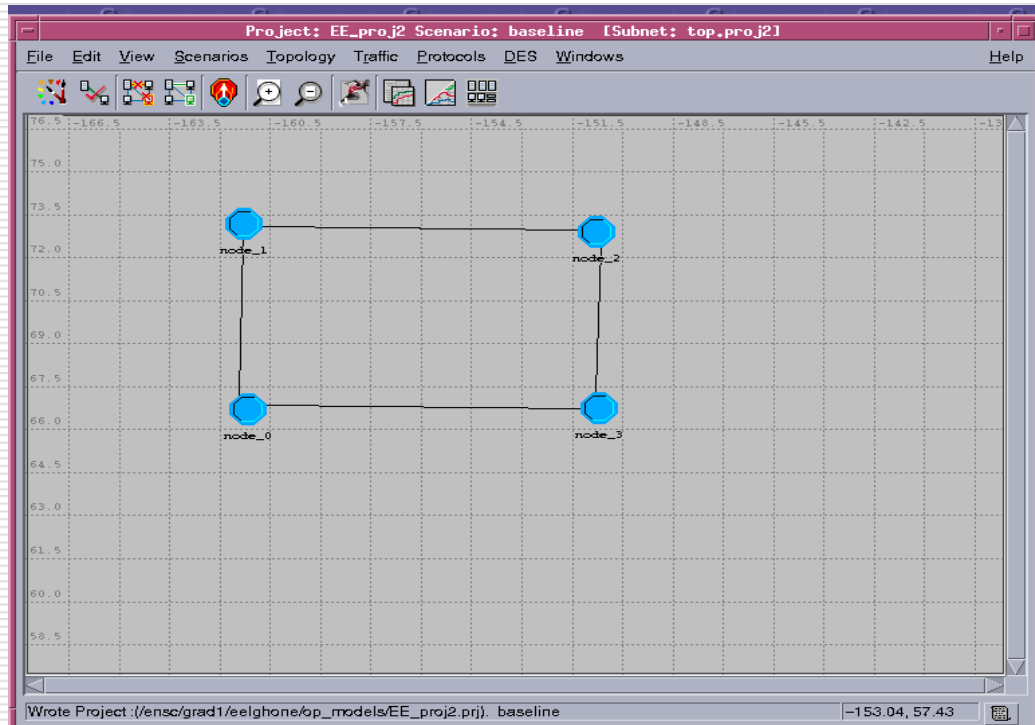
The screenshot shows a window titled "Packet Format: EE_proj2_format" with a menu bar containing "File", "Edit", "Fields", "Interfaces", "Windows", and "Help". The main area displays a table of fields:

orig_pkt_id (32 bits)
message_id (32 bits)
tfl (32 bits)
hops (32 bits)
sender_objid (32 bits)

