Routing Simulator Introduction

Routing

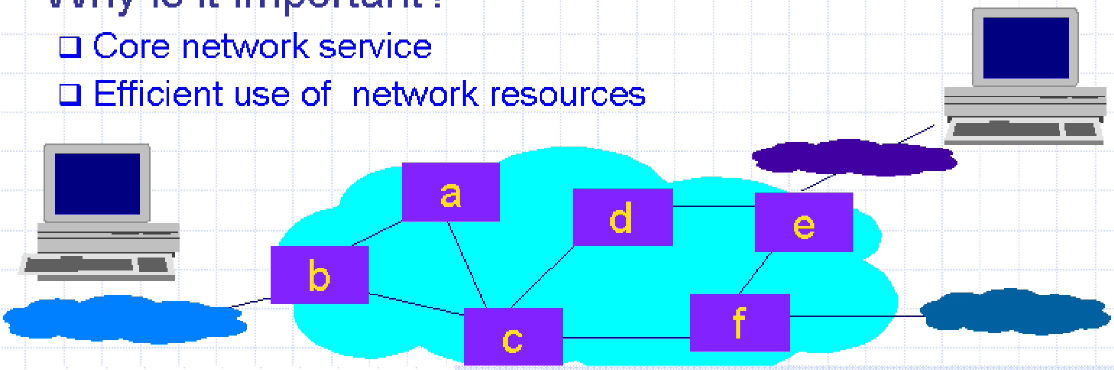
A simple definition of routing is "learning how to get from here to there." In some cases, the term *routing* is used in a very strict sense to refer *only* to the process of obtaining and distributing information, but not to the process of using that information to actually get from one place to. Since it is difficult to grasp the usefulness of information that is acquired but never used, we employ the term *routing* to refer in general to all the things that are done to discover and advertise paths from here to there and to actually move packets from here to there when necessary. The distinction between routing and forwarding is preserved in the formal discussion of the functions performed by OSI end systems and intermediate systems, in which context the distinction is meaningful.

Routing is the act of moving information across an inter network from a source to a destination. Along the way, at least one intermediate node typically is encountered. **Routing** is the process of finding a path from a source to every destination in the network. It allows users in the remote part of the world to get to information and services provided by computers anywhere in the world. Routing is accomplished by means of routing protocols that establish mutually consistent routing tables in every router in the Network.

When a packet is received by the router or is forwarded by the host, they both must make decisions as to how to send the packet. To do this, the router and the host consult a database for information known as the routing table. This database is stored in RAM so that the lookup process is optimized. As the packet is forwarded through various routers towards its destination, each router makes a decision so as to proceed by consulting its routing table.

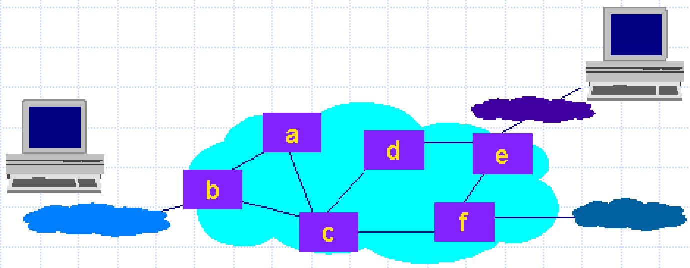
Introduction to Routing

- Goal: efficient end-to-end packet delivery
- Key Functions:
  - Translation: address => route
  - Coordinated distributed decisions: which outgoing link?
  - Global performance objectives: e.g., minimize delay
  - Other: priority, security, domain-policies
- Why is it important?
  - Core network service
  - Efficient use of network resources



Why Is It Difficult?

- Problem 1: scalable address =>route computation
  - Requires knowledge of global topology & coordinated distributed decisions
- Problem 2: handling connectivity changes
  - How to propagate updates and minimize temporal inconsistencies
- Problem 3: efficient utilization of network resources
  - E.g., minimize congestion, delay
- Problem 4: scalability for large multi-domain networks
- Problem 5: support flexible routing policies
- Problem 6: support applications QoS



Routing Table

A routing table consists at least two columns: the first is address of a destination point or destination Network , and the second is the address of the next element that is the next hop in the "best" path to its destination. When a packet arrives at a router the router or the switch controller consults the routing table to decide the next hop for the packet. Not only the local information but the global information is also consulted for routing. But global information is hard to collect, subject to frequent changes and is voluminous.

The information in the routing table can be generated in one of two ways. The first method is to manually configure the routing table with routes for each destination network. This is known as static routing. The second method for generating routing table information is to make use of dynamic routing protocol. A dynamic routing protocol consists of routing tables that are built and maintained automatically through and ongoing communication between routers. Periodically or on demand, messages are exchanged between routers for the purpose of updating information kept in their routing tables.

Router

To reach network:	Send to:
27	Node A
57	Node B
17	Node C
24	Node A
52	Node A
16	Node B
26	Node A
. . .	. . .

Routing Table

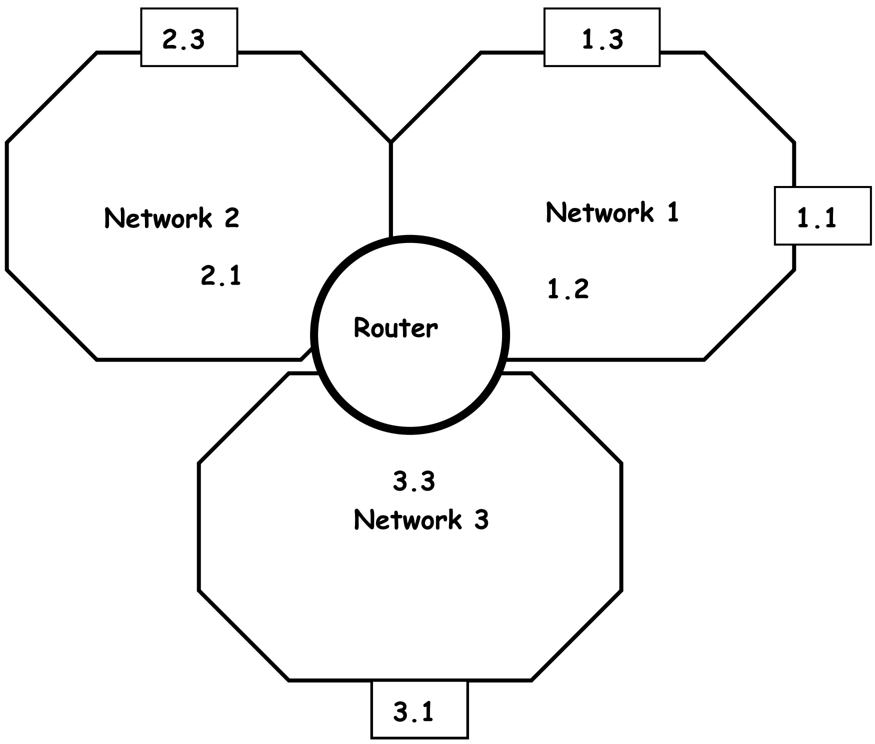
S1283a

The Network forwards IP packets from a source to a destination using destination address field in the packet header. A router is defined as a host that has an interface on more than one Network.

Every router along the path has routing table with at least two fields:

A Network number and the interface on which to send packets with that network number.

The router reads the destination address from an incoming packet's header and uses the routing table to forward it to appropriate interface. By introducing routers with interfaces on more than one cluster, we can connect clusters into larger ones. By induction we can compose arbitrarily large networks in this fashion, as long as there are routers with interfaces on each subcomponent of the Network.



A router which has interface on more than one Network

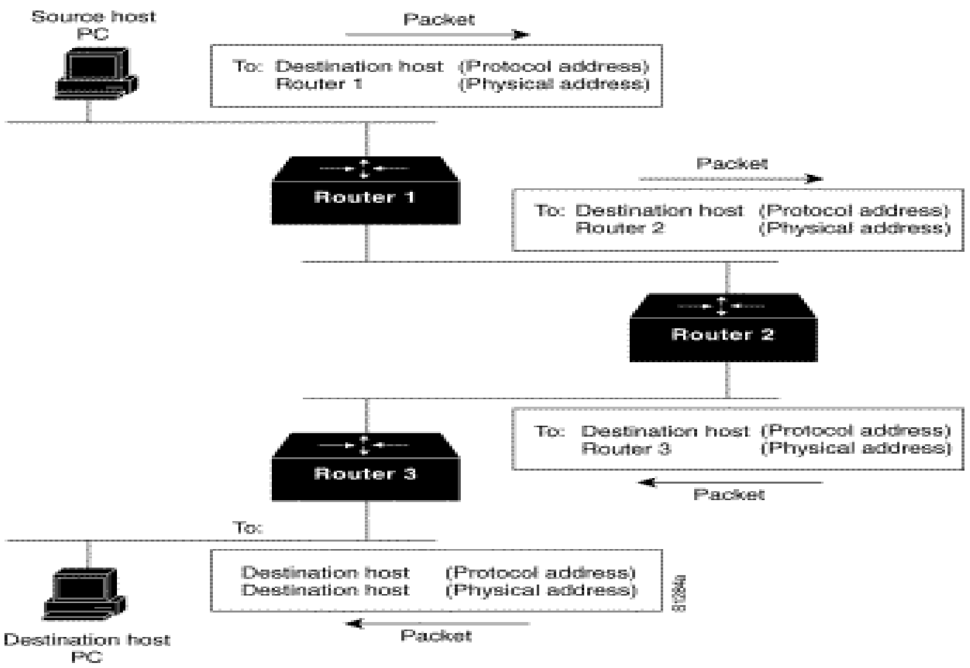
Packet

The Network carries all the information using packets.

A packet has two parts:

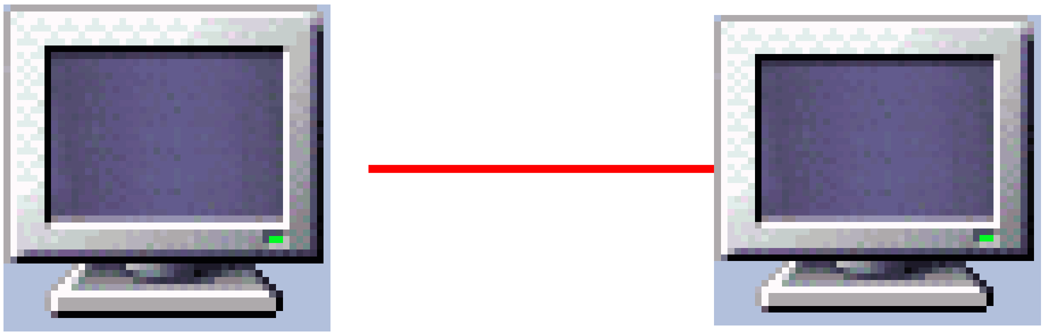
The information content called the payload, and the information about the payload, called the meta-data.

The meta-data consists of fields such as the source ands destination addresses, data length, sequence number and data type. The introduction of meta-data is a fundamental innovation in networking technology. The Network cannot determine where samples originate, or where they are going without additional context information. Meta-data makes information self-descriptive, allowing the network to interpret the data without additional context information. In particular if the meta-data contains a source and destination address, no matter where in the network the packet is, the Network knows where it came from and where it wants to go. The Network can store a packet, for hours if necessary, then "freeze" it and still know what has to be done to deliver the data. Packets are efficient for data transfer, but are not so attractive for real-time services such as voice.



Link

Link is the connection between two routers. If there are two routers the messages are sent from one to other using the link. So link acts as a bridge between two routers. If a link goes down then information will not be transferred to the routers. We have to search for the other alternative links to reach from source to destination. Hence link plays a major role in the transmission of data as it acts as a carrier of the messages sent by the routers.



Link between two routers

Routing Algorithm

Routing is accomplished by means of routing protocols that establish mutually consistent routing tables in every router in the Network. A routing protocol written in the form of code is routing algorithm. A routing algorithm asynchronously updates routing tables at every router or switch controller. The global information to be maintained by routing tables is voluminous. Routing algorithm summarizes this information to extract only the portions relevant to each node. The heart of routing algorithm does all the chores.

The various concepts for discussion are:

- Design goals of Routing Algorithm
- Factors that decide the best Path
- Choices in Routing

Routing algorithms often have one or more of the following design goals:

- **Optimality**

Optimality refers to the capability of the routing algorithm to select the best route, which depends on the metrics and metric weightings used to make the calculation. One routing algorithm, for example, may use a number of hops and delays, but may weight delay more heavily in the calculation. Naturally, routing protocols must define their metric calculation algorithms strictly.

- **Simplicity and low overhead**

Routing algorithms also are designed to be as simple as possible with a minimum of software and Utilization overhead. In other words, the routing algorithm must offer its functionality efficiently, with a minimum of software and utilization overhead. Efficiency is particularly important when the software implementing the routing algorithm must run on a computer with limited physical resources.

- **Robustness and stability**

Routing algorithms must be robust, which means that they should perform correctly in the face of unusual or unforeseen circumstances, such as hardware failures, high load conditions, and incorrect implementations. Because routers are located at network junction points, they can cause considerable problems when they fail. The best routing algorithms are often those that have withstood the test of time and have proven stable under a variety of network conditions.

- **Rapid convergence**

Routing algorithms must converge rapidly. Convergence is the process of agreement, by all routers, on optimal routes. When a network event causes routes either to go down or become available, routers distribute routing update messages that permeate networks, stimulating recalculation of optimal routes and eventually causing all routers to agree on these routes. Routing algorithms that converge slowly can cause routing loops or network outages.

- **Flexibility**

Routing algorithms should also be flexible, which means that they should quickly and accurately adapt to a variety of network circumstances. Routing algorithms can be programmed to adapt to changes in network bandwidth, router queue size, and network delay, among other variables.

Factors that decide the best path

Routing algorithms have used many different metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics, combining them in a single (hybrid) metric. All the following metrics have been used:

Path Length

Path length is the most common routing metric. Some routing protocols allow network administrators to assign arbitrary costs to each network link. In this case, path length is the sum of the costs associated with each link traversed. Other routing protocols define *hop count*, a metric that specifies the number of passes through internetworking products, such as routers, that a packet must take en route from a source to a destination.

Reliability

Reliability, in the context of routing algorithms, refers to the dependability (usually described in terms of the bit-error rate) of each network link. Some network links might go down more often than others. After a network fails, certain network links might be repaired more easily or more quickly than other links. Any reliability factors can be taken into account in the assignment of the reliability ratings, which are arbitrary numeric values usually assigned to network links by network administrators.

Delay

Routing delay refers to the length of time required to move a packet from source to destination through the internetwork. Delay depends on many factors, including the bandwidth of intermediate network links, the port queues at each router along the way, network congestion on all intermediate network links, and the physical distance to be traveled. Because delay is a conglomeration of several important variables, it is a common and useful metric.

Bandwidth

Bandwidth refers to the available traffic capacity of a link. All other things being equal, a 10-Mbps Ethernet link would be preferable to a 64-kbps leased line. Although bandwidth is a rating of the maximum attainable throughput on a link, routes through links with greater bandwidth do not necessarily provide better routes than routes through slower links. If, for example, a faster link is busier, the actual time required to send a packet to the destination could be greater.

Load

Load refers to the degree to which a network resource, such as a router, is busy. Load can be calculated in a variety of ways, including CPU utilization and packets processed per second. Monitoring these parameters on a continual basis can be resource-intensive itself.

Communication Cost

Communication cost is another important metric, especially because some companies may not care about performance as much as they care about operating expenditures. Even though line delay may be longer, they will send packets over their own lines rather than through the public lines that cost money for usage time.

Choices in Routing

Routing algorithms can be classified by type. Key differentiators include:

Static versus dynamic (Non-adaptive versus Adaptive)

Non-adaptive algorithms do not base their routing decisions on measurements or estimates of the current traffic and topology. The choice of route is computed in advance, offline and downloaded to the routers when the network is booted. Adaptive algorithms in contrast change their decisions.

Single-path versus Multi-path

Some sophisticated routing protocols support multiple paths to the same destination. Unlike single-path algorithms, these multi path algorithms permit traffic multiplexing over multiple lines. The advantages of multi path algorithms are obvious: They can provide substantially better throughput and reliability.

Flat versus Hierarchical

Some routing algorithms operate in a flat space, while others use routing hierarchies. In a flat routing system, the routers are peers of all others. In a hierarchical routing system, some routers form what amounts to a routing backbone. Packets from non-backbone routers travel to the backbone routers, where they are sent through the backbone until they reach the general area of the destination. At this point, they travel from the last backbone router through one or more non-backbone routers to the final destination.

Host-intelligent versus Router-intelligent (Source Routing versus Hop by hop)

Some routing algorithms assume that the source end-node will determine the entire route. This is usually referred to as *source routing*. In source-routing systems, routers merely act as store-and-forward devices, mindlessly sending the packet to the next stop. Other algorithms assume that hosts know nothing about routes. In these algorithms, routers determine the path through the inter network based on their own calculations. In the first system, the hosts have the routing intelligence. In the latter system, routers have the routing intelligence.

Intradomain versus Interdomain

Some routing algorithms work only within domains; others work within and between domains. The nature of these two algorithm types is different. It stands to reason, therefore, that an optimal intradomain- routing algorithm would not necessarily be an optimal interdomain- routing algorithm.

Centralized versus Decentralized

In centralized routing, a central processor collects information about the status of each link and processes this information to compute a routing table for every node. It then distributes these tables to all the routers. In decentralized routing, routers must cooperate using a distributed routing protocol to create mutually consistent routing tables.

