

# Load Balancing on Multi-Link ATM Network

Steve Chen

University Undergraduate Research Fellow, 1994-95

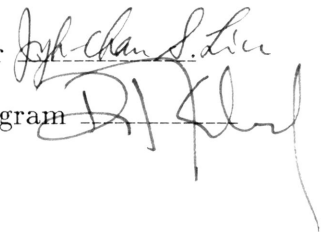
Texas A&M University

Department of Computer Science

APPROVED

Undergraduate Advisor

Exec. Dir., Honors Program

The image shows two handwritten signatures in black ink. The first signature is written over the text 'Undergraduate Advisor' and appears to be 'John Chan Salter'. The second signature is written over the text 'Exec. Dir., Honors Program' and is more stylized and difficult to decipher, possibly reading 'D. J. [unclear]'. Both signatures are written in a cursive, flowing style.

# Contents

Abstract

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>ATM Network and ATM Switch Models</b>	<b>3</b>
<b>3</b>	<b>Design Analysis</b>	<b>6</b>
3.1	Design Objective . . . . .	6
3.2	Performance Metrics . . . . .	8
3.3	Load Balancing Schemes . . . . .	9
3.4	Proof of Correctness . . . . .	10
<b>4</b>	<b>Hardware Design</b>	<b>12</b>
<b>5</b>	<b>Software Simulation</b>	<b>17</b>
5.1	Assumptions . . . . .	17
5.2	Simulation Parameters . . . . .	19
5.3	Operations . . . . .	20
<b>6</b>	<b>Performance Analysis</b>	<b>21</b>
6.1	Cell Loss Probability . . . . .	21
6.2	Delay Variance . . . . .	24
6.3	Link Utilization . . . . .	26
<b>7</b>	<b>Conclusions</b>	<b>29</b>
<b>8</b>	<b>References</b>	<b>30</b>

## **Abstract**

The purpose of this research project is to explore several possible solutions for the load balancing problem of multi-trunk ATM networks. In this paper, we will examine three different methods to balance the load of a multi-trunk ATM network, and these three methods include sender initiated load balancing, receiver initiated load balancing, and a hybrid of both methods. As part of this paper, we will present a hardware implementation to realize the load balancing mechanism, and we will also include a comparison of the expected performance improvements of different load balancing schemes using software simulation. The comparison of performance gain will be based on three factors. These factors are cell drop ratio, packet delivery time variance, and link utilization.

# 1 Introduction

As the explosion of modern technology drives the world toward a information revolution, the conventional information communication infrastructure is no longer sufficient to satisfy the demand for data communication requirements. As a result, there exists an urgent need for a high performance communication network. Based on the technologies currently available, the packet switching ATM network is the only plausible solution to satisfy these needs. However, even with the speed of 155 Mbps (mega bits per second) offered by ATM, more bandwidth and data throughput are needed in order to anticipate the volume of network traffic expected to be introduced by the information super highway. In order to further increase the throughput of ATM networks, we can either try to design network devices with even higher speed or to parallelize the data transfer.

The first approach is to design and built very fast network devices that has the capacity to handle the data transfer rate requirements. Because of the high speed requirements, these devices tend to be very complex and expensive. As the speed and complexity of a network device increases, it is more susceptible to errors cause by the external environment such noise. Another disadvantage for this approach is that even if the technologies to build these devices exists; the switch may be too expensive for practical use.

Parallelization of the data transfer process is the other alternative. In this approach, we use multiple communication mediums (trunks) to interconnect ATM switches in an ATM network instead of using only a single trunk between switches. The rationale behind this approach is that we should have twice the data throughput if we double the numbers of trunks use to connect the network. However, this performance can only be achieve under ideal situations. In other words, in a multi-trunk network, some of the trunks may be overloaded while other trunks connecting the same two switches may be under utilized. Because of this

inherent problem associated with the multi-trunk network, a load balancing mechanism is needed in order to achieve maximum parallel data transfer and for this type of network to be cost-effective.

As a research project, we will focus our attention only on the parallel data transfer approach. The main objective is to provide fast and reliable data transfer with minimum cost. As part of this project, we plan to look at different issues regarding the multi-trunk ATM network. Then, we will design a load balance mechanism to utilize each trunk to its fullest extent. Also, as part of the project, we will test the feasibility of our design by software simulations.

## 2 ATM Network and ATM Switch Models

The load balancing mechanisms and the simulation program that we designed and implemented is based upon the following models of the ATM network and the ATM switch.