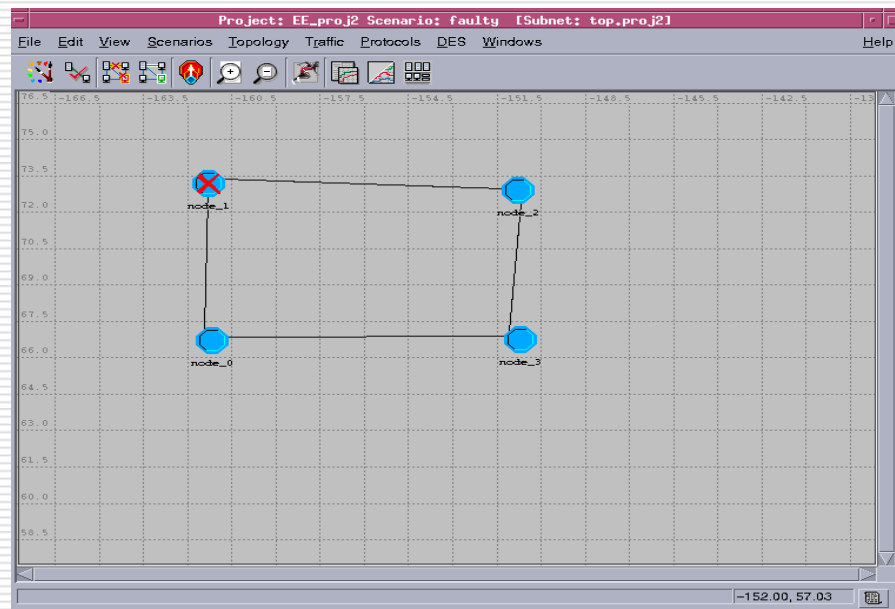
Link format

- Duplex link
- Support customized packet only

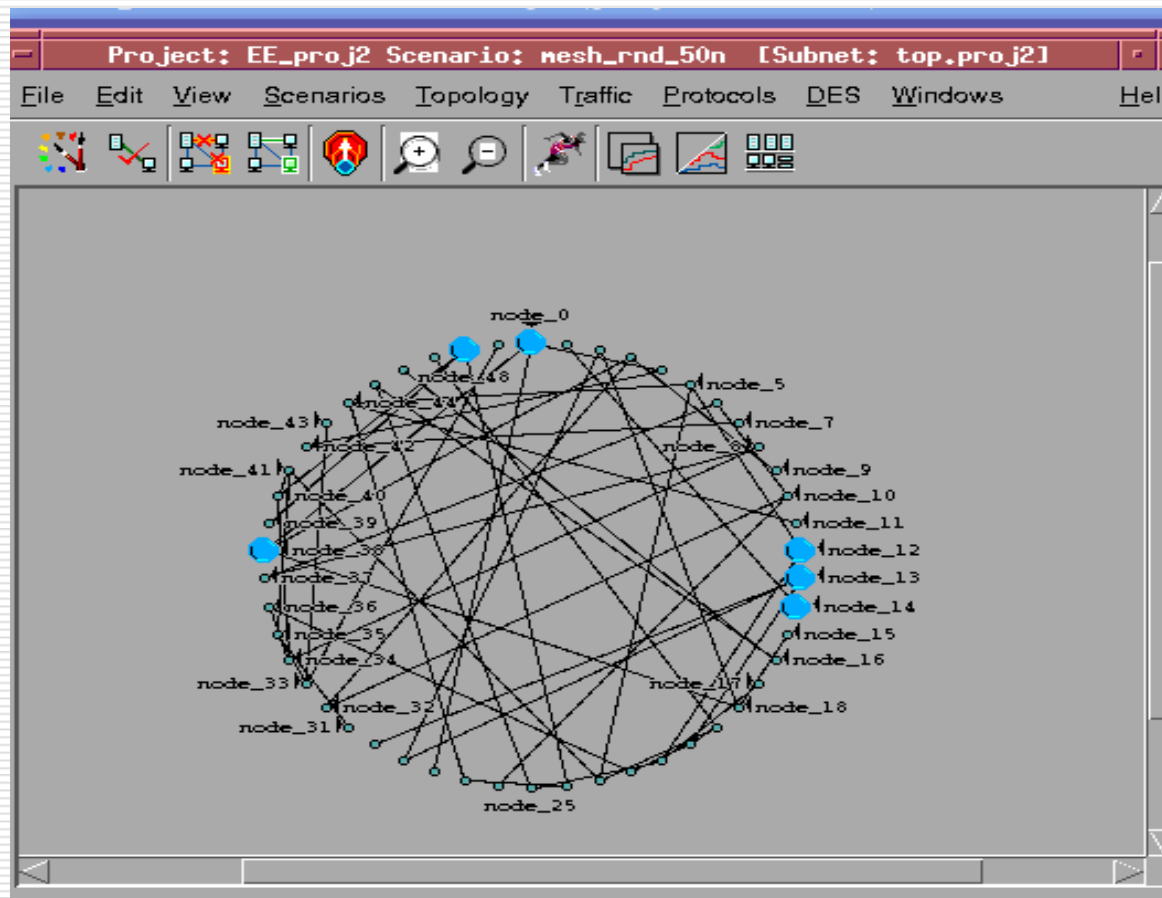
Simple network (ring_4n)



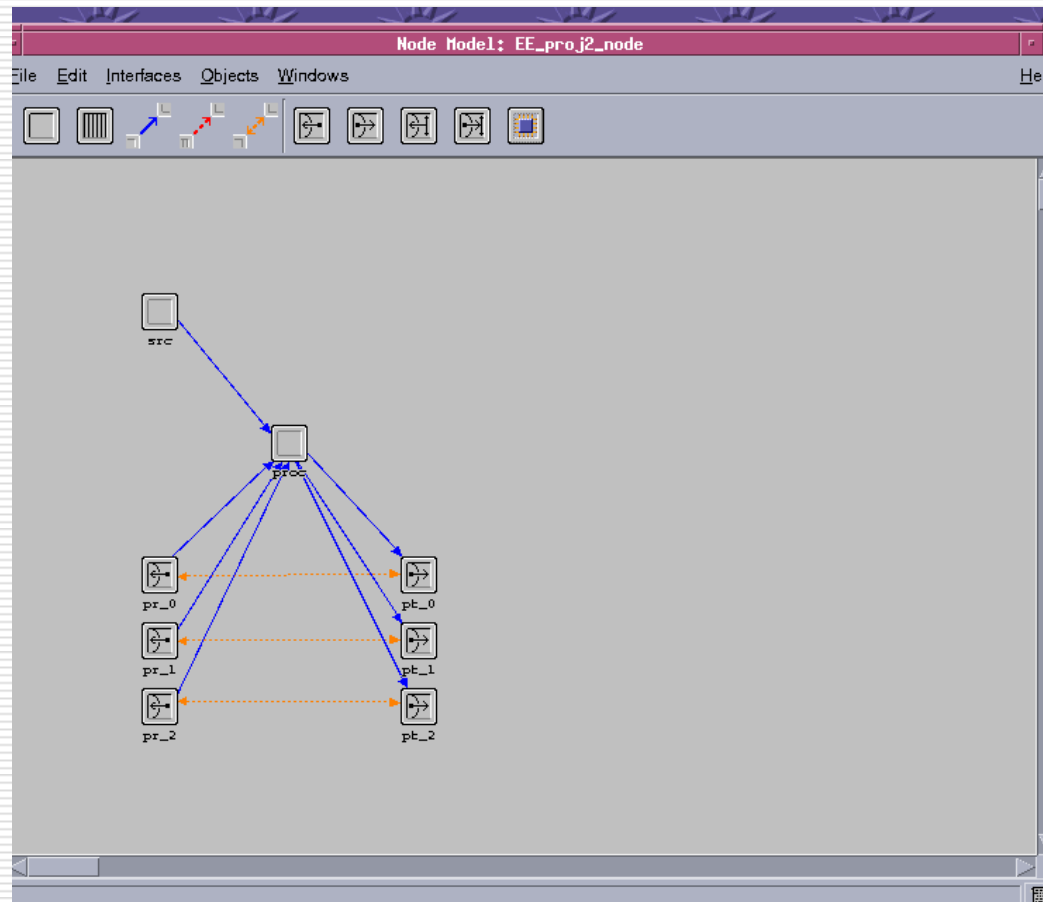
Faulty network (faulty_ring_4n)