Routing Algorithms

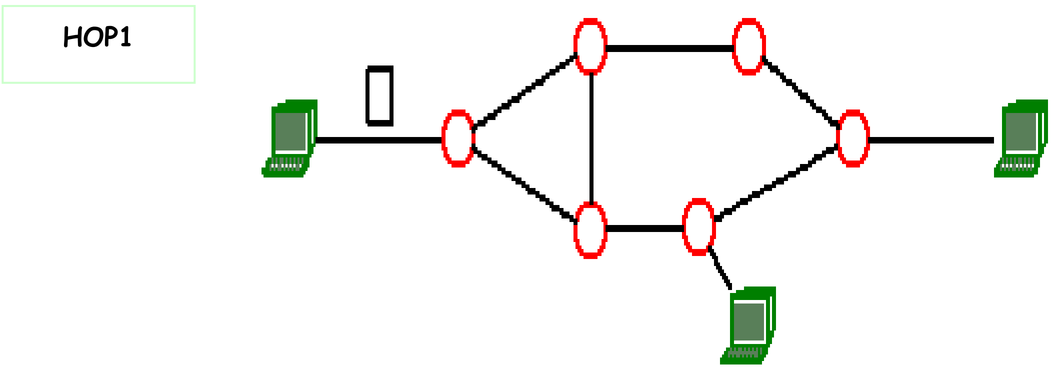
Algorithms Used

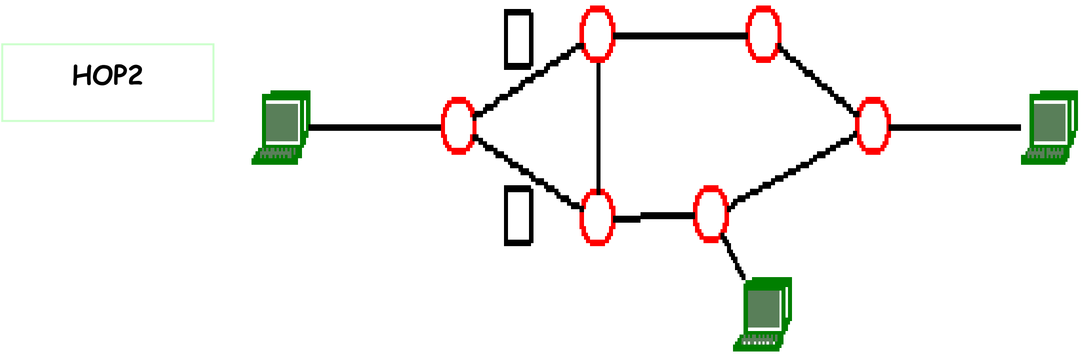
- **Flooding**
- **Hot-Potato**
- **Source Routing**
- **Distance Vector (Bellman-Ford)**
- **RIP (Routing Information Protocol)**
- **Link state**

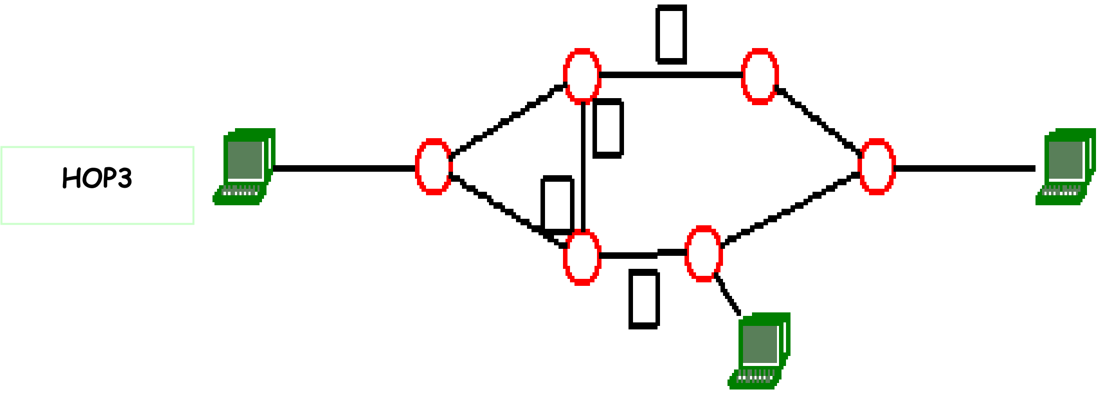
Flooding

Every incoming packet is sent out on every other link by every router. Super simple to implement, but generates lots of redundant packets. Interesting to note that all routes are discovered, including the optimal one, so this is robust and high performance (best path is found without being known ahead of time). Good when topology changes frequently (USENET example).

Some means of controlling the expansion of packets is needed. It could try to ensure that each router only floods any given packet once. Could try to be a little more selective about what is forwarded and where. The station initiating a packet stores the distance of the destination in the submitted packet (or the largest distance in the network). Each node reduces the counter by one, and resubmits the packet to all the adjacent nodes (but not to the node from where it received the packet). Packets with counter 0 are discarded. The destination node doesn't resubmit the packet .







Stages in Flooding

Advantages:

Highly robust
Suitable for setting virtual circuits
Useful for broadcasting

Disadvantage:

High traffic load

Hot - Potato Routing

Hot-potato routing, or **deflection routing**, the nodes of a network have no buffer to store packets in before they are moved on to their final predetermined destination. In normal routing situations, when multiple packets contend for a single outgoing channel, packets that are not buffered are dropped to avoid congestion. But in hot potato routing, each packet that is routed is constantly transferred until it reaches its final destination because the individual communication links can not support more than one packet at a time.

The packet is bounced around like a **"hot-potato,"** sometimes moving further away from its destination because it has to keep moving through the network. This technique allows multiple packets to reach their destinations without being dropped. This is in contrast to **"store and forward"** routing where the network allows temporary storage at intermediate locations.

This is a simple and effective way to route packets in communication networks. In these networks, nodes have no buffer to store the messages in transit, thus causing the messages to move from node to node each time. In other words, the messages are treated like **hot potatoes.**

Hot-potato routing is used for the following applications:

Real Networks

Immediate applications

Optical Networks

Hot potato routing has applications in optical networks where messages made from light can not be stored in any medium.

Non-optical networks

We obtain cheaper and easier to build networks, since nodes are simpler without buffers.

Source Routing

Source Routing is a technique whereby the sender of a packet can specify the route that a packet should take through the network. As a packet travels through the network, each router will examine the "destination IP address" and choose the next hop to forward the packet to. In source routing, the "source" (i.e. the sender) makes some or all of these decisions.

In **strict** source routing, the sender specifies the *exact* route the packet must take. This is virtually never used.

The more common form is **loose source record route (LSRR)**, in which the sender gives one or more hops that the packet must go through. In high-level terms, it may look like:

To : A
From : D
Via : T

Source routing is used for the following purposes:

Mapping the network

Used with trace route in order to find all the routes between points on the network.

Troubleshooting

Trying to figure out from point "A" why machines "F" and "L" cannot talk with each other.

Performance

A network manager might decide to force an alternate link (such as a satellite connection) that is slower, but avoids congesting the correct routes.

Hacking

LSRR can be used in a number of ways for hacking purposes. Sometimes machines will be on the Internet, but will not be reachable. (It may be using a private address like 10.0.0.1). However, there may be some other machine that is reachable to both sides that forwards packets. Someone can then reach that private machine from the Internet by source routing through that intermediate machine.

Distance Vector

(Also known as Bellman-Ford or Ford-Fulkerson)

The heart of this algorithm is the routing table maintained by each router. The table has an entry for every other router in the subnet, with two pieces of information: the link to take to get to the router, and the estimated distance from the router. For a router A with two outgoing links L1, L2, and a total of four routers in the network, the routing table might look like this:

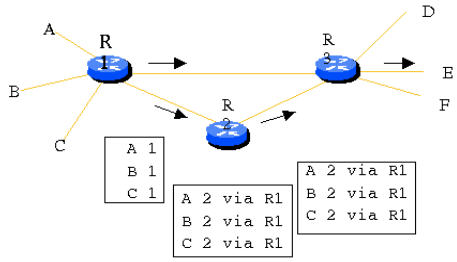
Router	Distance	link
B	5	L1
C	7	L1
D	2	L2

Neighboring nodes in the subnet exchange their tables periodically to update each other on the state of the subnet (which makes this a dynamic algorithm). If a neighbor claims to have a path to a node which is shorter than your path, you start using that neighbor as the route to that node. Notice that you don't actually know the route the neighbor thinks is shorter - you trust his estimate and start sending packets that way.

Issues in DV protocols

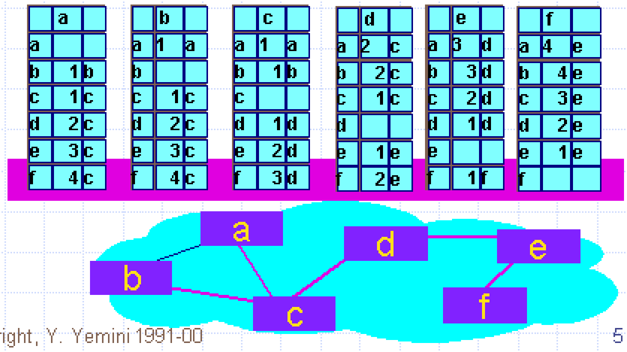
- Forwarding loops
- Slow convergence
  - Count to infinity
  - Split Horizon
  - Poison Reverse
- Protocol Overhead - each router exchanges complete routing table periodically

Distance Vector Routing Example



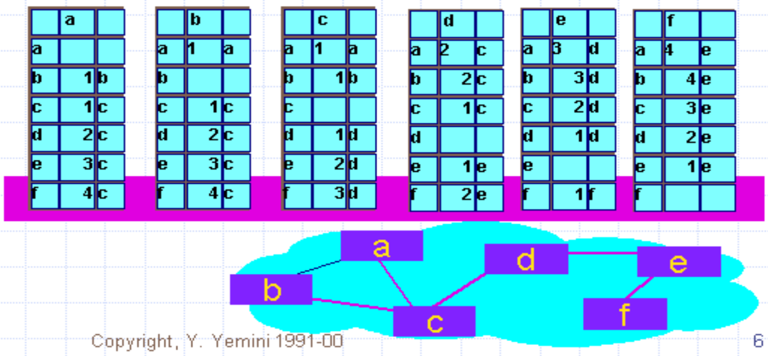
Distance Vector Routing

- Shortest-path routing
  - Origins: Ford-Bellman algorithm, Paul Baran, "Old Arpanet" routing algorithm, RIP...
- Each node maintains a routing table:
  - Entry for each destination
  - Estimated shortest distance to destination
  - Next router on shortest path tree



Distance Vector Routing

- Distance metrics: hop-count, delay, bandwidth...
- Distributed data structure
  - Rows encode structure of shortest path tree



RIP

(Routing Information Protocol)

The Routing Information Protocol (RIP) is a distance-vector protocol that uses hop count as its metric. RIP is widely used for routing traffic in the global Internet and is an *interior gateway protocol* (IGP), which means that it performs routing within a single autonomous system. Peer routers exchange distance vectors every 30 sec, and a router is declared dead if a peer does not hear from it from 180 sec. The protocol uses split horizon with poisonous reverse to avoid the count-to infinity problem.

RIP sends routing-update messages at regular intervals and when the network topology changes. When a router receives a routing update that includes changes to an entry, it updates its routing table to reflect the new route. The metric value for the path is increased by one, and the sender is indicated as the next hop. RIP routers maintain only the best route (the route with the lowest metric value) to a destination. After updating its routing table, the router immediately begins transmitting routing updates to inform other network routers of the change. These updates are sent independently of the regularly scheduled updates that RIP routers send.

RIP Stability Features

To adjust for rapid network-topology changes, RIP specifies a number of stability features that are common to many routing protocols. RIP, for example, implements the split-horizon and hold-down mechanisms to prevent incorrect routing information from being propagated. In addition, the RIP hop-count limit prevents routing loops from continuing indefinitely.

Applications

RIP is useful for small subnets where its simplicity of implementation and configuration more than compensates for its inadequacies in determining with link failures and providing multiple metrics.

Link-State Routing

Widely used today as OSPF in the Internet, replaced Distance Vector in the ARPANET. Link State improves the convergence of Distance Vector by having everybody share their idea of the state of the net with everybody else (more information is available to nodes, so better routing tables can be constructed).

Neighbor discovery

Send a HELLO packet out. Receiving routers respond with their addresses, which must be globally unique.

Measure delay

Time the round-trip for an ECHO packet, divide by two. Question arises: do you include time spent waiting in the router (i.e. load factor of the router) when measuring round-trip ECHO packet time or not?

Bundle your info

Put information for all your neighbors together, along with your own id, a sequence number and an age.

Distribute your info

Ideally, every router would get every other router data simultaneously. This can't happen, so in effect you have different parts of the subnet with different ideas of the topology of the net at the same time. Changes ripple through the system, but routers that are widely spread can be using very different routing tables at the same time. This could result in loops, unreachable hosts, and other types of problems.

Compute shortest path tree

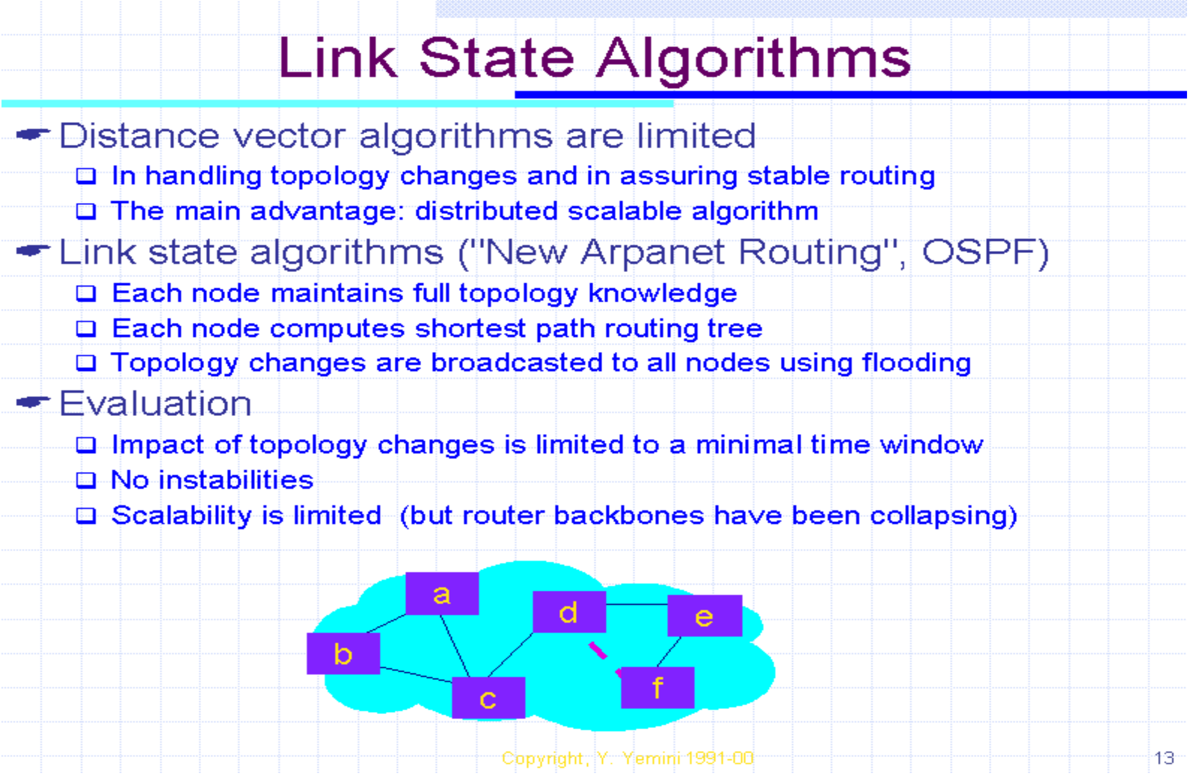
Using an algorithm like Dijkstra's, and with a complete set of information packets from other routers, every router can locally compute a shortest path to every other router.

Advantages

OSPF uses **authentication** for routing messages. This avoids the problem of miss configured, "crazy" routers suddenly trying to tell the world that they are the center of the universe and are directly connected to a wormhole time travel machine. It also helps prevent malicious attacks on networks via their routing tables.

OSPF uses the **idea of "area"** within a routing domain. This decreases the amount of state information, and exchange of routing messages

OSPF allows for **load balancing** among multiple routers



Problem Definition

Problem Definition

For sending information from one network to other network through a subnet efficiently, one has to select a better routing technique among the several techniques available. So far no routing Algorithm is reported to be outright choice for all possible cases. So an attempt is made to provide such a routing technique which provides better results for a given configuration of the subnet in real time.

The main objective of our project is to maximize the efficiency of the routing process by suggesting the potential user a better algorithm.

Calculation of Efficiency of Subnet:

Efficiency of Routing Algorithm = $\sum \eta_i / n$

Where

η_i is Efficiency of Router i

n is Number of Routers in the Subnet

Objectives of the System

The **Routing Simulator** has the following **objectives**:

1. The topology of the subnet should be displayed with routers designated with computer images and links with lines.

2. The congestion table should be printed showing the congestion on various links.

3. Various Statistics for the router like efficiency, average packet size are to be displayed when required.

4. Statistics for the link like propagation delay, buffers filled are to be displayed when congestion table is clicked.

5. A provision for router crash is required which should show an outline when a router is crashed.

6. When a link is down, it should be highlighted. This gives an idea of how the routing goes when a link is down.

7. The statistics for the router and the link are to be calculated for every 500 m sec.

8. A provision for redrawing the congestion table and network is to be provided.

9. The routing should be controlled by a speed controller.

System Design

Design

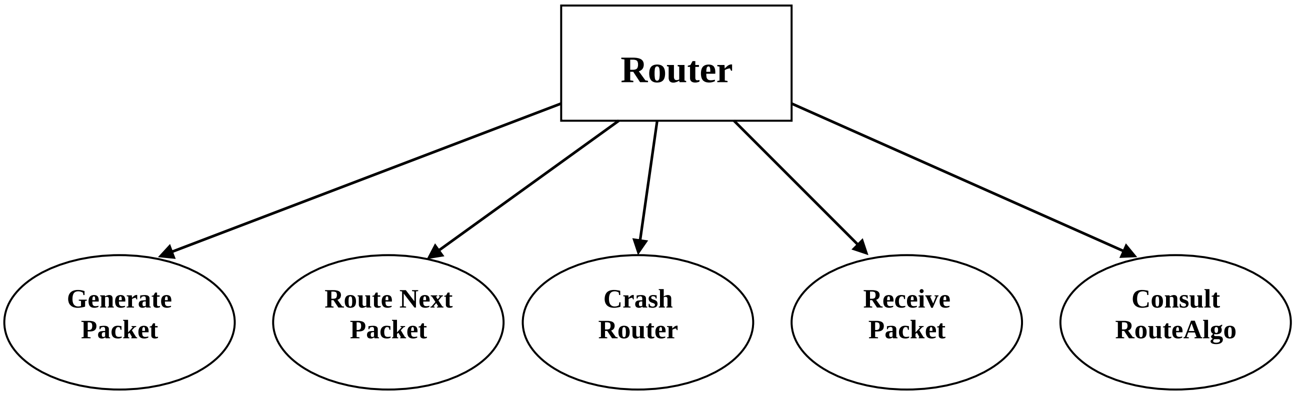
In the design phase we identify the different objects that are required to get the required results:

Router

A router generates packets and places them in the router buffer. It consults the routing algorithm and places the appropriate packets on the link from which packets are placed on the link and transmitted. The router at the other end of the link picks up the packets from the link and places them in its buffer. From the router's buffer packets are processed. Processing includes checking if the packet is destined to that router or not. If yes the router reads the message and sends acknowledgement else it sends it to the router buffer from which it is forwarded to the next router.