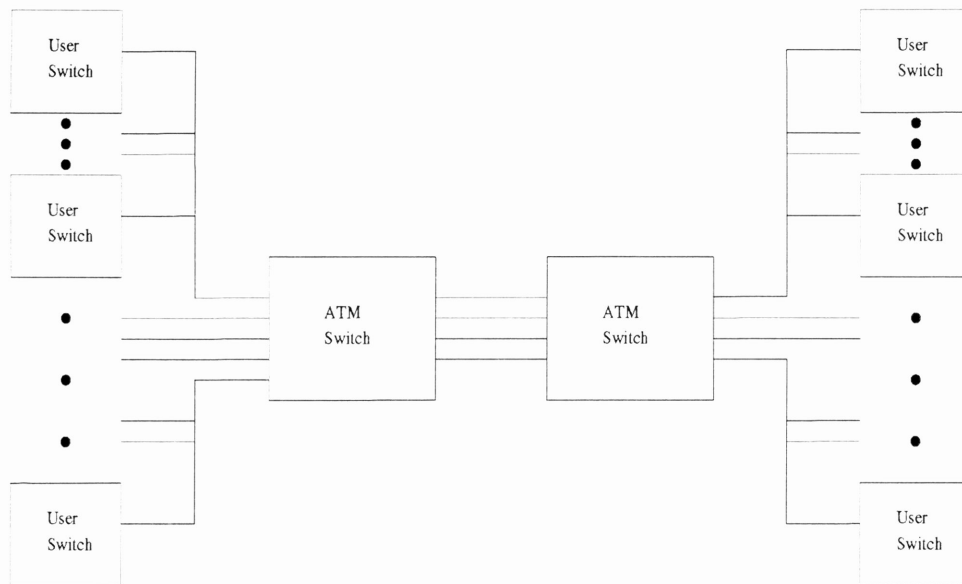


Figure 1 Multi-Trunk ATM Network Model

The network model shown in the above figure contains three basic components. These components include user switches and ATM switches connected by optical fibers. Using this model, we define the purpose of the user switch as a device used to generate, send, and receive ATM packets. In another words, user switches can be thought as individual computers on the network that communicate with each other.

The second component in the figure are the ATM switches. The role ATM switches are to direct the network traffic and to ensure that data reaches its correct destination. Lastly, optical fibers are simply the medium that the switches use to transfer data.

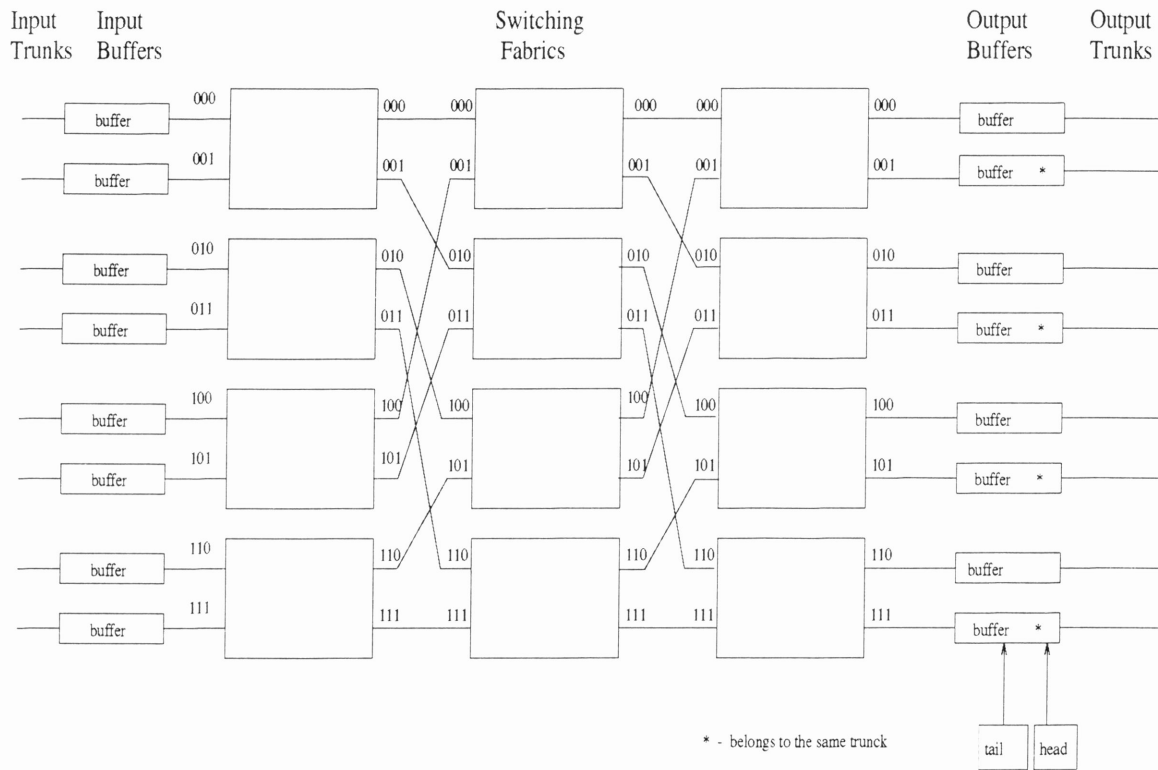


Figure 2 ATM Switch Model

In addition to a network model, we also have a model for the ATM switch. The ATM switch model we use also consists of three components - input buffer, switching fabric, and output buffer. The input buffer is a FIFO (first in first out)

queue used to temporarily store the ATM cells while they are waiting to be rerouted through the switching fabrics. The switching fabric is just a network of simple switches that allows an ATM cell to be transmitted from any input buffer to any output buffer. Finally, the output buffer, another FIFO queue, is used to store ATM cells while they are waiting to be transmitted to the next switch.