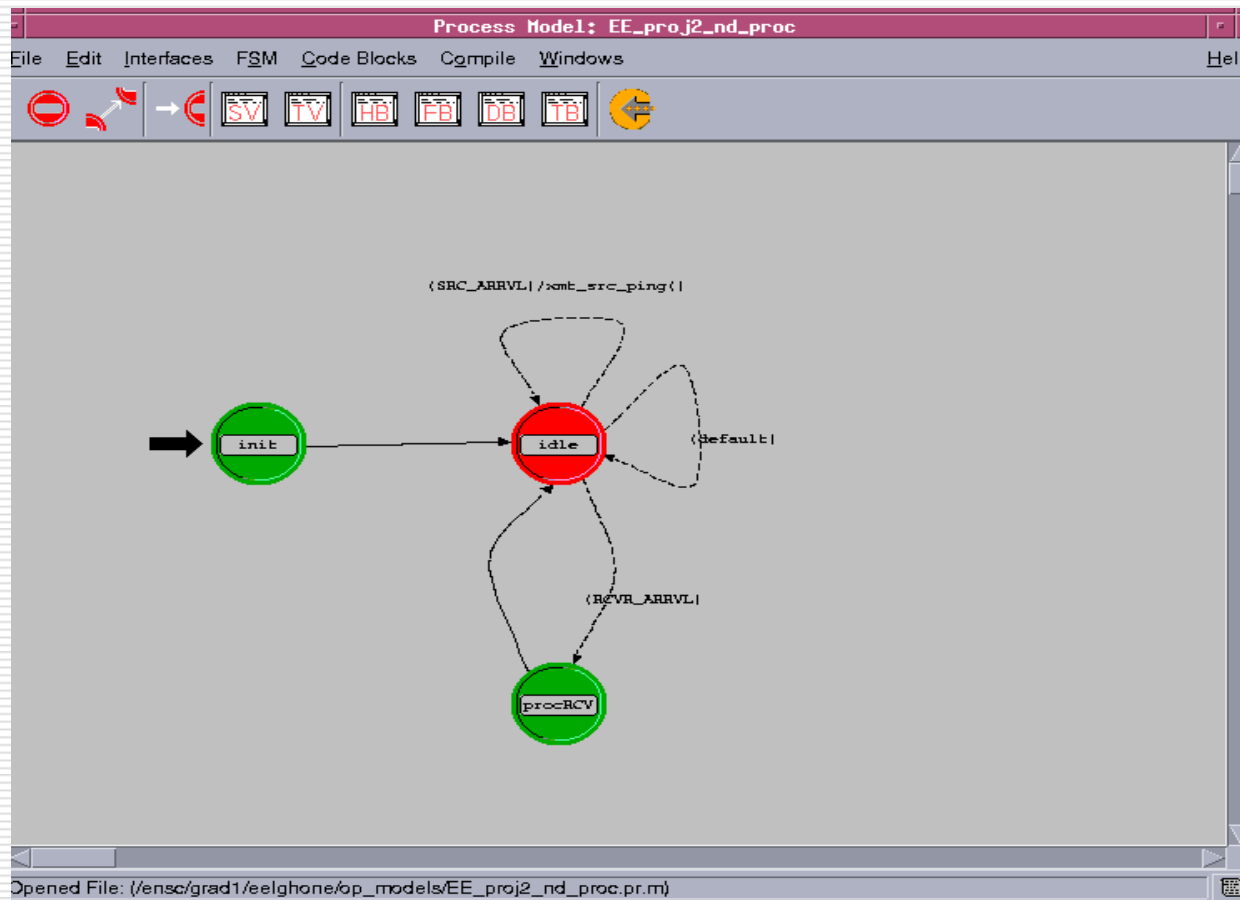
Larger network (mesh_rnd_50n)