The router contains the following fields:

- Id of router
- Size of Network
- Distance matrix to every router
- Simcore object
- Buffer object
- Maximum Size of Buffer
- Current Size of Buffer;
- Routing Algorithm
- Routing Algorithm Status
- Router Status
- Checking Probability
- Threshold
- Start Delay
- Outgoing link matrix
- Incoming Links matrix



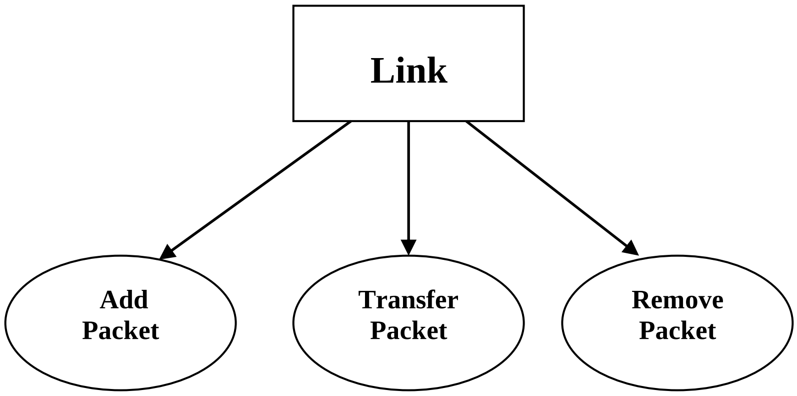
Router and its methods

Link

The link acts as a bridge between the two routers. There are buffers at both ends of a link in which packets are placed while routing. The link transmits the packets based on the propagation delay specified. According to the delay the packets are sent immediately to the next router or delayed in the link buffer. When the state of a link is up transmission occurs. If the link goes down packets in the link buffers are lost.

The link contains the following fields:

- Head Packet
- Tail Packet
- Queue Size
- Thread for Link
- Propagation Time
- Bit Rate
- Status of Link
- Router at one end
- Router at the other end
- Simcore object



Link and its methods

Routing Algorithm

A routing algorithm decides the path from source to the destination. It performs three major functions:

Routing of next packet

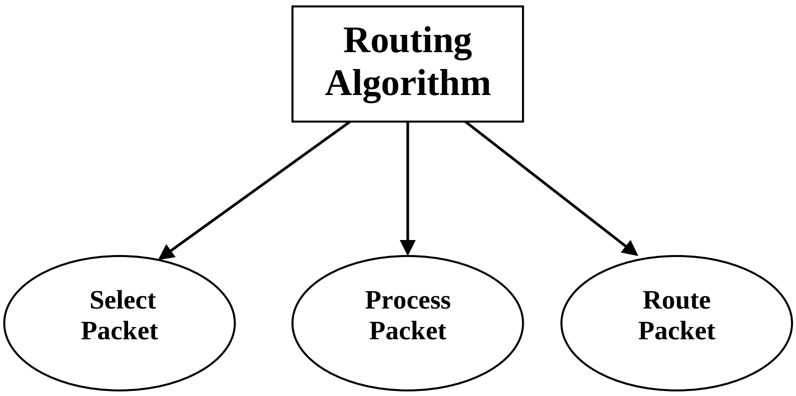
To do with packet as algorithm decides.

Processing of Packet

If the packet is for that router, we can use this method to send some kind of hello messages or replies.

Selection of Packet to be routed

A packet from the router's buffer will be selected to be routed next. This is useful when packets have different priorities.



Routing Algorithm and its methods

Packet

A packet is an entity which contains the actual message to be sent to the router. The router generates packets that are transmitted through a link. The packets are dummy entities and are just required to know that the path given by the routing algorithm is followed or not. The rate of transfer of packets depends on bandwidth of the link.

The Packet has the following fields:

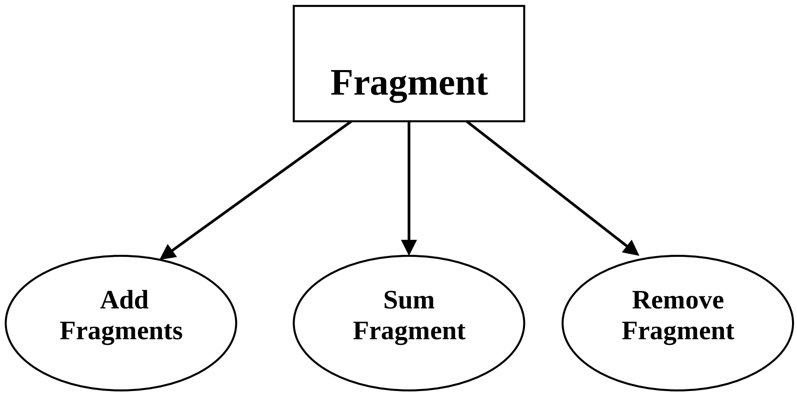
- Size of the Packet
- Packet Source
- Packet destination
- Sequence Number (equivalent to packet id)
- Fragment Offset
- Total Size
- Time when Packet reaches other side of link
- Creation Time
- Time when Put In Link
- Previous Router
- Next Router
- Hop count
- Time when Inserted In Buffer
- Fragment Flag
- Fragmented Flag

Fragment

If the size of the packet is large as compared with the size of the buffers, the packet is divided into number of fragments. These fragments are numbered and are placed on links for transmission. At the other end of the buffer, the fragments are again combined to get the original packet.

Fragment contains the following fields:

- Sequence number
- Time when Fragmented
- Fragment Number
- Fragment Size



<u>Fragment and its methods</u>

Core of the Simulator

This acts as the project manager and is the heart of the Simulator. It takes the input from the input file and initializes the routers and links based on the Network Configuration. It manages the time constraint based on which packets are generated and lost. It consults the routing algorithm and decides the path and gives instructions to the packets in the buffers about the paths. Based on probabilities of link to be down, it downs a link and after sometimes brings it back to normal state. It checks each and every condition of every other object in the system and takes decisions accordingly. It is responsible for drawing the congestion table and Network diagram in the Panel.

Parameters of the network include:

- Factor for Converting Computer Time to loop-count i.e. our clock
- Frequency of generation of packets at a particular router
- Scaling factor for generating packets
- Distance between routers i & j. Set by the user
- Maximum Packet Size
- Minimum Packet Size
- Number of routers in the N/W. Set by the user
- Header Size
- Array of references to routers
- Head of linked list containing packets which are on their path on a link
- Tail of linked list containing packets which are on their path on a link
- Number of packets lost
- Number of packets sent from particular router
- History of packets which have reached their
- History of packets which have been sent
- Lost History
- When The Underlying layer will be free
- Propagation Delay between Router i & j
- Bit Rate of links between Router i & j
- No of protocol Packets from i to j

- Gross Lost - lost but duplicate of packet may have reached destination
- Net lost - no copy has reached destination
- Protocol packets lost
- Probability of Packet Loss On Link
- Maximum Link Size
- Snap Shot Interval
- Maximum fragment size

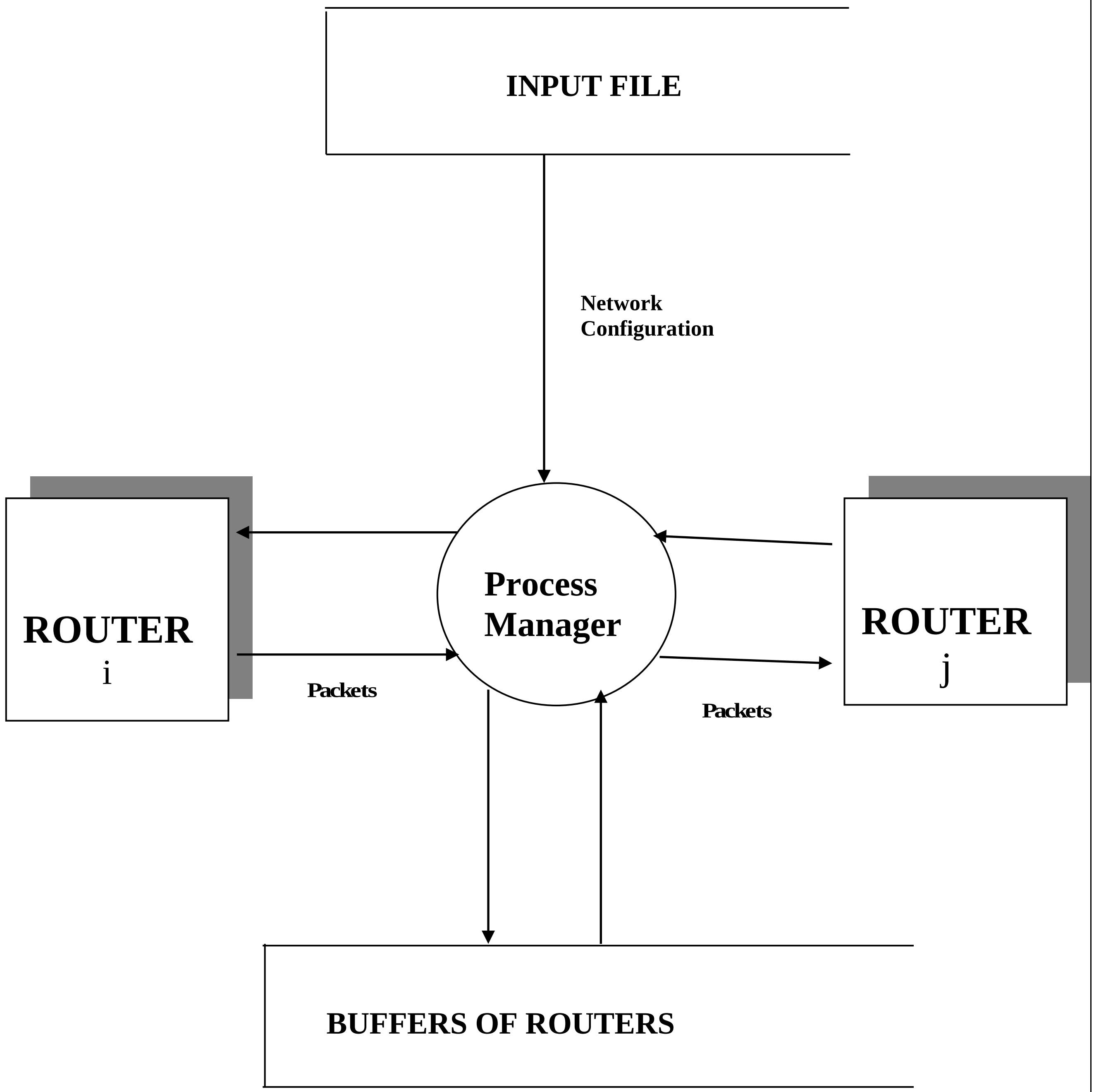
The different issues handled by the Simulator are:

- Throughput
- Read speed of graphical display/routing
- warning message shown for how much time

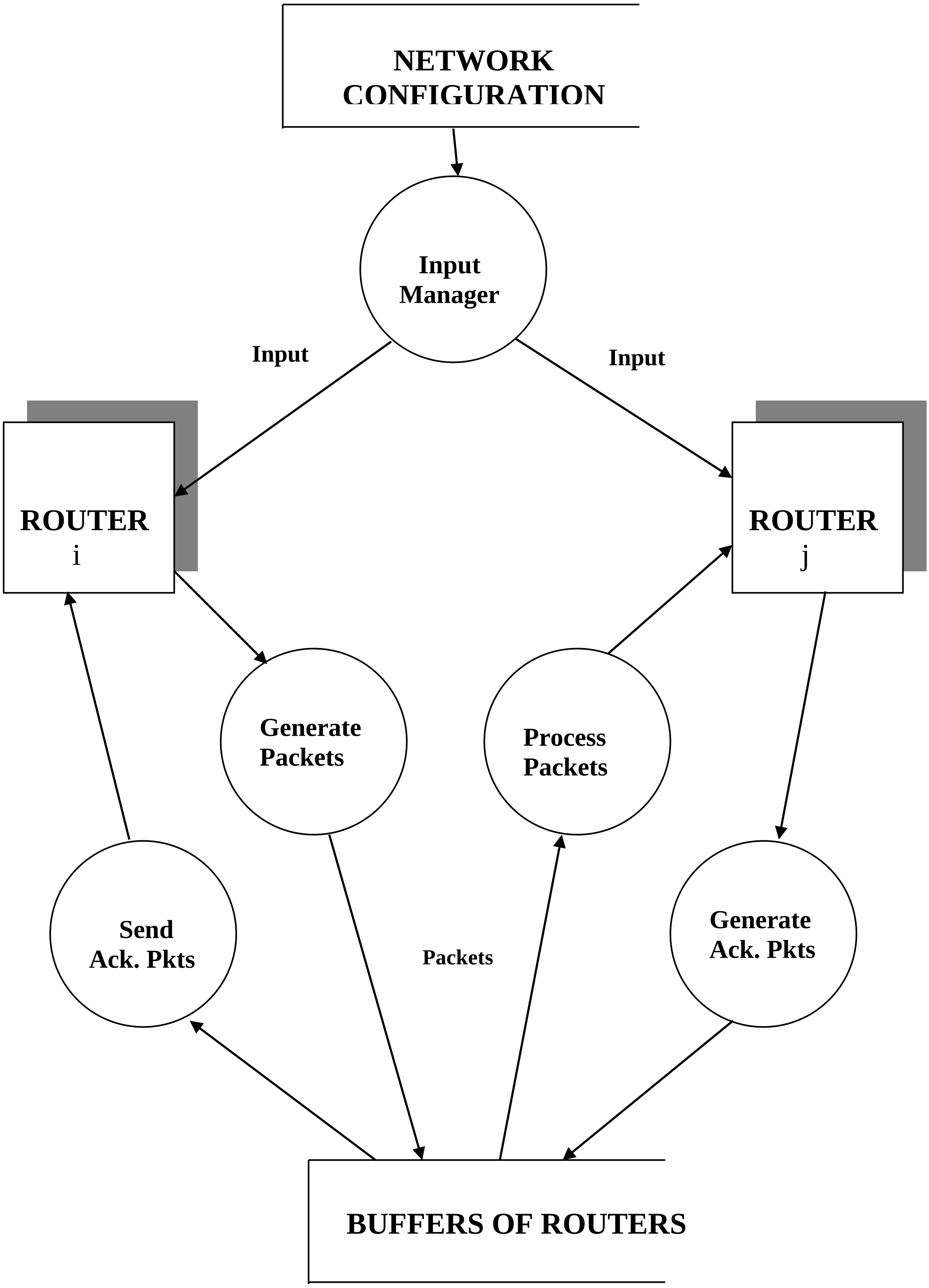
The some of the modules in the core of the Simulator include:

- Setting the Topology
- Drawing the Table
- Filling the Table
- Drawing the Network
- Restoring the State
- Setting the Physical distance
- Notifying the Link
- Notifying the Router
- Making the Router Status Down
- Making the Link Status Down
- Making the Router Status Up
- Making the Link Status Down

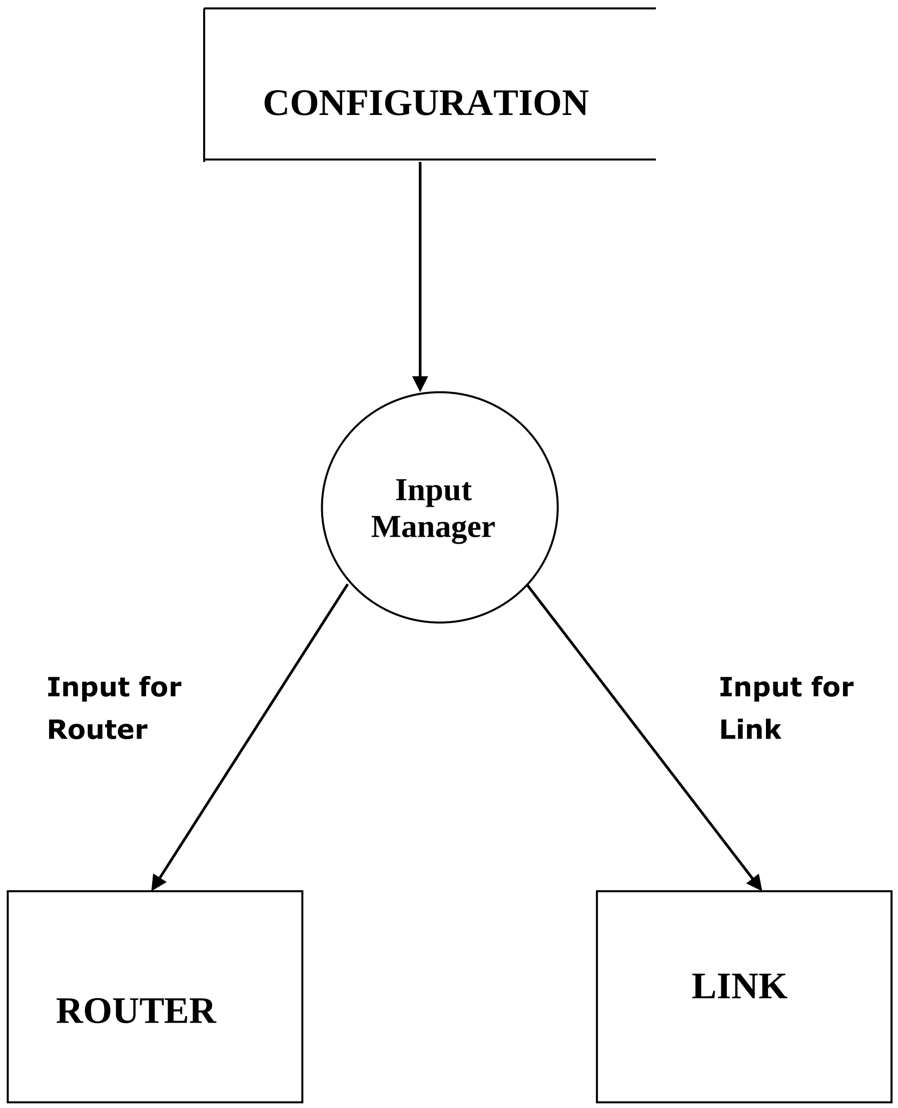
Data Flow Diagrams


INPUT FILE
Network
Configuration
ROUTER
i
Process
Manager
ROUTER
j
Packets
Packets
BUFFERS OF ROUTERS