With the network model and the ATM switch model, we can identify the exact steps that an ATM cell has to follow in order to travel from the source switch to the destination switch. According to our network diagram, the network traffic (data to be delivered from one user switch to another) will be generated by user switches on the left. This traffic, generated in bytes, will be partitioned and inserted into the ATM cells which will be delivered to ATM switch 1. As soon as the ATM switch 1 receives the cell, the switch will queue the cell in the input buffer (figure 2). When the ATM cell reaches the front of the queue, the ATM switch will lookup its routing table, modify the ATM cell header, and reroute the cell to the proper output buffer. If the load balancing mechanism is activated, the destination output buffer will also depend on the loads of output buffers. From the output buffers of ATM switch 1, the ATM cell will then be delivered to ATM switch 2. The ATM switch 2 will go through similar process as the ATM switch 1 and send the ATM cell to the proper destination user switch.



## 3 Design Analysis

### 3.1 Design Objective

There are three design goals we try to achieve when we design our load balancing mechanisms. We wish to design and implement a product that meets the correctness, speed and cost constraints.

Correctness is the first design goal. Obviously, a wrong solution is worse than no solution at all. Correctness means ATM cells will reach its proper destination as well as maintaining the original order of send. Getting ATM cells to the correct final destination is not a major design problem. Since the header, which specifies the virtual path, is only defined between two switching points [3], any local changes, such as using another fiber to reach the next switch, will not affect the over all routing schemes of the packets, so ATM cells do not have any problems reaching their destination. The second issue we face is to maintain the order of the packets. For an ATM network without any load balancing mechanism, the packet order is always maintained. Because ATM uses virtual path connection and all of the ATM cells will follow the same path from source to destination. When we implemented the load balancing scheme, we actually forced some of the ATM cells to take a different physical path. In order to guarantee order, we have decided to reroute the entire virtual circuit instead of individual cells. With this scheme, we only have to worry about the very first cell or packet that has been rerouted. If we have a situation as shown in figure 3, where X is the last packet put on the queue from the same virtual channel, our load balancing mechanism will find the proper offset to insert the next cell in the proper location to maintain the packet order. Since we have rerouted the entire virtual channel, all the packets in virtual channel X will be routed to buffer 2 instead of buffer 1, so the order of all subsequent cells are maintained.

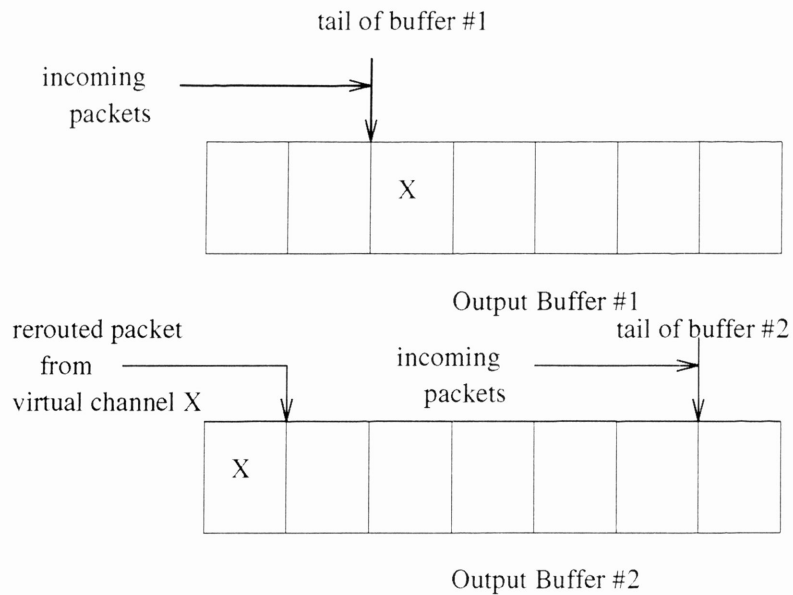


Figure 3 Rerouting ATM Cells

Speed is another major design issue. Because the speed of ATM network is so fast, the speed of the proposed load balancing mechanism must be fast enough to react to the change in network conditions. Due to the speed constraints, most of the load balancing mechanism needs to be implemented in hardware. Also, effort has been put forth to reduce the levels of logic that the signals have to travel in order to further reduce the switching propagational delays.

Our final design criterion is the cost to incorporate the load balancing mechanism. Because one of our original goal is to achieve high throughput by using multiple inexpensive ATM switches in place of a single costly ATM switch, the cost to implement the hardware becomes a major concern. In the design process, we have to carefully evaluate our hardware design in order to ensure that the hardware logics are both functionally correct and cost-effective.

## 3.2 Performance Metrics

The performance improvements of the load balancing mechanism is judged base on three criteria. These three criteria are cell loss probability, time delay variances, and link utilizations.

The first and most obvious measurement of the performance difference is the ATM cell loss probability. Minimizing the cell loss probability is important. When cells are dropped, data that are being transmitted through network is lost. In a multi-trunk ATM network ATM cells get dropped only when the load of a particular link gets so heavy that the output buffers eventually fill up. Since the sole purpose of load balancing is to redistribute the loads so that the traffic through each trunk is roughly the same. By equalizing the load and decreasing the chance of overloading any one trunk, we have made the trunks less susceptible to buffer overflow hence reducing cell loss probability.

The delay variance is another important factor which measures performance gain. Delay variance, simply put, is the time difference between reception of consecutive packets. It is an important characteristic especially for networks that are used to transmit voice data. Because the delay variance directly affects the pitch of the voice being transmitted, in applications such as video teleconferencing and other real-time voice communication applications; delay variance is extremely important. Large delay variance is usually related to high cell loss probability. As a result, one of the benefits derived from a well designed load balancing scheme is that the delay variances are minimized.