Node model



Node processor



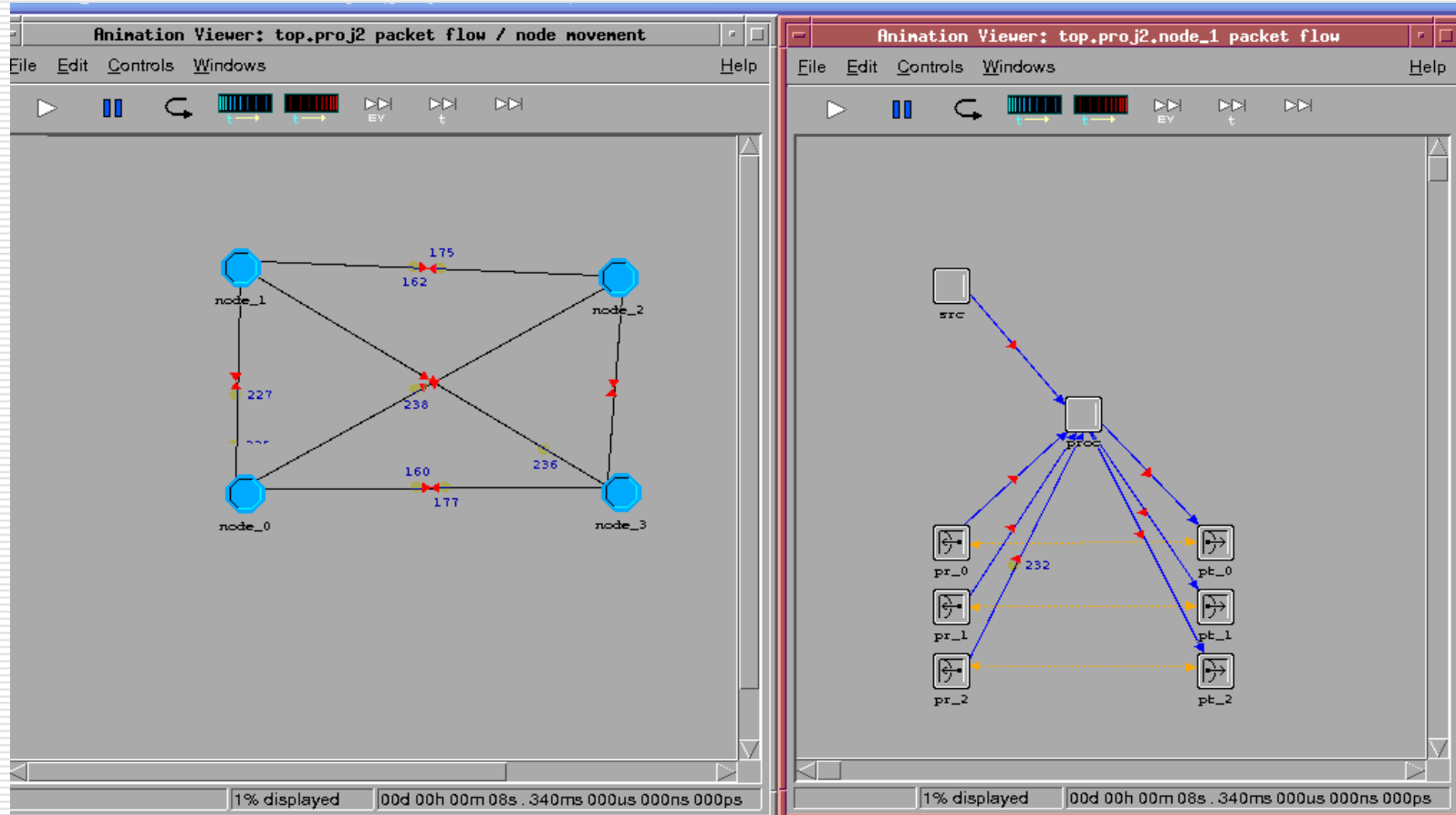
ProcRCV state

- the received packet is read, if the packet is a **ping** message, the following steps are followed:
 1. Reply with a pong message, set $ttl = received_hops + 1$
 2. If this packet is a duplicate packet, discard the packet and exit.
 3. Save the packet to the cache.
 4. If $ttl > 1$, forward the message to other peers, set $ttl = TTL_INIT$
- For **pong** messages, the same steps are followed, except for step 1.