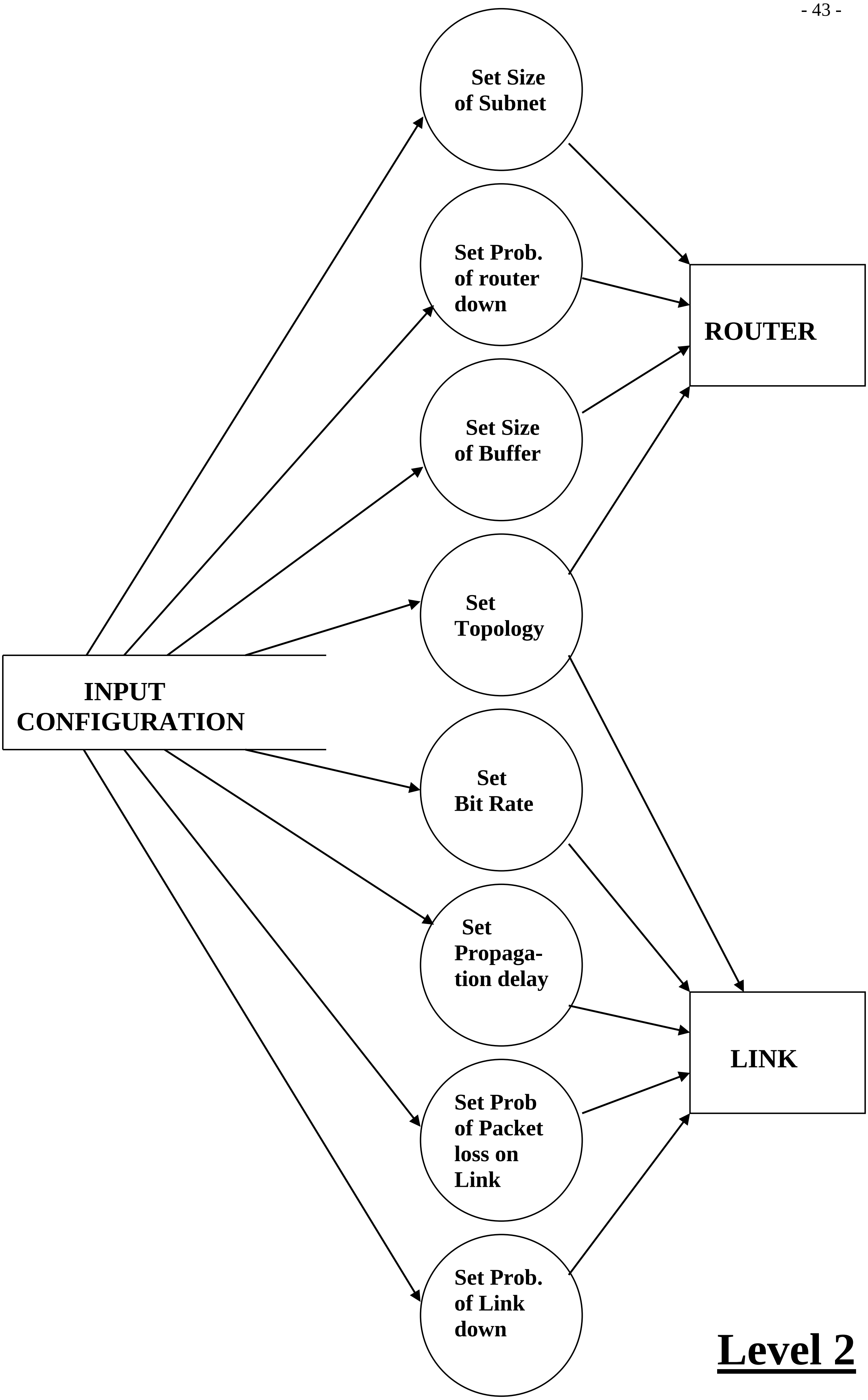
Level 0


NETWORK CONFIGURATION
Input Manager
Input
Input
ROUTER i
ROUTER j
Generate Packets
Process Packets
Send Ack. Pkts
Packets
Generate Ack. Pkts
BUFFERS OF ROUTERS

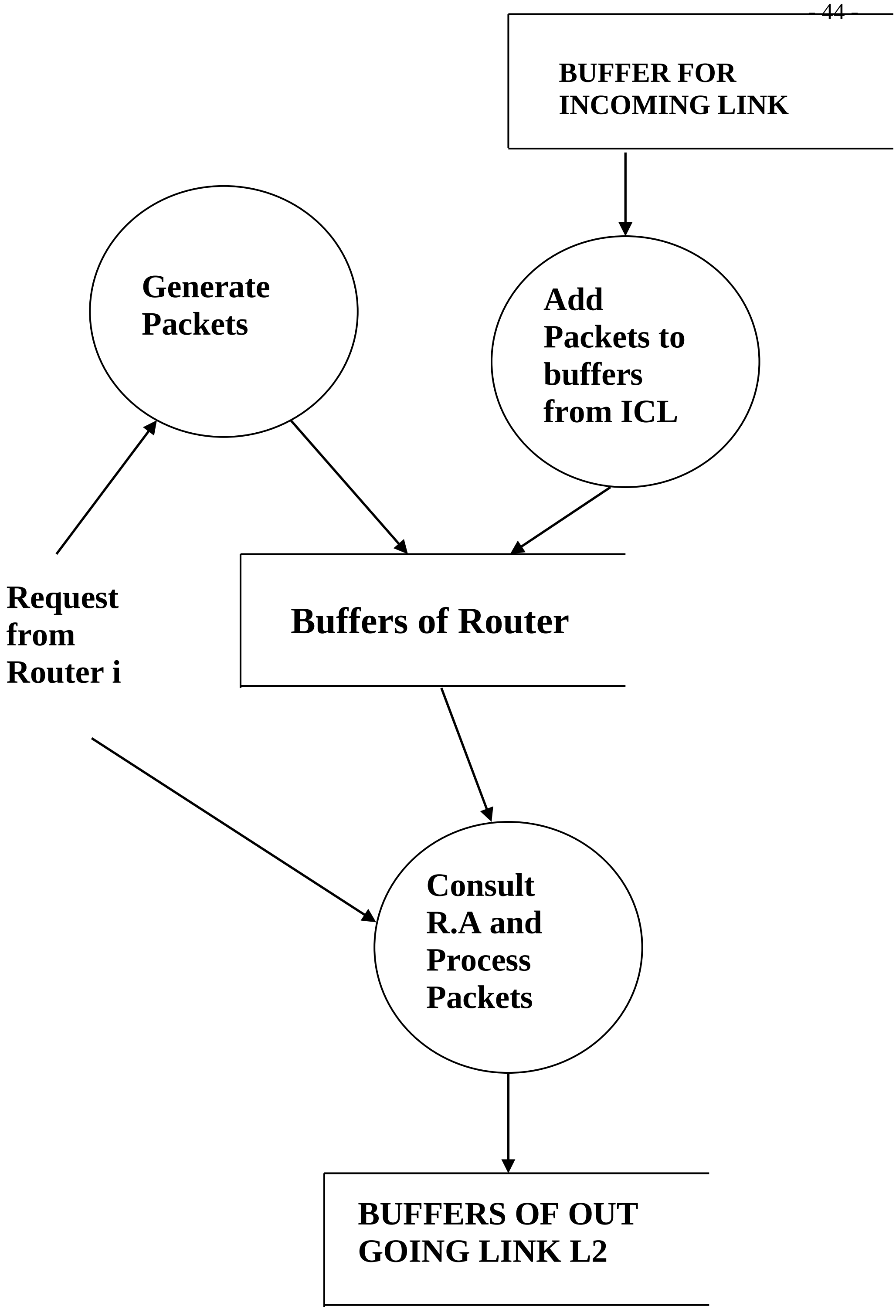
Level 1


CONFIGURATION
Input Manager
Input for Router
Input for Link
ROUTER
LINK

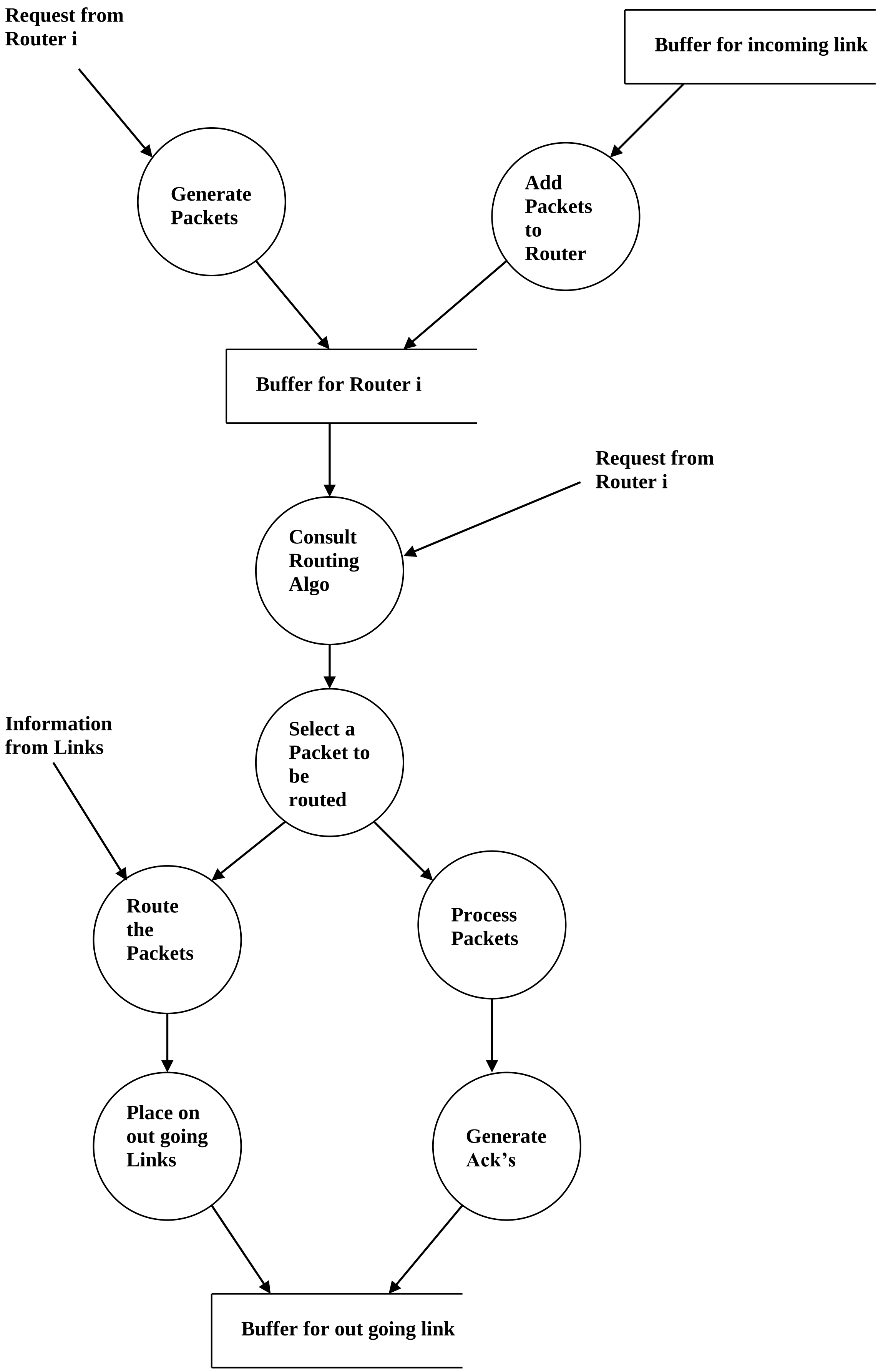
Level 1


Set Size of Subnet
Set Prob. of router down
Set Size of Buffer
Set Topology
INPUT CONFIGURATION
Set Bit Rate
Set Propaga-tion delay
Set Prob of Packet loss on Link
Set Prob. of Link down
ROUTER
LINK

Level 2


BUFFER FOR
INCOMING LINK
Generate Packets
Add Packets to buffers from ICL
Request from Router i
Buffers of Router
Consult R.A and Process Packets
BUFFERS OF OUT GOING LINK L2

Level 2


Request from Router i
Buffer for incoming link
Generate Packets
Add Packets to Router
Buffer for Router i
Request from Router i
Consult Routing Algo
Information from Links
Select a Packet to be routed
Route the Packets
Process Packets
Place on out going Links
Generate Ack’s
Buffer for out going link

Level 3

Design Implementation

Software Used

Java Development Kit 1.3

The JDK 1.3 is a development environment for writing GUI and applications that confirm to the Java core API. Its compiler and other tools are run from a shell and have no GUI interface.

Java Tools:

Java Compiler (javac)

Compiles programs written in Java programming Language into byte codes.

Java Interpreter (java)

It executes java byte codes. In other words it runs programs written in the Java programming language.

Java run-time Interpreter (jre)

It is similar to Java interpreter, but intended for end users who do not enquire all the development-related options available with the java tool.

Java Debugger (jdb)

It helps in finding bugs in Java programs.

Class File Disassembler (javap)

It disassembles compiled Java files and prints out a representation of Java byte codes.

Java Documentation Generator (javadoc)

Parses the declarations and documentation comments in a set of Java source files and produces a set of HTML pages describing the public and protected classes, interfaces, constructors, methods and fields. Also produces a class hierarchy on an index of all members.

Java Archive Tool (jar)

Combine many Java class files and other resources into a single jar file it also prepares an executable which can be run by javaw.exe.

User Documentation

Requirements:

- The user has to install JRE 1.3 or above to run this **Routing Simulator 1.0**

How to Run

- The user has to double click the **executable jar file** to run the **Routing Simulator 1.0.**

- The user will see a window to operate the **Routing Simulator 1.0.**

Operating

- First the user have to select the topology in the box provided in the lower right corner of the screen then there will be a dialog appears on the screen prompting the user to give number nods in the network enter a positive integer then click “OK” button then on the left side of the screen the network topology can be viewed with the selected no nodes.

- The can view the routing tables for each and every router by just clicking i\on the routers for this the user have to select a algorithm in the panel provided on the top right corner of the window then clicking the **“Simulate”** button

- The routing algorithm efficiency can be viewed by clicking the button in the panel provided in the top right corner of the screen.

- For the process of evaluating the Routing Algorithm one can even crash the Router by right Clicking on the Router and selecting the **“Down”** button in the popup menu the same process for reactivating a router.

- The user can also view the routing buffer by clicking the "Router Buffer" option in the popup menu, before that user have to send some number of packets using different algorithms using the button **"Send Packets"** which is provided in the top right corner of the screen.

- The user also can use the short notes provided on different routing algorithms for verification by just clicking the **"Notes"** button in the **"Help"** menu.

Testing

System testing makes a logical assumption that if all parts of the system are correct the goal will be successfully achieved. System Testing is utilized as user-oriented vehicle before implementation. Programs are invariably related to one another and interact in a total system. Each portion of the system is tested to see whether it conforms to related programs in the system. Each portion of the entire system is tested against the entire module with both test and live data before the entire system is ready to be tested.

The first test of a system is to see whether it produces correct output. The other tests that are conducted are:

1. Online - Response

When the mouse is clicked on the router the statistics of the router for the selected algorithm have to be displayed on the screen. The router must crash immediately when it the **“Down”** button clicked in the popup menu.