The third criterion of measuring the performance improvement is based on the link utilization. Basically, link utilization is the percentage of time that a particular link (communication medium) is actually used to carry data. Ideally, we want the utilization of each link to be roughly the same. If we distribute the load so that each link transmit about the same number of cells, we will minimize the cell loss

probability as well as the delay variance.

By measuring the performance difference of the multi-trunk ATM network based on the cell loss probability, delay variance, and the link utilization. We gain a comprehensive view of the behavior of a network with a particular load balancing mechanism. Also, because link utilization is a good indication of the load in the network, we can also plot cell loss probability and delay variance against link utilization to attain a graphical representation of the performance difference. By viewing the data two different ways, we obtain a much through understanding of the behavior of the network which, in turn, helps us in determining the feasibility of our current load balancing scheme as well as any future designs and revisions.

### **3.3 Load Balancing Schemes**

The objective of the load balancing scheme is to equalize the link utilizations, minimize both the cell loss probability and delay variance while maintaining the order of the cells being delivered. With these requirements and conditions in mind, we have designed a mechanism to reroute an entire virtual circuit based on three load balancing schemes. These three load balancing schemes are sender initiated load balancing, receiver initiated load balancing, and the hybrid of both methods.

The first scheme that we designed and implemented is the sender initiated load balancing (SILB). SILB is a scheme based on the idea that if the load of a link reaches a preset level, a control mechanism will detect this condition and find an alternate route to redirect the cells. For a link to be chosen as an alternative route, this link must satisfy two conditions. First, the link must connect to same destination switch. Second, this link have to have a load that is lighter than the link that activated the SILB. If more than one link satisfied these two conditions, the

load balancing mechanism will choose the link with the lightest load as the alternative route.

The second scheme we used is the receiver initiated load balancing (RILB). The RILB is activated exactly the opposite as SILB. RILB is triggered when the load of a buffer drops below a predetermined value. As soon as the RILB is activated, the link that is under utilized will request the load balancing mechanism for more network traffic. When the load balancing mechanism receives the request, it will search all alternative routes for links that have a heavier load than the link that initiated the load balancing mechanism. Again, if more than one alternative is found, the RILB will pick the buffer with the heaviest load. Then, it will reroute a virtual circuit from the link with the heaviest load to the link activated the load balancing mechanism.

Finally, the last scheme is the hybrid of both methods. For the SILB load balancing mechanism, the performance fell dramatically under heavy network traffic. The RILB, on the other hand was not effective at moderate loads. The hybrid scheme tries to combine the best of the two method and achieve the maximum performance improvements.

With SILB, RILB, and the hybrid schemes available to us, we are now ready to design and implement the hardware to realize these functionalities and the software to predict the behavior of the hardware to be designed.

### **3.4 Proof of Correctness**

**part #1 rerouted packets will always be delivered after the previous packet**

Let Q1 and Q2 be output buffers to the trunks. Let P1 be the packet to be delivered in the original route. Let P2 be the next packet which is to be rerouted.

proof by contradiction

assume P2 is in a position to be delivered before P1

for this condition to be true the following conditions must be met

1. P1 must be still in the output buffer.
2.  $\text{current\_load} < \text{load} - \text{bytes\_delivered}$
3.  $\text{min\_offset} < \text{load} - \text{bytes\_delivered}$

By looking at the pseudo code for inserting a rerouted packet, we can immediately verify that the first two conditions can be met without any problems.

Since  $\text{min\_offset}$  is always  $\text{min\_offset} = \text{load} - \text{bytes\_delivered} + \text{ATM packet size}$  and ATM packet size is 53.  $\text{min\_offset}$  will always be greater than  $\text{load} - \text{bytes\_delivered}$

Because all three conditions can never be met at the same time, our assumption must be false.

Therefore, P2 will always be delivered after P1.

## **part #2**

Based on the argument made in part #1, packets from the same source will always leave a switch in the order in which they come in.

So, these packets will always be put in the input buffers of the next switch in the proper order.

According to our assumptions, time delay for congestion within the switch is small enough to be ignored.

Hence, these packets should be placed in the output buffers in the order relative to their positions in the input buffers.

Since the relative order is maintained, this load balancing scheme should work.

## 4 Hardware Design

According to our design, the hardware will involve congestion detection, activation of the load balancing mechanism, and provision of network status. All of information gathered by the hardware will interface with softwares in order to perform tasks such as rerouting ATM cells and updating the routing table. In this section, we assume some softwares already exist to interface with our hardware. As for the hardware implementation, we will discuss the hardware that will handle load balancing for four links as an example.