Simulation: Verification

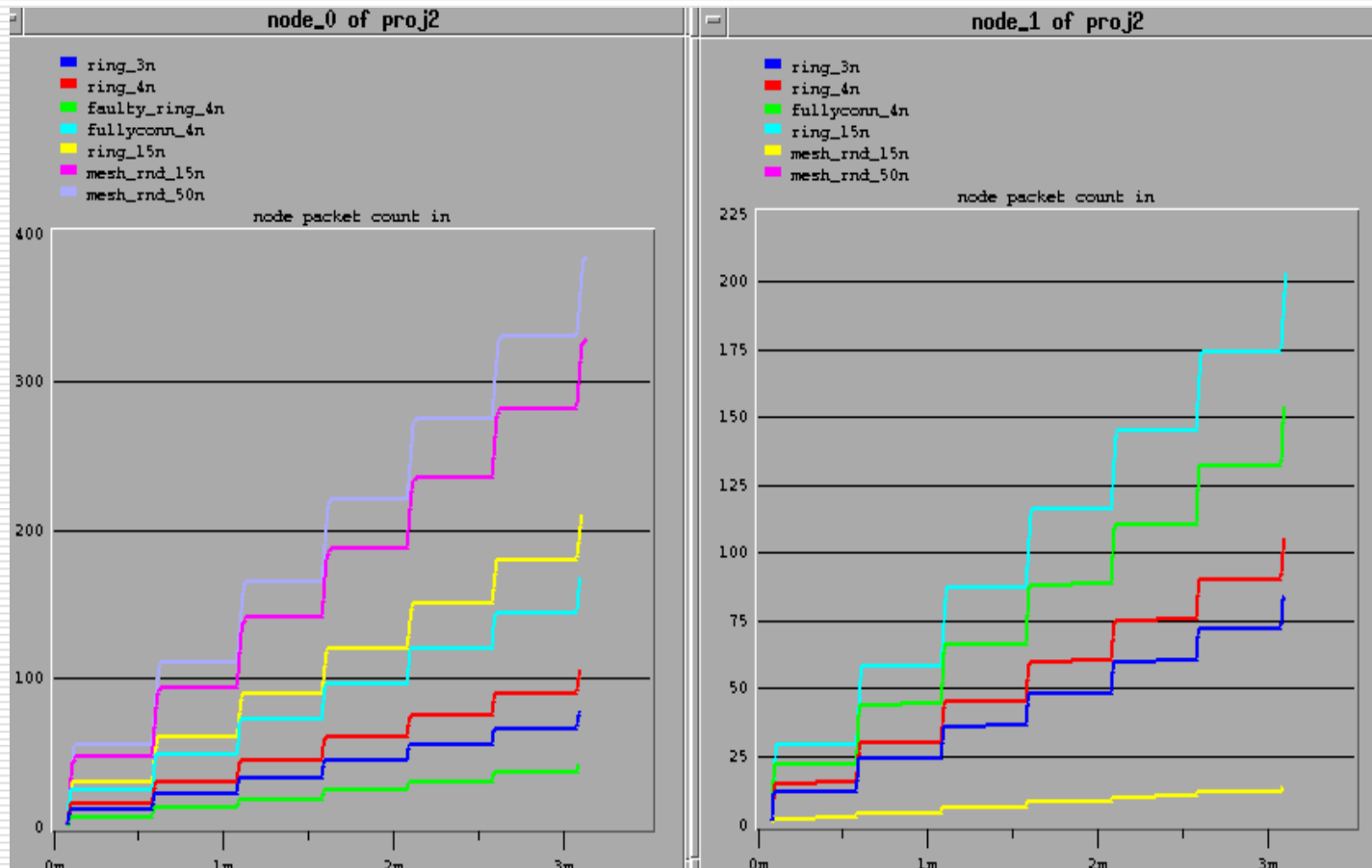
```
node_1 rcv:pk_in 8,pk_out 12
  ...received from rcvr 0: msg_id 1, pkt_id 10, orig_pkt_id 3, ttl 4, hops 1, sender node_3
  ...gen pong to xmtr 0: pkt_id 46, orig_pkt_id 10, ttl 2, hops 0, sender node_1
    Wrote ping to ping cache, ping_ptr=4
  ...fwd ping to xmtr 1: orig_pkt_id 3, ttl 3, hops 2, sender node_3
  ...gen ping to xmtr 2: orig_pkt_id 3, ttl 3, hops 2, sender node_3
node_2 rcv:pk_in 8,pk_out 15
  ...received from rcvr 2: msg_id 1, pkt_id 27, orig_pkt_id 3, ttl 4, hops 1, sender node_3
  ...gen pong to xmtr 2: pkt_id 48, orig_pkt_id 27, ttl 2, hops 0, sender node_2
  ....destroy duplicate ping packet
node_3 rcv:pk_in 8,pk_out 16
  ...received from rcvr 0: msg_id 2, pkt_id 24, orig_pkt_id 3, ttl 1, hops 0, sender node_2
    Wrote pong to pong cache, pong_ptr=1
node_0 rcv:pk_in 8,pk_out 14
  ...received from rcvr 2: msg_id 1, pkt_id 25, orig_pkt_id 3, ttl 4, hops 1, sender node_3
  ...gen pong to xmtr 2: pkt_id 49, orig_pkt_id 25, ttl 2, hops 0, sender node_0
  ....destroy duplicate ping packet
```