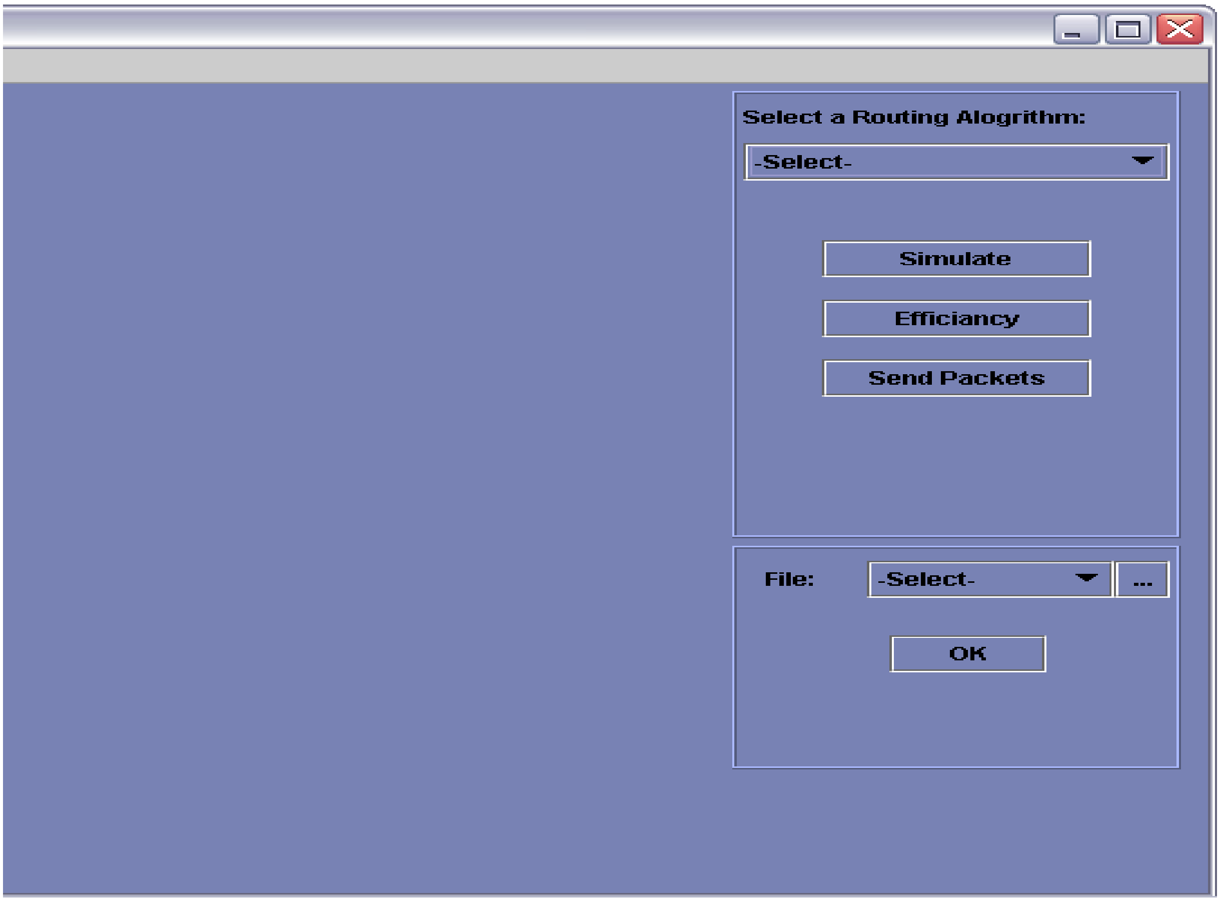
2. Stress Testing

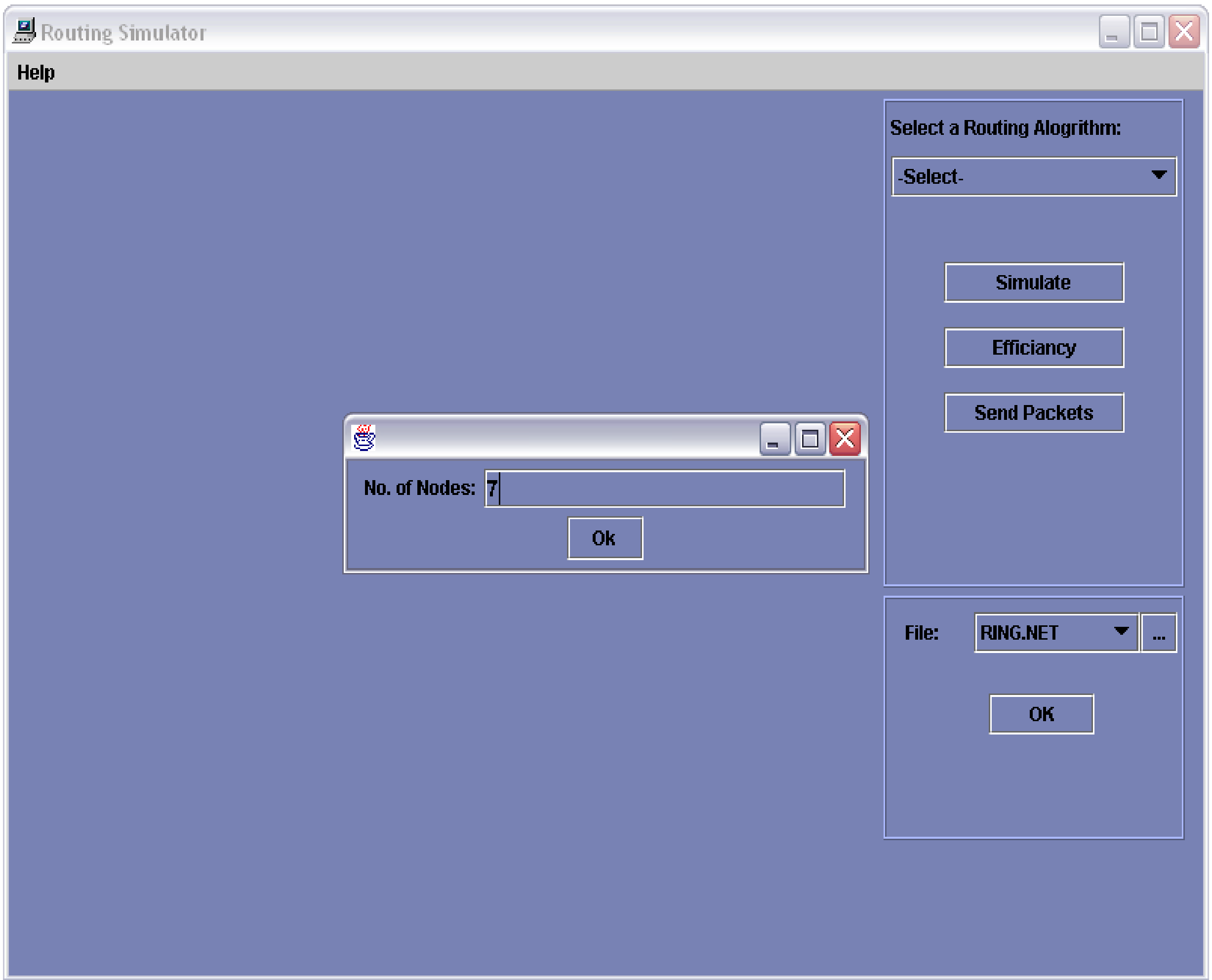
The purpose of stress testing is to prove that the system does not malfunction under peak loads. In the simulator we test it with the greater number of nodes and getting the correct results for each and every router applying different routing algorithms. All the routers are purposely crashed to generate a peak load condition and the working is tested.

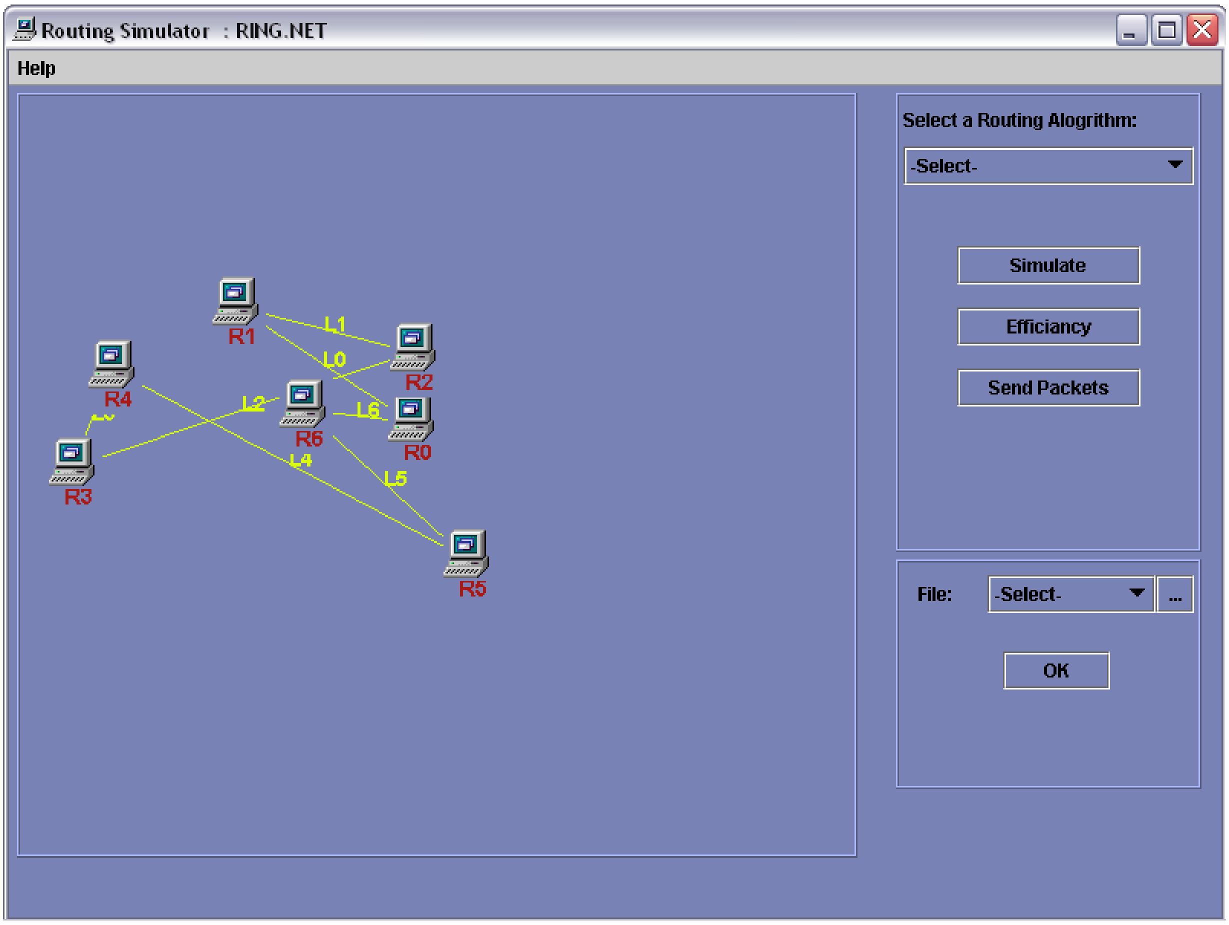
3. Usability Documentation and Procedure

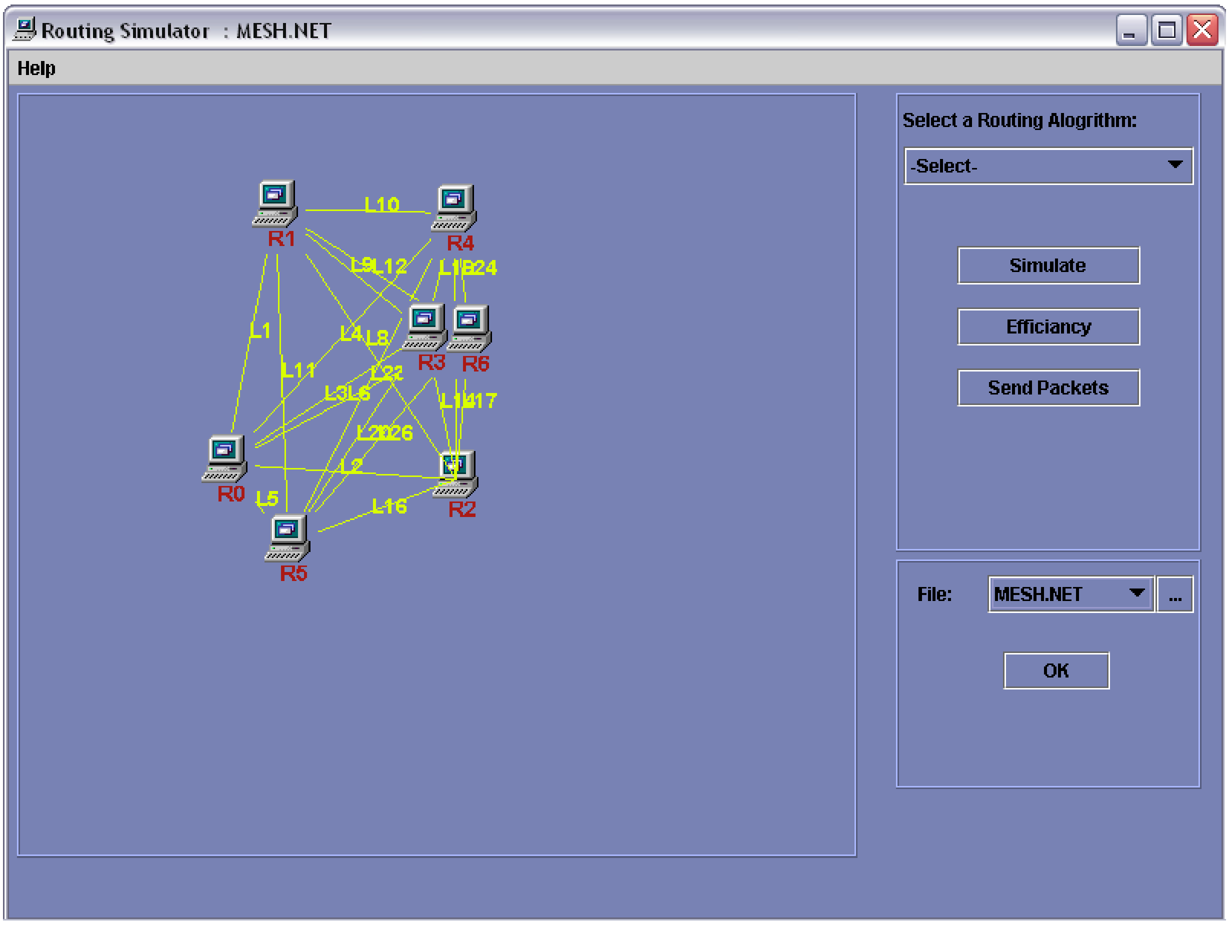
The usability test verifies the user friendly nature of the system. The user is asked to use only the documentation and procedure as a guide to determine whether the system can run smoothly.

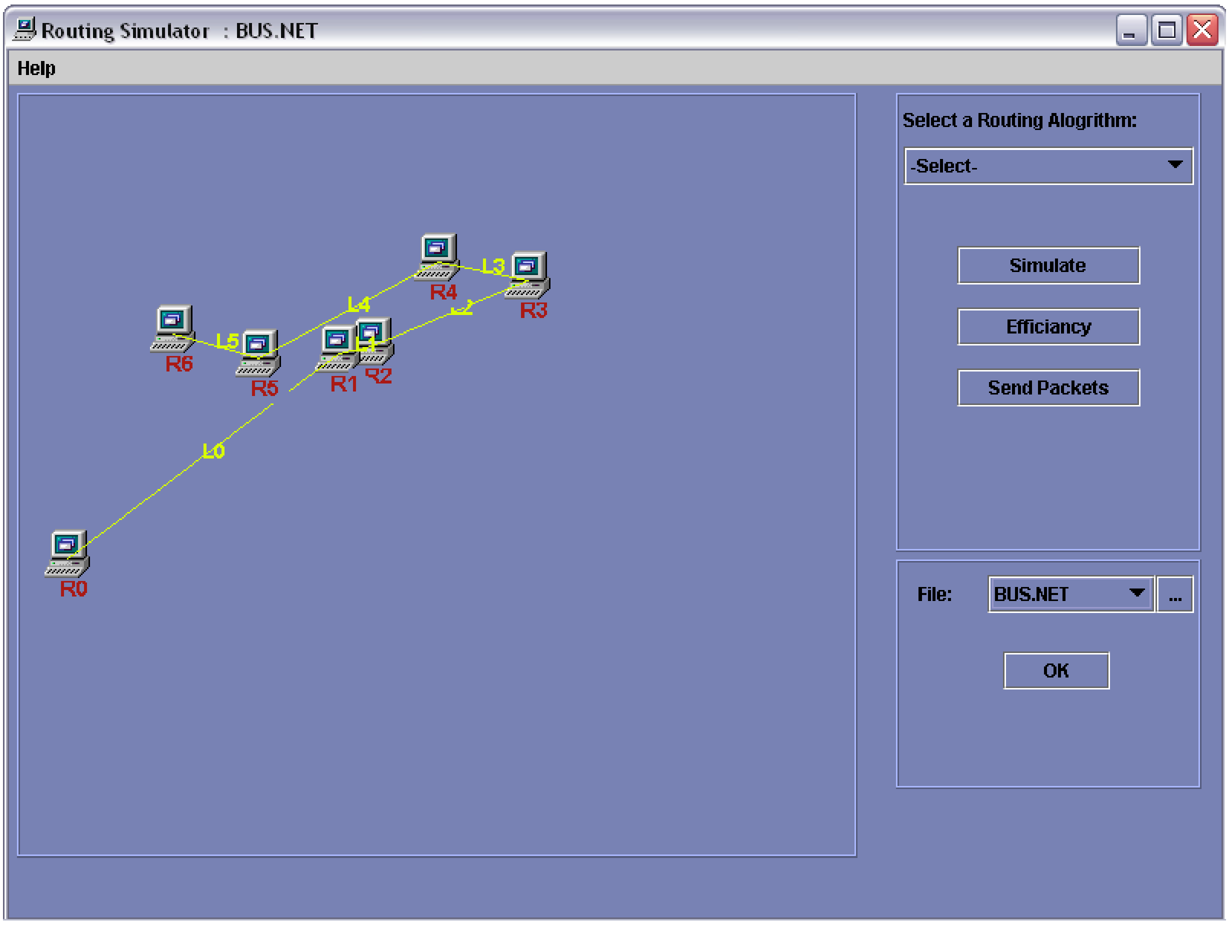
Sample Outputs

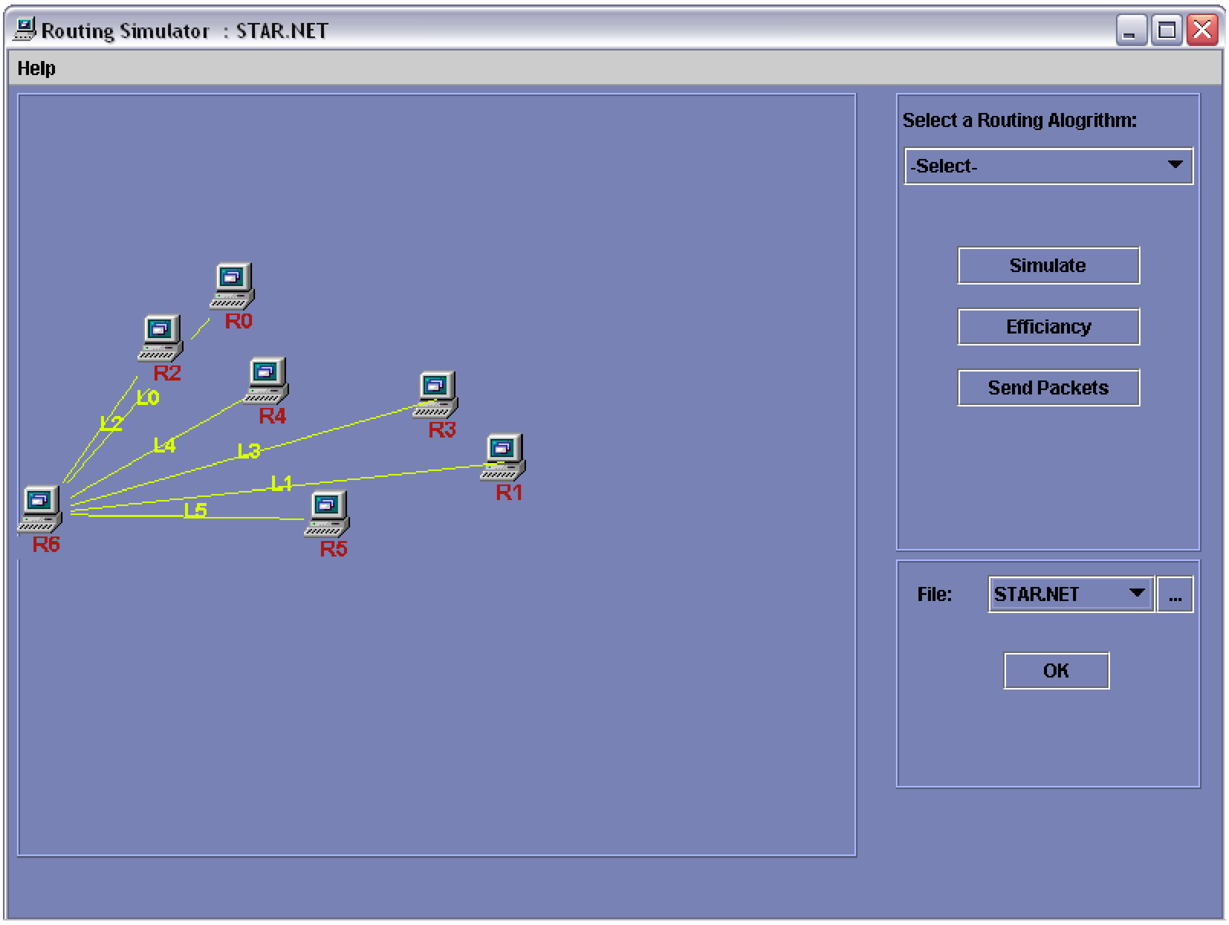

Select a Routing Alogrithm:
-Select-
Simulate
Efficiancy
Send Packets
File:
-Select-
...
OK