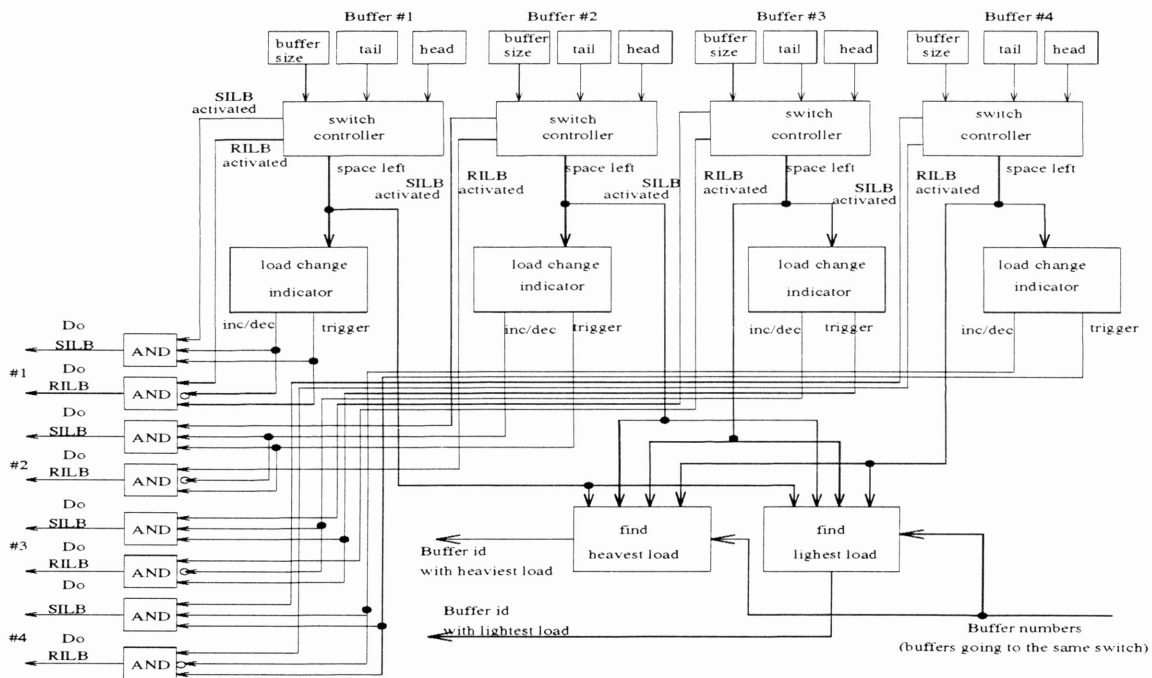


Figure 4 General System Block Diagram

The circuit shown in the figure 4 is the general block diagram of a load balancing mechanism circuit. The purpose of this circuit is to take the buffer size, buffer number, head pointer, and the tail pointer of a buffer and determine whether the load of this particular buffer is acceptable with respect to other buffers in the circuit. If the load gets out of the desired range, this circuit will request the appropriate load balancing scheme to be activated and supply the target buffer with the information required to perform load balancing. The circuit above consists of four major components - switch controller, load change indicator, and target buffer locator.

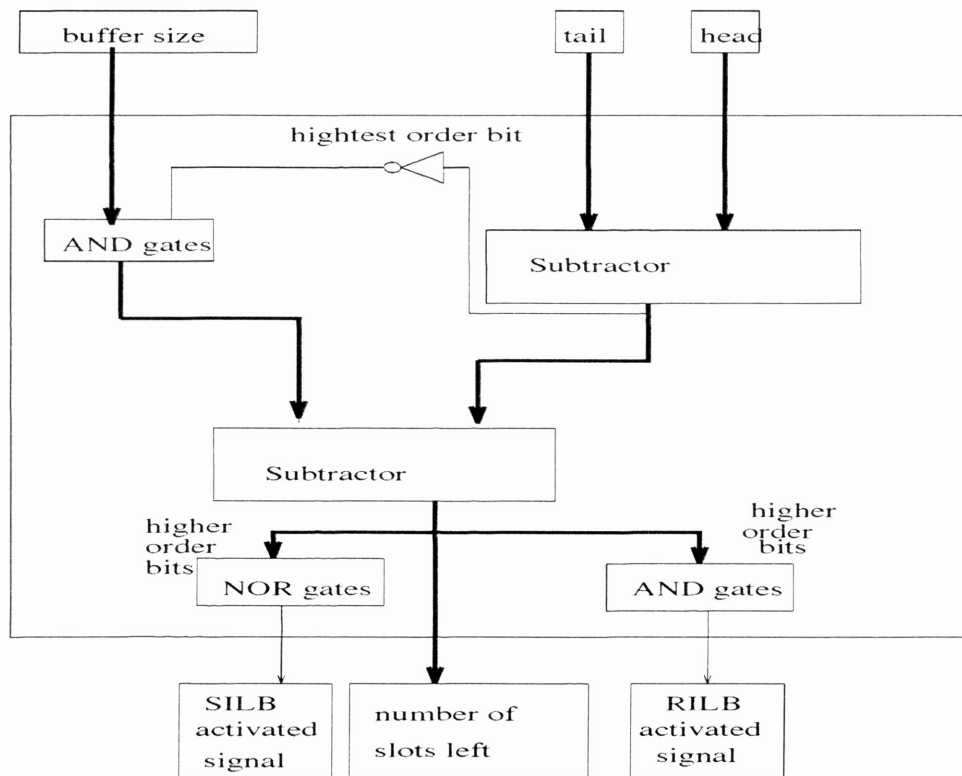


Figure 5 Switch Controller



The switch controller circuit (shown above ) performs three major functions. This circuit is responsible for generating the control signals to enable both SILB and RILB as well as calculating the amount of space left in the buffer. The implementation of this circuit is quite straight forward, it takes the buffer size and values of the head and tail pointer as the input. Normally calculating the amount of space left in the queue involves simple subtraction of the head from tail. However, because we implement our buffers as circular queues, it adds some complexity to the circuit. In order to take the wrap around of the queue into account, we added the left side of the circuit. If the difference between tail and head is negative ie the sign bit is 1, this section will add the buffer size to the difference to obtain the correct result. On the other hand, if the difference is positive, the sign bit will be 0 which also gives us the correct result.