Packet flow animation



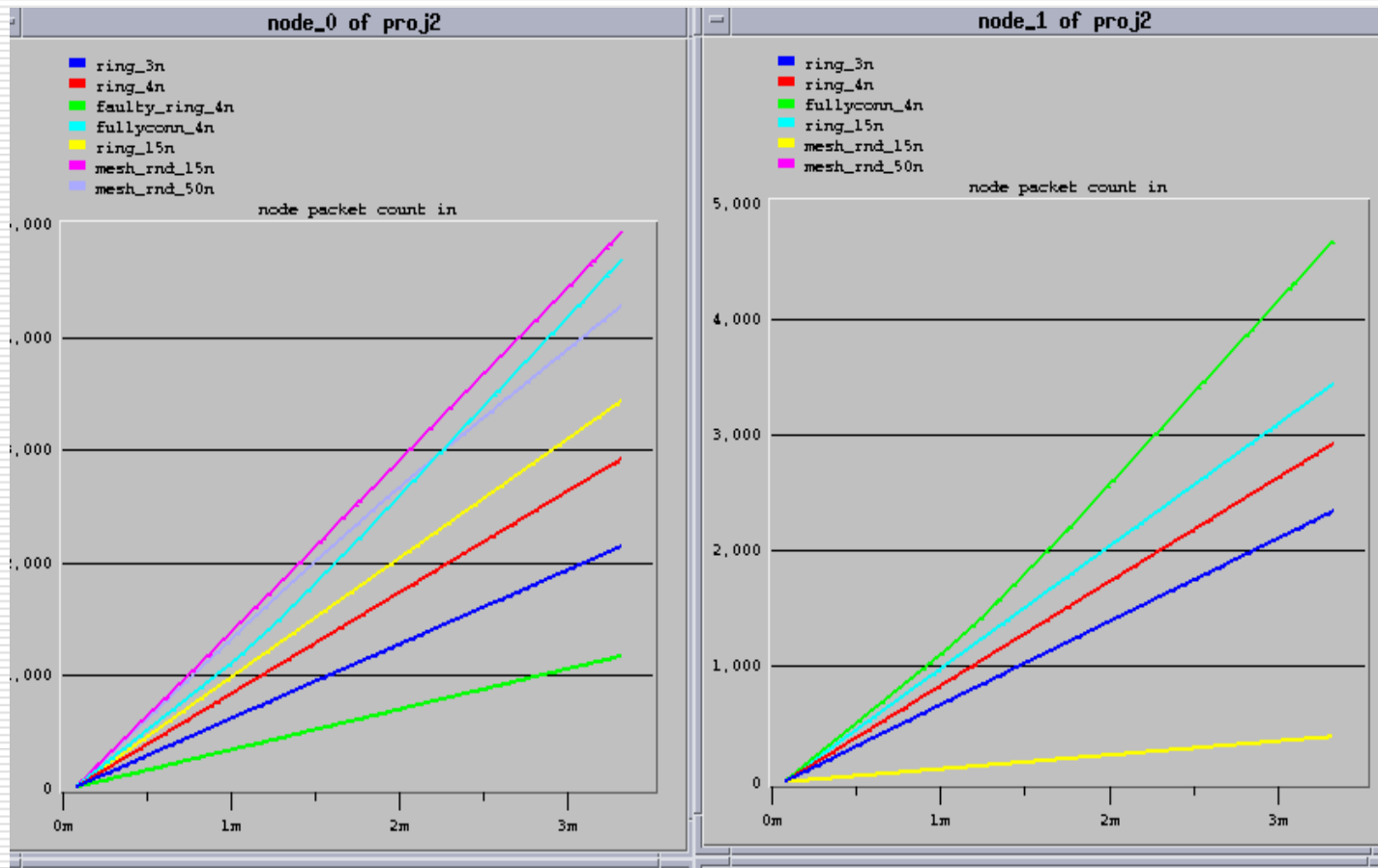
Results

Incoming packets, ping inter-arrival time = 30 sec

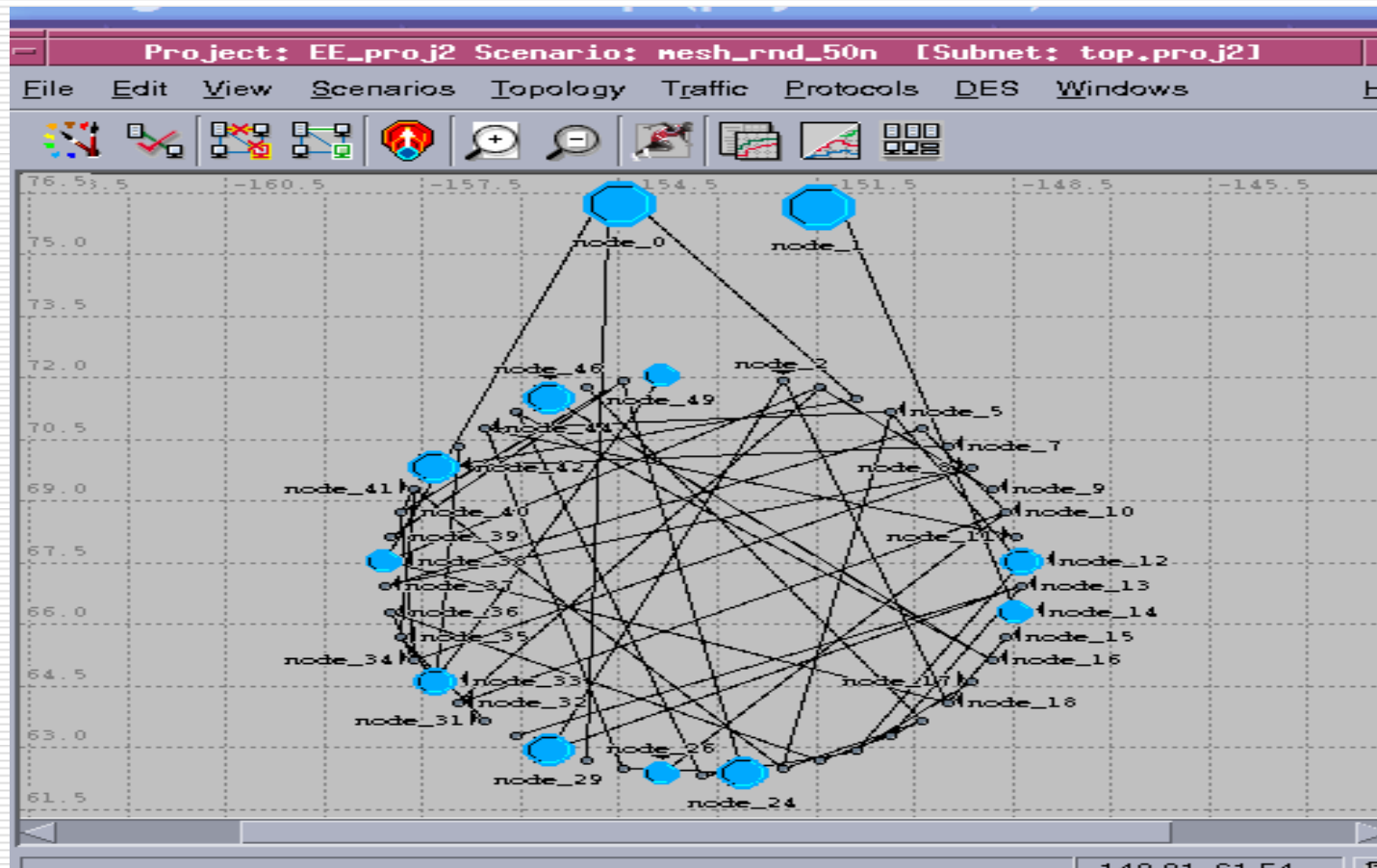


Results

Incoming packets, ping inter-arrival time = 1 sec

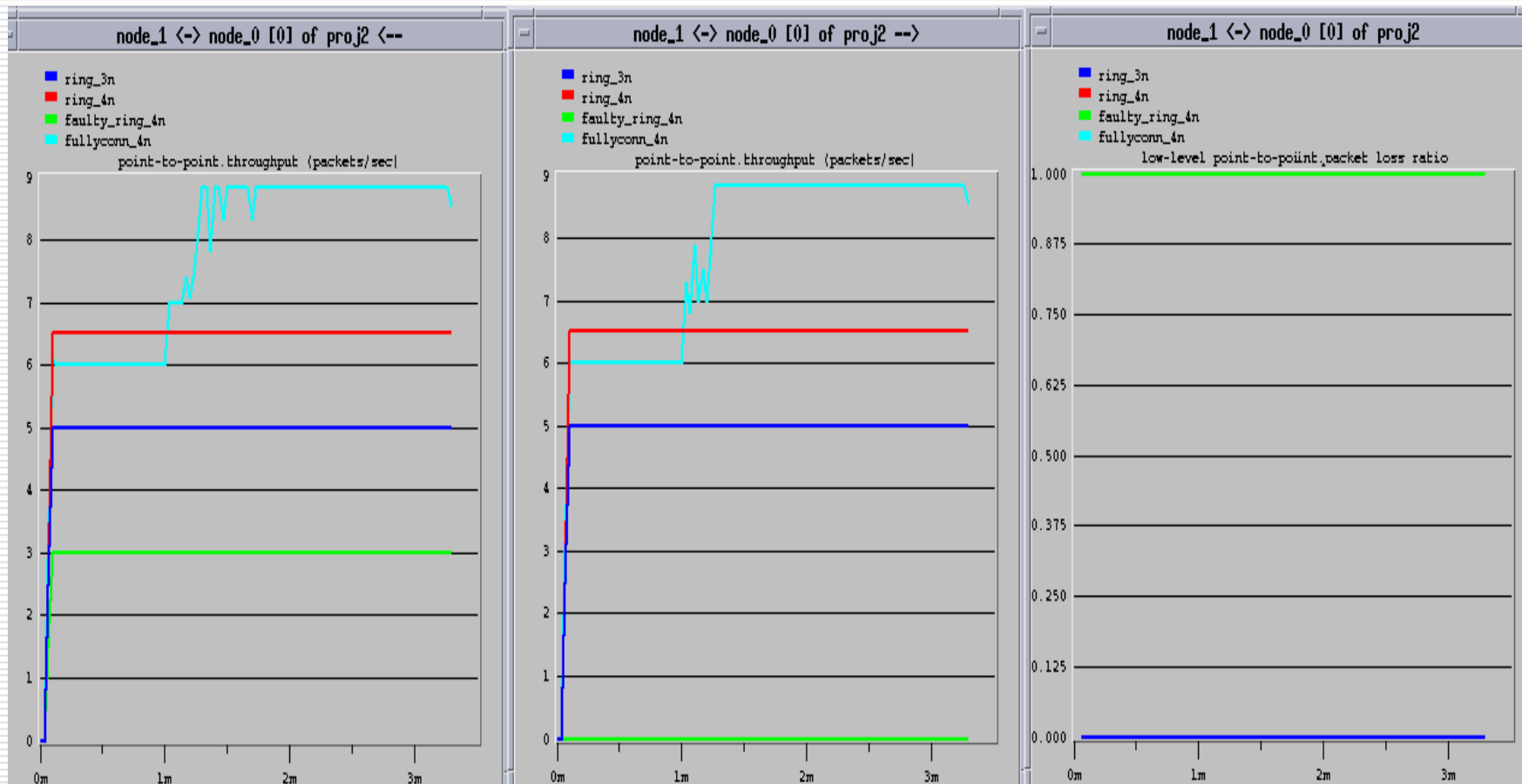


Node connectivity



Link statistics

(ping inter-arrival time = 1 sec)



Discussion

- ❑ Large networks do not always carry higher number of packets than small networks. **Connectivity** of the nodes plays an important role in the amount of message exchange.
- ❑ Networks with **faulty** nodes still function by exchanging messages throughout the network. However, the number of packets is less than that of similar networks with no faulty nodes.
- ❑ Ping generation **inter-arrival time** have a big effect on the number of packets in the network, which is expected.

Challenges

- ❑ Simulating a logical network
- ❑ Simulating the dynamic nature of the network.
- ❑ In OPNET, there is no main program that can be used to dynamically set the network structure or node behaviour.