Routing Simulator
Help
Select a Routing Alogrithm:
-Select-
Simulate
Efficiancy
Send Packets
No. of Nodes: 7
Ok
File:
RING.NET
...
OK

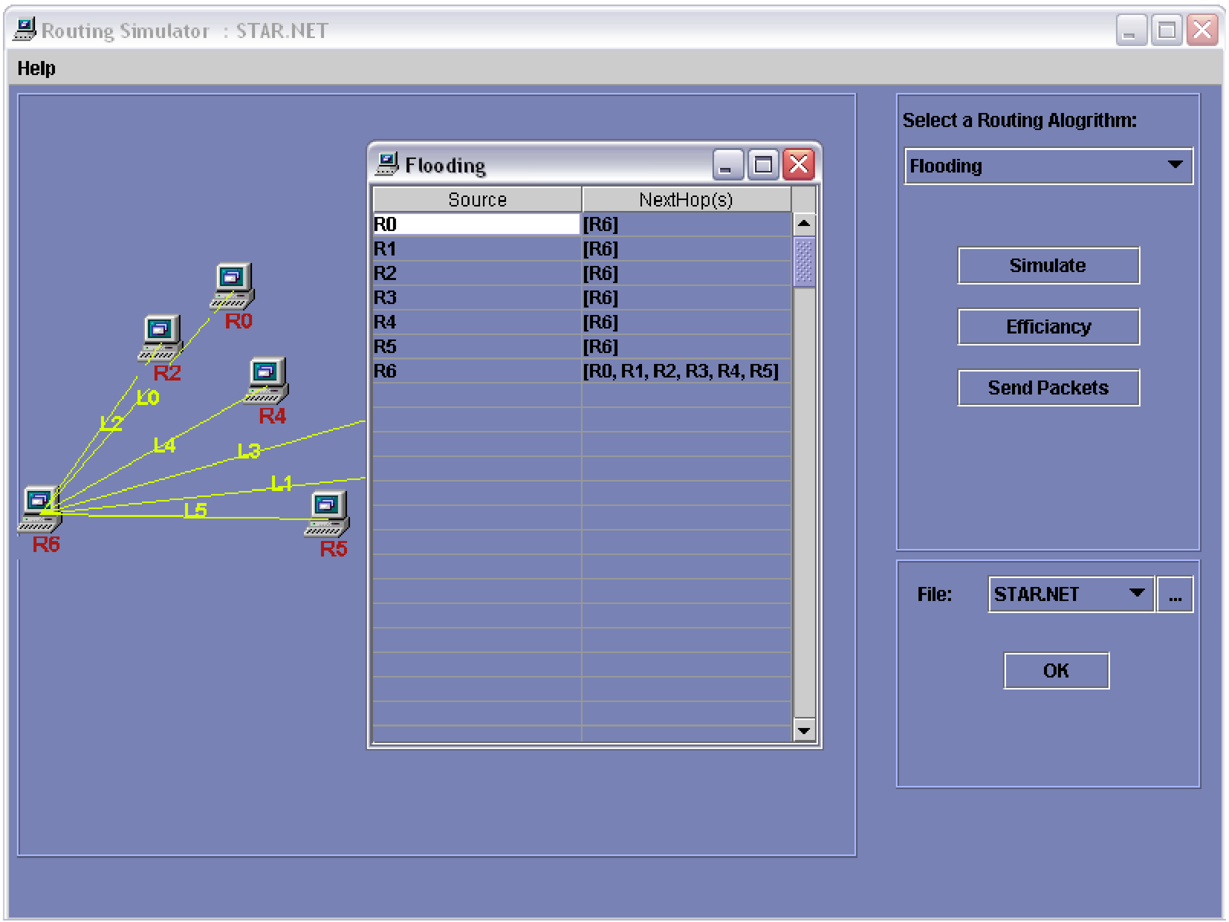

Routing Simulator : RING.NET
Help
R1
R2
R4
R6
R0
R3
R5
L1
L0
L2
L6
L4
L5
Select a Routing Alogrithm:
-Select-
Simulate
Efficiancy
Send Packets
File:
-Select-
...
OK


Routing Simulator : MESH.NET
Help
Select a Routing Alogrithm:
-Select-
Simulate
Efficiancy
Send Packets
File:
MESH.NET
...
OK
R0
R1
R2
R3
R4
R5
R6
L10
L1
L11
L5
L16

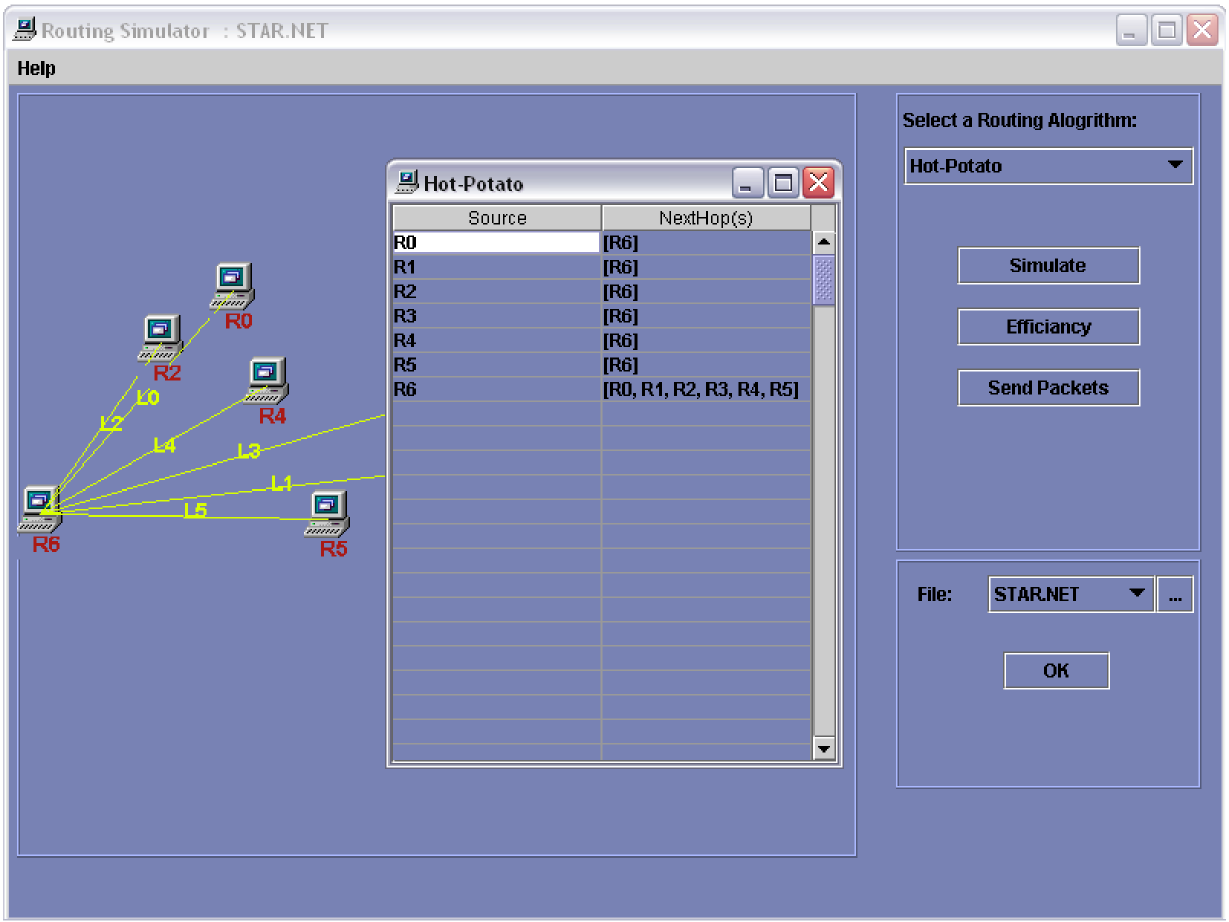

Routing Simulator : BUS.NET
Help
Select a Routing Alogrithm:
-Select-
Simulate
Efficiancy
Send Packets
File:
BUS.NET
...
OK
R0
R1
R2
R3
R4
R5
R6
L0
L3
L4
L5



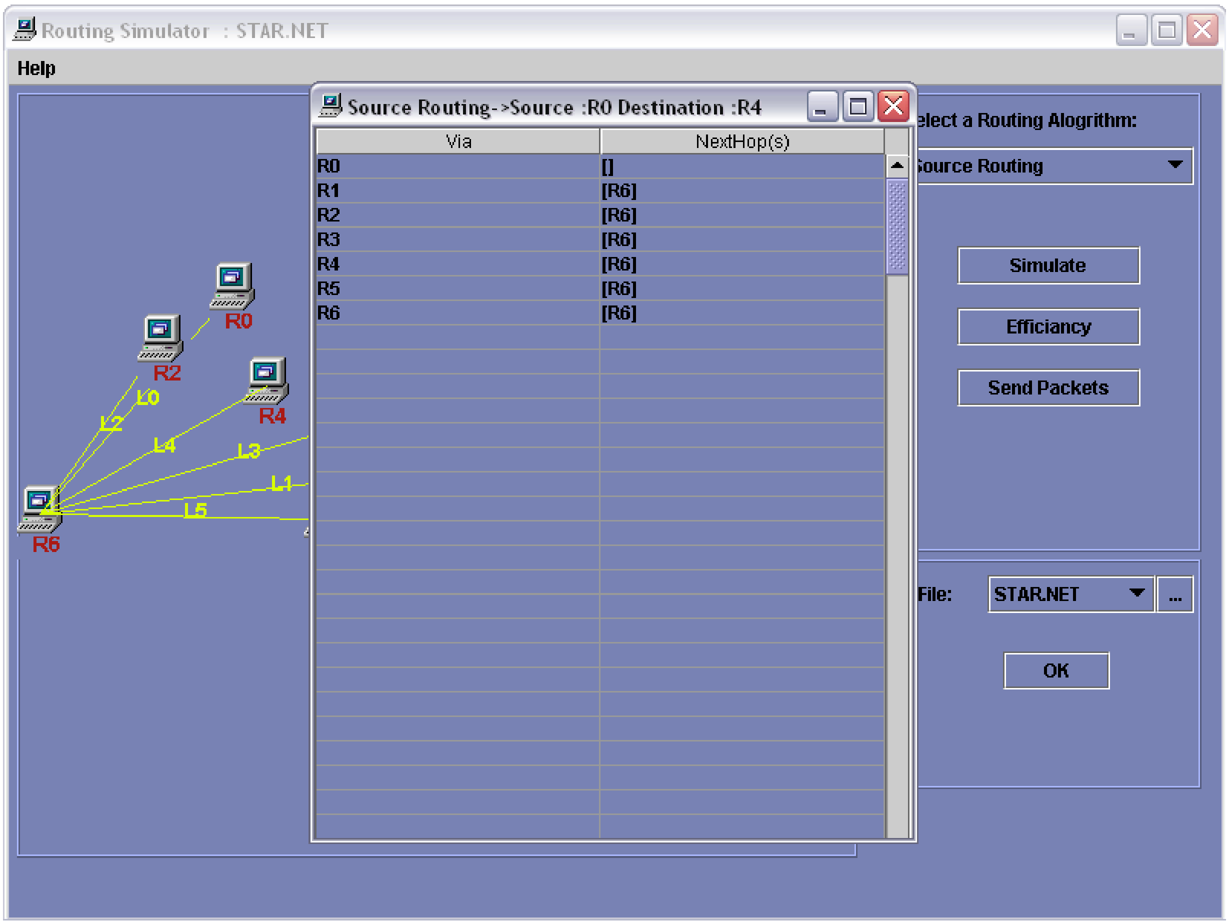
Routing Simulator : STAR.NET
Help
Select a Routing Alogrithm:
-Select-
Simulate
Efficiancy
Send Packets
File:
STAR.NET
...
OK
R0
R2
R4
R3
R1
R6
R5
L0
L2
L4
L3
L1
L5

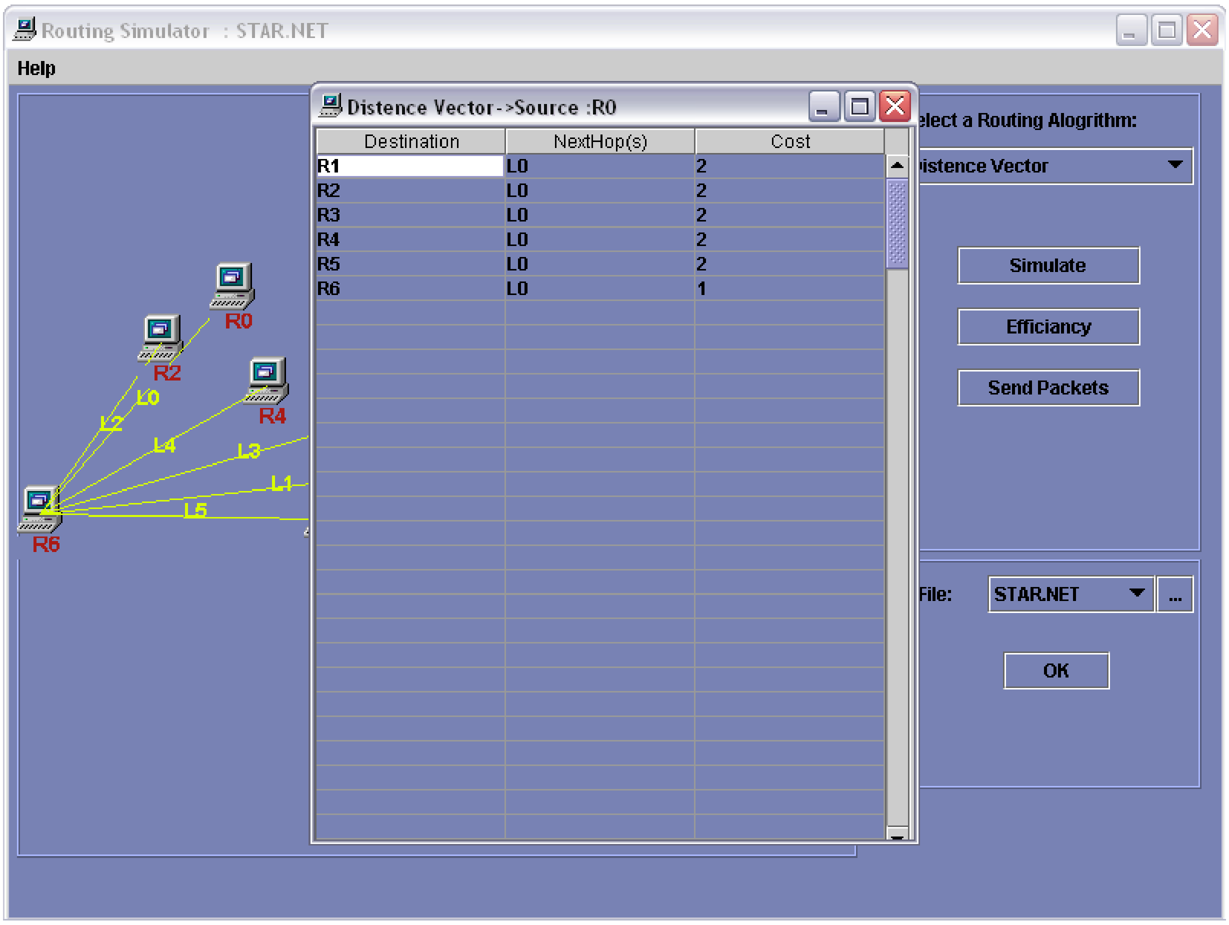


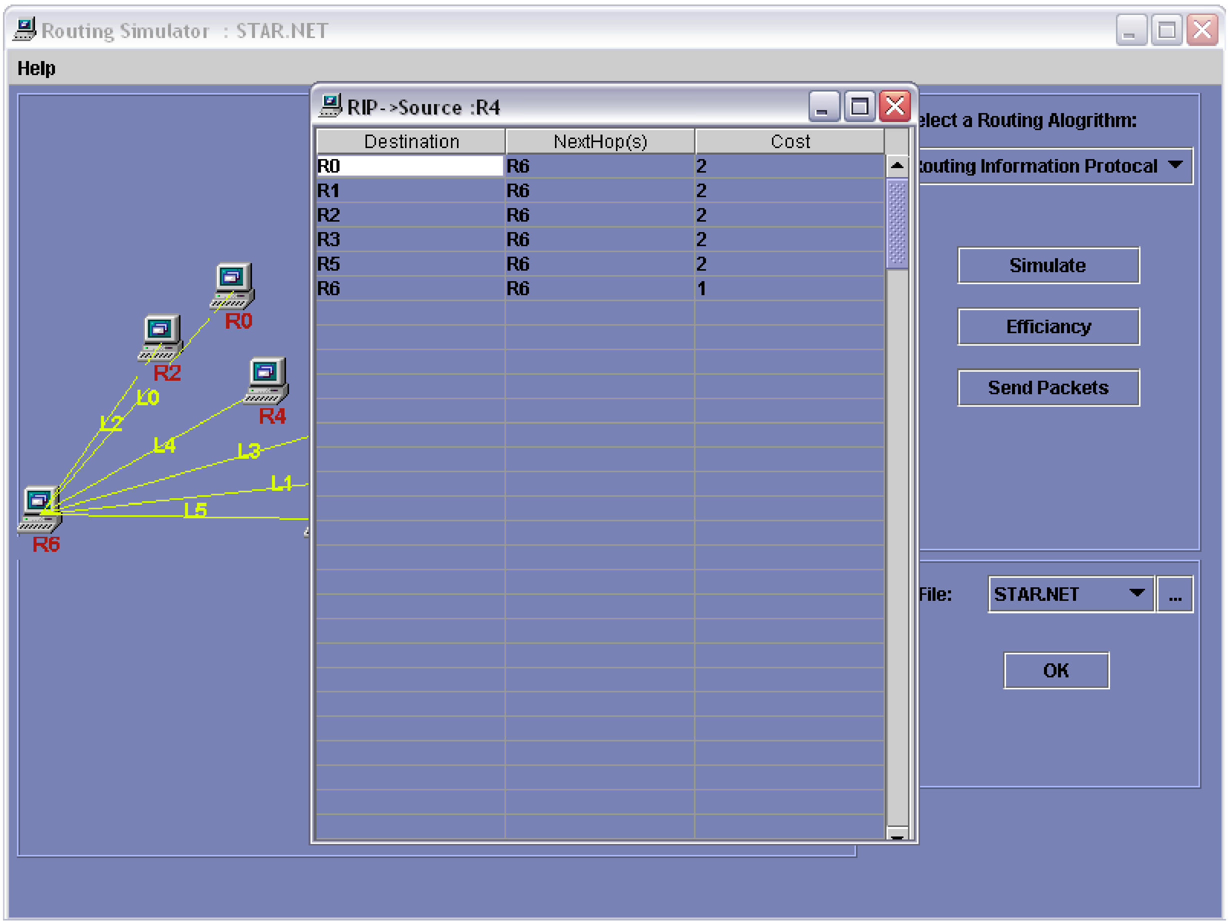
Routing Simulator : STAR.NET
Help
Flooding
Source
NextHop(s)
R0 [R6]
R1 [R6]
R2 [R6]
R3 [R6]
R4 [R6]
R5 [R6]
R6 [R0, R1, R2, R3, R4, R5]
R0
R2
R4
R6
R5
L0
L1
L2
L3
L4
L5
Select a Routing Alogrithm:
Flooding
Simulate
Efficiancy
Send Packets
File:
STAR.NET
...
OK