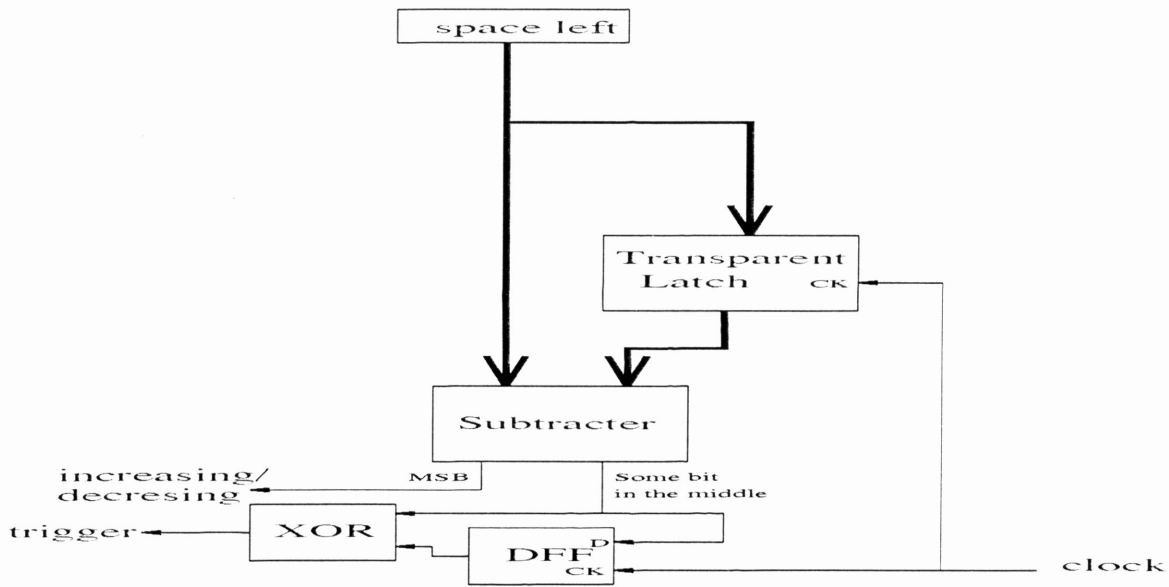


Figure 6 Load Change Indicator

The load change indicator circuit is the circuitry that activates the load balancing mechanism. The load change indicator takes the number of memory spaces left from the switch controller and the system clock as input to generate two control signals. The control signals generated consists of a trigger signal and a select signal. The trigger is simply a signal indicating a load balancing mechanism, either SILB or RILB, should be activated, and select is the signal that determined which load balancing mechanism to activate. The select signal comes from the sign bit of the subtractor. This circuit is designed to detect the change in load. In other words, if the space we have left now is smaller than the number of space we have previously stored; the sign bit will be set to one, which indicates the load is increasing. If the same logic is applied to the situation when current is larger than the load, a zero will appear on the select line to indicate that RILB should be activated. The trigger signal is generated in a similar fashion. Since we want the trigger to be generated at regular load intervals of the output buffer. We can use an exclusive OR gate and a D flip-flop (shown in figure 6) to detect any changes that may be of interest. Of course, the choice of which bit to use from the subtractor depends on the frequency at which the designer wishes the load balancing mechanism to be activated.

This circuit is designed to find the buffer with the heaviest load. This circuit will take the number of spaces left and their respective buffer number as inputs to the circuit. Then, the number of spaces left for each buffer will be compared with each other using a subtractor. By observing the sign bit, we can determine the buffer with smallest space left as the buffer with greatest load. One of the problems that we may encounter is that more than one of the buffers may be selected. If this situation should arise, there is a built in priority circuit to choose the buffer with the smallest buffer number as the output.