 - *While this poses a difficulty in modeling, it actually represents the true nature of decentralized systems. In such systems, all the functionality of the system lies in the individual nodes rather than one centralized process.*

Future work

- ❑ Perform more analysis to the results by examining more node results, larger scenarios and other faulty networks.
- ❑ Model the rest of the protocol such as Query and QueryHit.
- ❑ Model Gnutella v0.6 ultrapeer-leaf architecture
- ❑ Examine using TCP OPNET model for lower layer implementation.
- ❑ Investigate other simulators that can simulate dynamic topologies.

Summary

- ❑ Gnutella is a decentralized peer-to-peer (p2p) network protocol for file sharing.
- ❑ There are two major advantages of decentralized protocols. The first one is scalability, and the second one is robustness to failure.
- ❑ In this project, OPNET simulator is used to model the Gnutella protocol.
- ❑ Scalability is tested by adding a large number of nodes. Connectivity of nodes is as important as the size of the network.
- ❑ Robustness to failure is tested by arbitrary failing some nodes. Less traffic is generated, but the network is still connected.

References

1. "The Annotated Gnutella Protocol Specification v0.4", <http://rfc-gnutella.sourceforge.net/developer/stable/index.html>, accessed April 2006.
2. "Gnutella Protocol Development – RFC-Gnutella 0.6", <http://rfc-gnutella.sourceforge.net/developer/testing/index.html>, accessed April 2006.
3. L. Gong, "[Peer-to-Peer Networks in Action](#)," *IEEE Internet Computing*, vol. 5, no. 1, 2002, pp. 37-39.
4. www.bearshare.com, accessed April 2006.
5. P. Maymounkov and D. Mazieres, "[Kademlia: A Peer-to-peer Information System Based on the XOR Metric](#)," *Lecture Notes in Computer Science*, vol. 2429, revised papers from the First International Workshop on Peer-to-peer Systems, 2002, pp. 53-65.
6. www.emule-project.net, accessed April 2006.
7. www.wikipedia.org/wiki/BitTorrent, accessed April 2006.