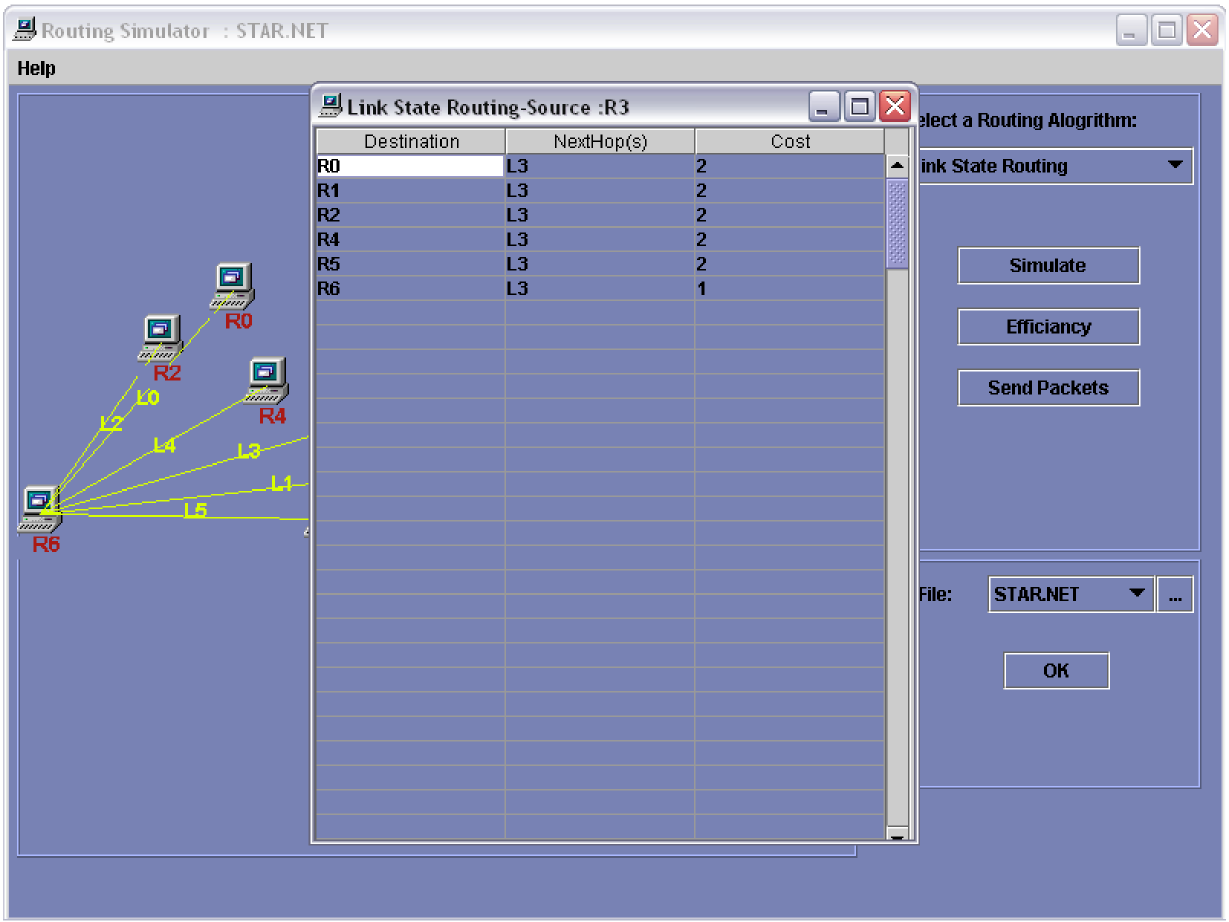


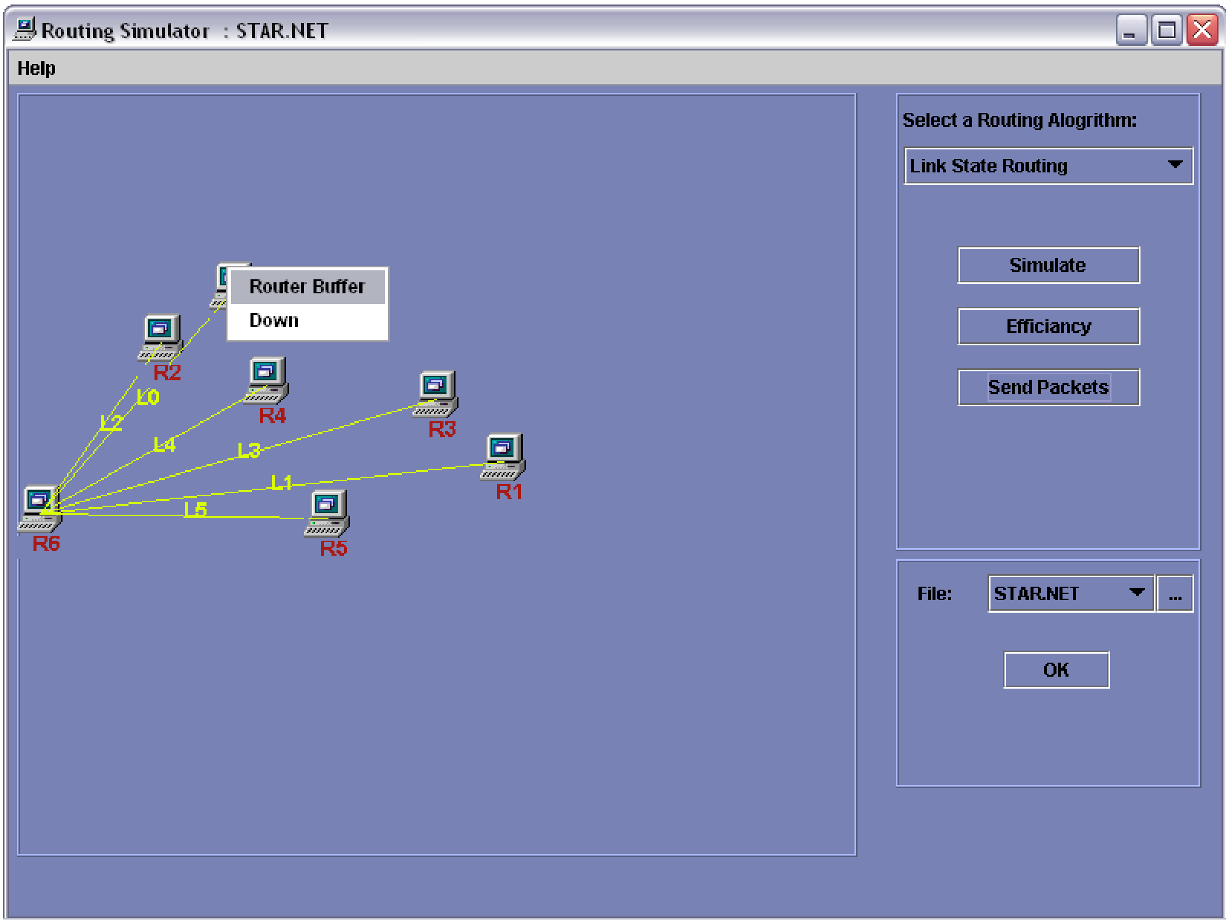
Routing Simulator : STAR.NET
Help
Hot-Potato
Source
NextHop(s)
R0 [R6]
R1 [R6]
R2 [R6]
R3 [R6]
R4 [R6]
R5 [R6]
R6 [R0, R1, R2, R3, R4, R5]
Select a Routing Alogrithm:
Hot-Potato
Simulate
Efficiancy
Send Packets
File:
STAR.NET
OK
R0
R2
R4
R5
R6
L0
L1
L2
L3
L4
L5

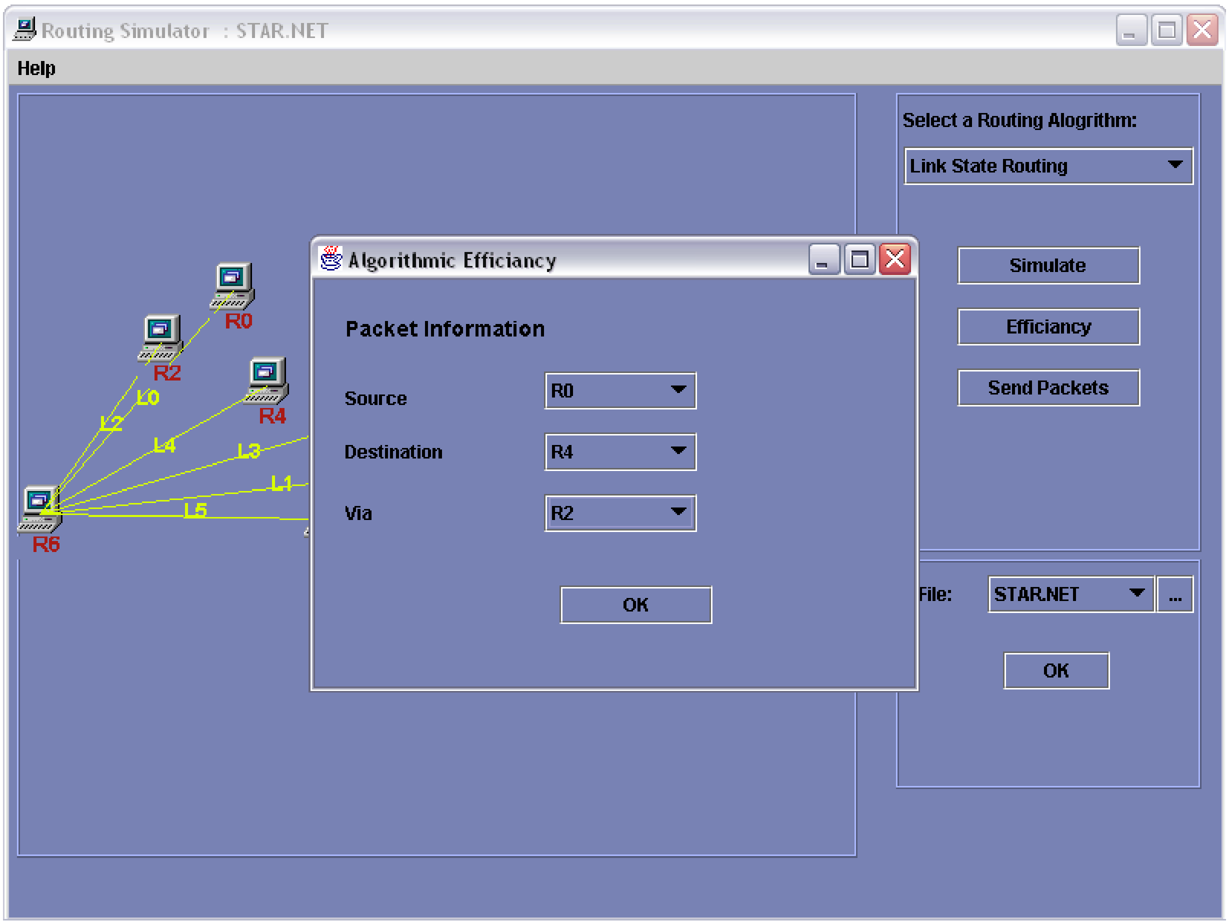

Routing Simulator : STAR.NET
Help
Source Routing->Source :R0 Destination :R4
Via
NextHop(s)
R0 []
R1 [R6]
R2 [R6]
R3 [R6]
R4 [R6]
R5 [R6]
R6 [R6]
elect a Routing Alogrithm:
Source Routing
Simulate
Efficiancy
Send Packets
File:
STAR.NET
OK
R0
R2
R4
R6
L0
L1
L2
L3
L4
L5

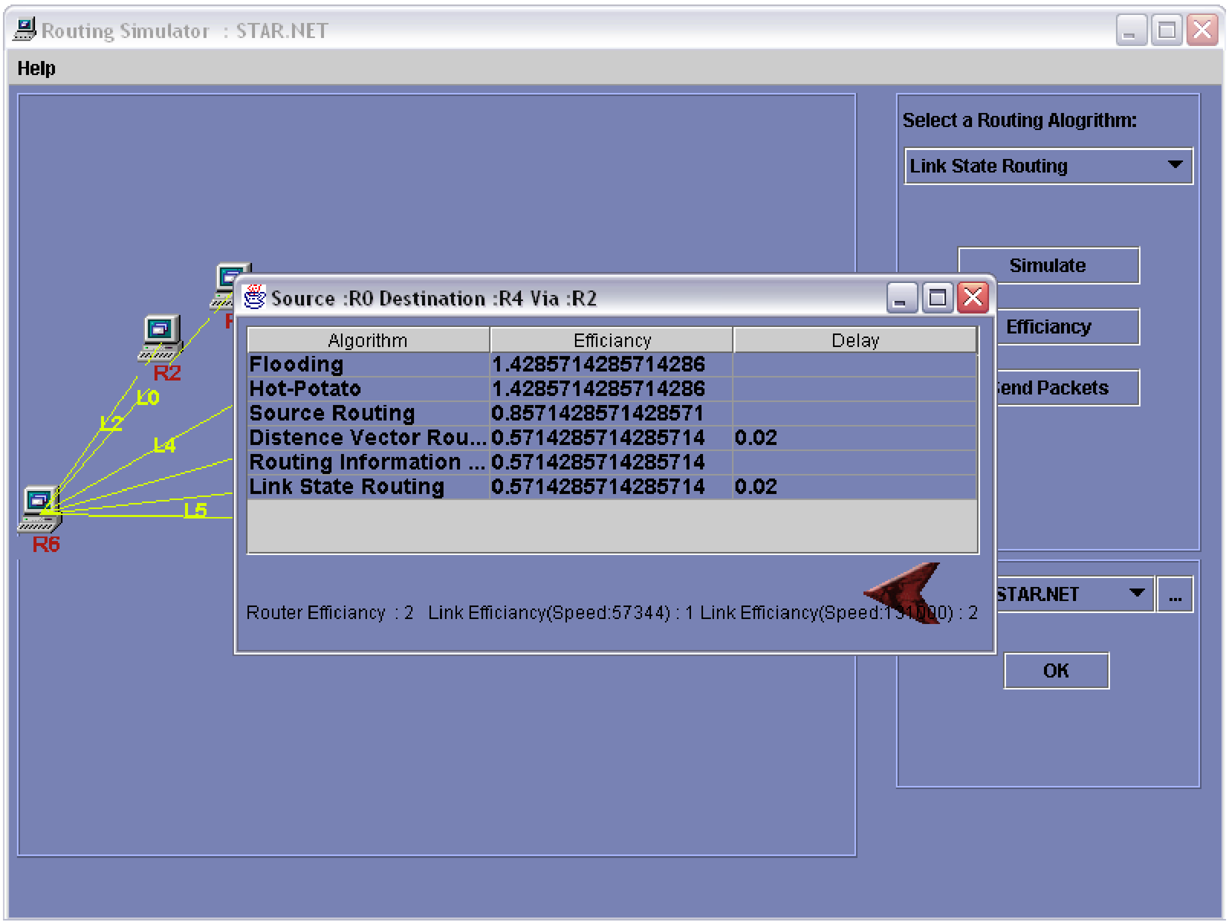

Routing Simulator : STAR.NET
Help
Distence Vector->Source :R0
Destination
NextHop(s)
Cost
R1 L0 2
R2 L0 2
R3 L0 2
R4 L0 2
R5 L0 2
R6 L0 1
elect a Routing Alogrithm:
istence Vector
Simulate
Efficiancy
Send Packets
File:
STAR.NET
OK
R0
R2
R4
R6
L0
L1
L2
L3
L4
L5