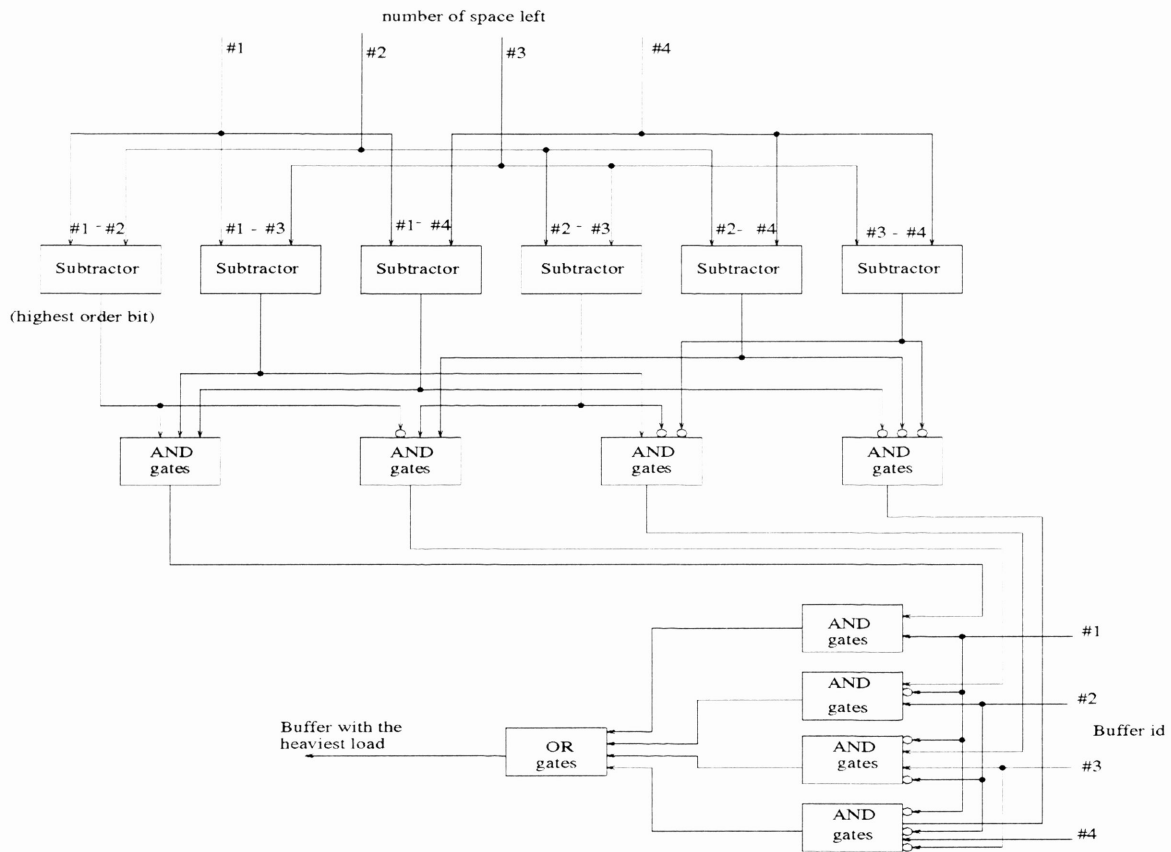


Figure 7 Priority Comparitor

The circuit used to find the lightest load is very similar to the circuit in the above figure with the exception that all the inputs to the *AND* gates from the subtractors are inverted. In principle, the two circuits functions exactly the same.

## 5 Software Simulation

The purpose of this multi-trunk ATM network simulation program is to obtain a better understanding of the behavior of different load balancing schemes under certain controlled conditions. In the following text, we will focus on the assumptions, the parameters, and the operations of the simulation program without going into implementation details.

### 5.1 Assumptions

In order to develop this network simulator, we have to make several assumptions regarding to the networks that we are simulating in order to make the design and implementation of the simulation software manageable. During the process of making these assumptions, we have spent considerable time and effort to avoid making assumptions that would compromise the accuracy of our simulation.

Assumptions concerning the components of the network to be simulated are among the first decisions we made. For the purpose of simulation, we define the ability to generate, deliver, and receive data as three essential characteristics that the simulated network should exhibit. From these three characteristics, we identify two types of network components that we need in order to run the simulation. A fundamental characteristic that a simulated network should have is the ability to send and receive data, and that is exactly what the first component does. The first component, simulated by the software, is the user switch. User switch is a component that is able to generate and receive data. These components can be thought of as hosts connected to a network. The next essential characteristic of the network is the ability to correctly deliver the data from the source to the destination which describes the function of the second component, the ATM switch. The purpose of the ATM switch, once again, is to deliver data correctly from one user switch to another user switch via virtual channel connection[3].

The next class of assumptions we made concern the routing scheme of the ATM network. First, we assume each user switch has the necessary information to place in the ATM cell header, so each cell contains enough information to eventually reach the correct destination. Another assumption we made concerns the routing table in the ATM switches. Since various algorithms to generate these routing table is a research topic of its own, we made the assumption that the routing tables are available at the beginning of the simulation, so the routing tables will not be generated by this simulation program. Congestion within the ATM switch is another major topic of research [4]. In the case of our simulation, we will take measures to minimize the internal congestion of the ATM switch itself, but for the purpose of this simulation. This effect will be largely ignored. Finally, the last assumption we made pertaining to routing scheme is the buffering scheme of the switches. For this simulation, we assume leaky bucket buffering scheme for all the queues implemented.

We also made some assumptions in regard to the type of traffic to be simulated on the network. Because different types of network traffic has different burst characteristics, the type of data going through the network will directly impact the performance of the network. Since one of the main applications of ATM networks is for real time audio and video transmission [5], we made the assumption to let the traffic flowing through our network exhibit the traffic patterns of MPEG and JPEG [1]. By making this assumption, we have to incorporate the traffic patterns of a useful application into our simulation.

Finally, the last set of assumptions deals with time and timing issues. The first assumption we made regard to timing is the definition of a time unit. Since the simulation and the performance are based mainly at the cellular level, there is really no need to deal with the timing, mutual exclusion, and other physical level details. Based on this reason, we have define the time to transfer 53 octets or 1 ATM cell to be the basic time unit. Processing speed of the user switches and ATM switches is another major issue. Because a network consists of a heterogeneous set of processors,

each processor could potentially be running at different speed. In order to mimic the network as closely as possible, we have assume that each user switch will generate and send packets at different rate. The ATM switches in our simulation program, on the other hand, use the SONET standard to communicate with each other. For this reason, we have assumed that all ATM switches are synchronized.