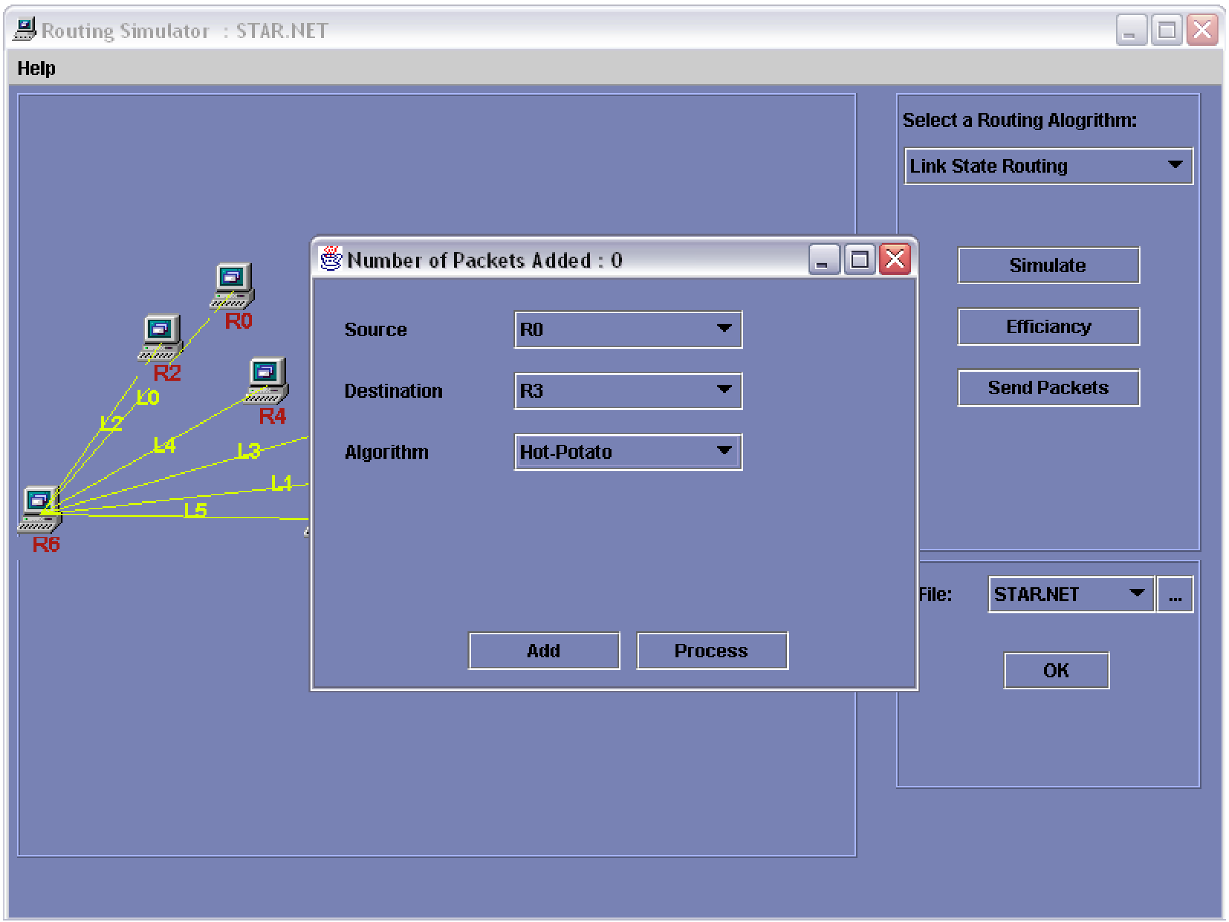

Routing Simulator : STAR.NET
Help
RIP->Source :R4
Destination
NextHop(s)
Cost
R0 R6 2
R1 R6 2
R2 R6 2
R3 R6 2
R5 R6 2
R6 R6 1
elect a Routing Alogrithm:
outing Information Protocal
Simulate
Efficiancy
Send Packets
File:
STAR.NET
...
OK
R0
R2
R4
R6
L0
L2
L4
L3
L1
L5


Routing Simulator : STAR.NET
Help
Link State Routing-Source :R3
Destination
NextHop(s)
Cost
R0 L3 2
R1 L3 2
R2 L3 2
R4 L3 2
R5 L3 2
R6 L3 1
elect a Routing Alogrithm:
ink State Routing
Simulate
Efficiancy
Send Packets
File:
STAR.NET
...
OK
R0
R2
R4
R6
L0
L1
L2
L3
L4
L5

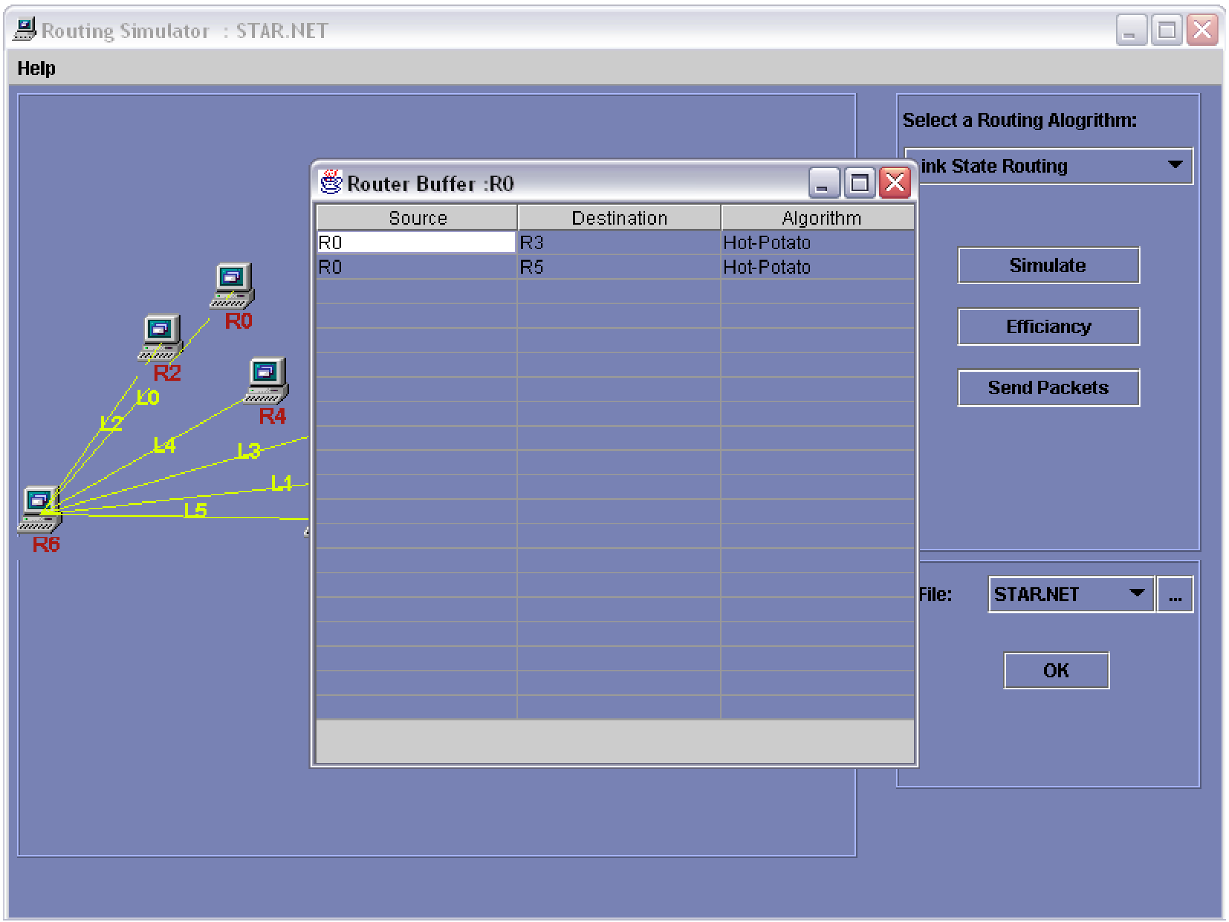

Routing Simulator : STAR.NET
Help
Router Buffer
Down
R2
R4
R3
R1
R6
R5
L0
L2
L4
L3
L1
L5
Select a Routing Alogrithm:
Link State Routing
Simulate
Efficiancy
Send Packets
File:
STAR.NET
...
OK

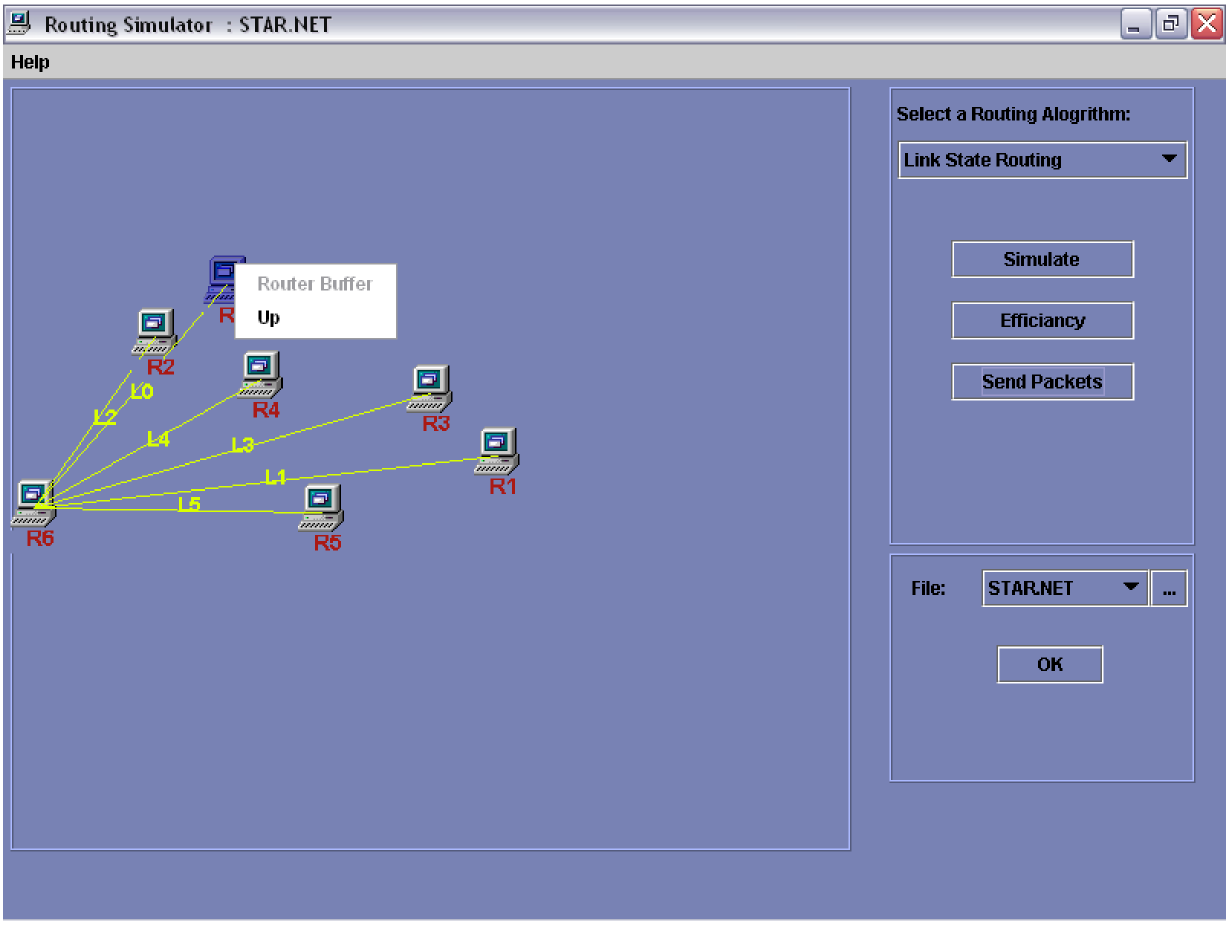

Routing Simulator : STAR.NET
Help
Select a Routing Alogrithm:
Link State Routing
Simulate
Efficiancy
Send Packets
File:
STAR.NET
OK
Algorithmic Efficiancy
Packet Information
Source
R0
Destination
R4
Via
R2
OK
R0
R2
R4
R6
L0
L1
L2
L3
L4
L5

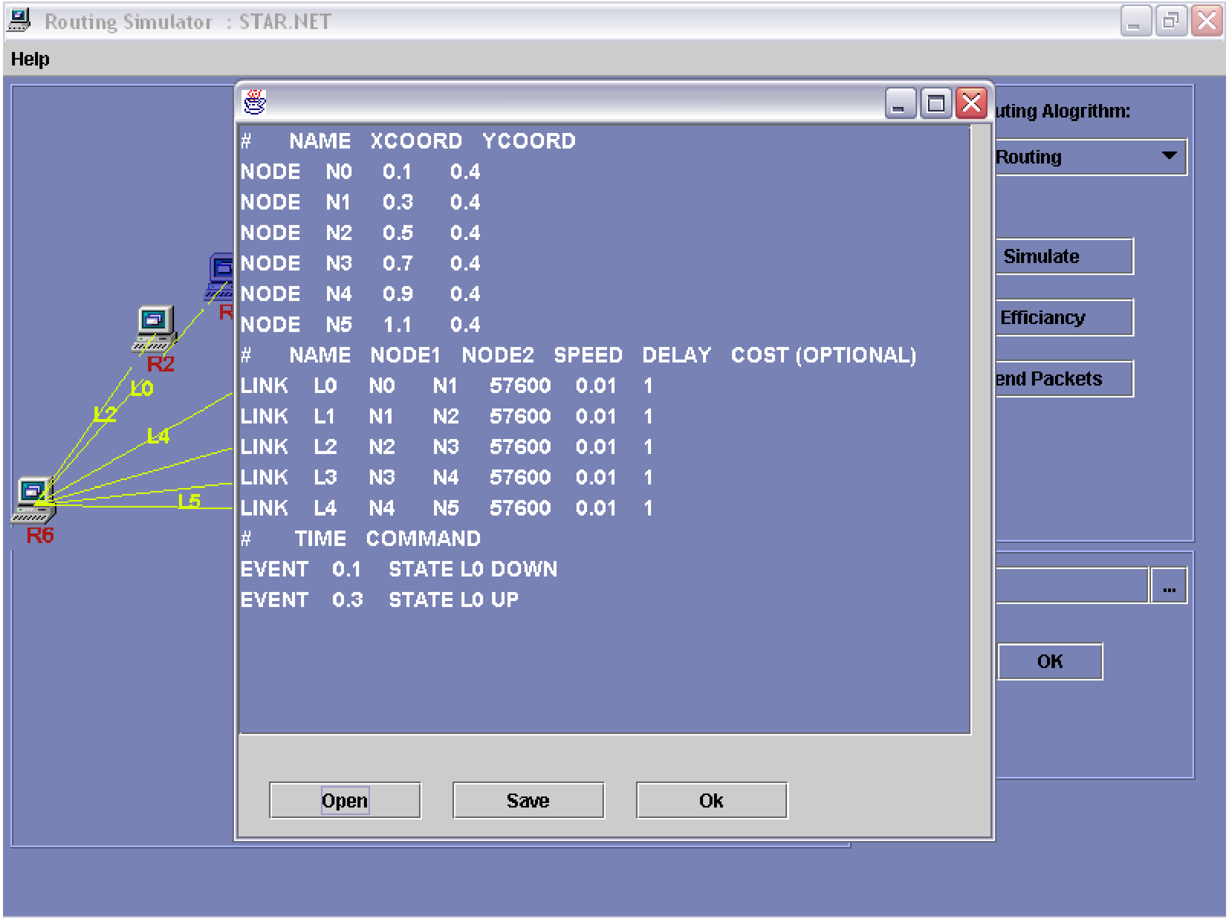

Routing Simulator : STAR.NET
Help
Select a Routing Alogrithm:
Link State Routing
Simulate
Efficiancy
R2
R6
L0
L2
L4
L5
Source :R0 Destination :R4 Via :R2
Algorithm
Efficiancy
Delay
Flooding
1.4285714285714286
Hot-Potato
1.4285714285714286
Source Routing
0.8571428571428571
Distence Vector Rou...
0.5714285714285714
0.02
Routing Information ...
0.5714285714285714
Link State Routing
0.5714285714285714
0.02
STAR.NET
OK



Routing Simulator : STAR.NET
Help
Select a Routing Alogrithm:
Link State Routing
Simulate
Efficiancy
Send Packets
File:
STAR.NET
OK
Number of Packets Added : 0
Source
R0
Destination
R3
Algorithm
Hot-Potato
Add
Process
R0
R2
R4
R6
L0
L1
L2
L3
L4
L5


Routing Simulator : STAR.NET
Help
Select a Routing Alogrithm:
ink State Routing
Simulate
Efficiancy
Send Packets
File:
STAR.NET
...
OK
Router Buffer :R0
Source
Destination
Algorithm
R0
R3
Hot-Potato
R0
R5
Hot-Potato
R0
R2
R4
R6
L0
L1
L2
L3
L4
L5


Routing Simulator : STAR.NET
Help
Router Buffer
Up
R2
R4
R3
R1
R5
R6
L0
L1
L2
L3
L4
L5
Select a Routing Alogrithm:
Link State Routing
Simulate
Efficiancy
Send Packets
File:
STAR.NET
...
OK


Routing Simulator : STAR.NET
Help
NAME XCOORD YCOORD
NODE N0 0.1 0.4
NODE N1 0.3 0.4
NODE N2 0.5 0.4
NODE N3 0.7 0.4
NODE N4 0.9 0.4
NODE N5 1.1 0.4
NAME NODE1 NODE2 SPEED DELAY COST (OPTIONAL)
LINK L0 N0 N1 57600 0.01 1
LINK L1 N1 N2 57600 0.01 1
LINK L2 N2 N3 57600 0.01 1
LINK L3 N3 N4 57600 0.01 1
LINK L4 N4 N5 57600 0.01 1
TIME COMMAND
EVENT 0.1 STATE L0 DOWN
EVENT 0.3 STATE L0 UP
Open
Save
Ok
uting Alogrithm:
Routing
Simulate
Efficiancy
end Packets
OK
R2
R6
L0
L2
L4
L5

Conclusion
And
Future Scope

Conclusion

The Simulator takes the configurations of the subnet as Input and gives the different statistics of the routers and links. By changing the routing algorithms and the different network configurations and recording the results we obtain the optimal algorithm. The optimal algorithm for a particular network is obtained by analyzing the results obtained. Simulation helps to achieve an optimal path that reduces the cost of routing.

The smaller networks can be analyzed and the results can be employed in larger networks to make routing efficient and economic. As the Simulator has provision for the crashing of routers, it gives an idea of which path is followed when a crash occurs. It can be employed in real networks to increase the performance of routers and links. As it not feasible in real networks to test algorithms and then implement a best one, **Routing Simulator** can be helpful. Hence it is useful for people who provide networking services and those who design networks.

Future Scope

This Routing Simulator model can be made a more realistic one by considering the effects of most of the System parameters.

Even though the mathematical model established for efficiency of Subnet yields acceptable results, I believe that an improved model can be generated.

This has the potential to be used as one of the tools for experimentation on design and analysis of Subnets.