## 5.2 Simulation Parameters

In order to obtain meaningful results, the simulation program must be able to simulate numerous different situations in a control matter. In order to achieve this objective, this network simulator is designed to run based on the conditions defined in an input data file. To increase flexibility, another program is developed to generate the input file in order to provide a wide range of conditions for the simulation program to run. Basically, the parameters defined by the input file can be divided into network physical layout information and traffic pattern information.

The parameters used to define the network physical layout includes information such as the number of user switches and ATM switches, the interconnections between these switches, and the internal structure of the switches. For the number of user switches, we use a random number generator. Because the number of user switches will directly affect the amount of traffic flowing through the network, we can test the network at different load levels by varying the number of user switches on the network. The number of ATM switches, on the other hand, will not vary so we can use a consistent configuration to measure performance differences. The configuration of a network deals with how the different switches are interconnected. The major issue for the configuration is to provide consistency and not variety, so a predetermined function is used to generate this information. The last parameter that determine the physical layout of the network is the internal structure of the switches. These parameters include the number of input and output links a switch

has and the size of the buffer used by these communication links.

Another category of parameters are used to define the characteristics of the traffic. One of the parameters that determine the traffic patterns is the virtual circuit connections between two user switches. The simulation program uses this information to build routing tables in the ATM switches and to provide user switches with information so that they can send their cells to the desired destination. In order to add more “randomness” to the network, the input data file includes information such as the prefer output buffer number in the ATM switch and data rate of each user switch. A user switch can either not generate any packet at all, or it can generate network traffic based on the traffic patterns of MPEG and JPEG [1] applications. The way this traffic pattern is generated is based on the data rate and the duration of that rate. In the simulation program, both parameters are also generated by the random number generator.

### **5.3 Operations**

The way to execute this simulation program and to obtain the performance results is quite strait forward. A PERL script is used to simulate a large set of simulations at one time. During each simulation, the PERL script will simulate several hundred different situations based on the parameters set in the input data file which is dynamically generated by another program. Because we need to compare the results of different load balancing schemes, the simulation program will execute four times for each set of input data. The simulator will first execute with the load balancing mechanism turned off. Then, it will simulate, using the same parameters, with only SILB, only RILB, and finally the combination of both. The results of these simulations will be stored in a set of output files which will be analyzed by another shell script at the end of the entire simulation execution.

## 6 Performance Analysis

In order to better describe the behavior of the multi-trunk ATM network, we will examine the results of our simulation from two different perspectives. First we will analyze the simulation results in terms of cell loss probability, delay variance, and link utilization. Based on the text output of these three performance criteria, we should have a clear understanding of the behavior of the network. As for the second perspective, we will look at the performance data in a graphical representation. In this step, we will plot the cell loss probability and delay variance against link utilization. By examining these graphs, we should have a better understanding about performance difference of various load balancing schemes under different load conditions.

### 6.1 Cell Loss Probability

Cell loss probability is the first criterion that we use to measure the performance gain. In the table below, the cell loss probability of different load balancing schemes are listed in the order of increasing loads. The amount of network load or traffic is directly related to the number of user switches used in the simulations, which is listed at the left most column of the table.

According to table 1, RILB and the hybrid have dramatic performance improvements over a network without any load balancing. While the performance improvement of SILB is significantly less, it degrades dramatically under extremely heavy network load. Even under these conditions, SILB is still showing approximately a 30% decrease in cell loss probability.



user switches	<i>Hybrid</i>	<i>SILB</i>	<i>RILB</i>	<i>None</i>
120 or less	0.0000	0.0008	0.0001	0.0108
121 - 140	0.0004	0.0117	0.0009	0.0515
141 - 160	0.0048	0.0311	0.0051	0.0820
161 - 180	0.0159	0.0560	0.0171	0.0958
181 - 200	0.0315	0.0729	0.0320	0.0951
201 - 220	0.0451	0.0845	0.0453	0.1161
221 - 240	0.0479	0.0869	0.0482	0.1127

Table 1 Cell Loss Probability

When we plot the cell loss probability against the relative load of the network being simulated (link utilization). This difference becomes obvious. The curve represent the cell loss probability of no load balancing at all (line labeled as *none*), again, is use as a bases to compare the performance difference of the three different load balancing methods.

The first curve is the cell loss probability of SILB with various load. According to the graph, SILB performs very well under light to moderate loads. However, once the load gets sufficiently high. The performance of SILB degrades dramatically in terms of cell lost probability. The reason for this is simple. SILB is activated based on the load of the output trunks which is determined by looking at the amount of data in the output buffer. SILB is implemented to be activated above a preset threshold and triggered at fixed intervals based on the changes in the output buffer load. Specifically, in the simulation program, the SILB will be activated when a output buffer reaches 50% of its total capacity, and the ATM packets will be rerouted at increments of 10% of its capacity above this threshold. In other words, SILB will rerouting packets when the output buffer load reaches 50%, 60%, 70%,

... When the load is relatively light, SILB is able to keep the loads of different trunks balanced by reroute the entire virtual circuit from a link with higher loads to an idle link. As the number of virtual circuits increases, the change in network traffic also increase. When the rate that the traffic changes reaches a certain point, SILB will not be able to keep up with the changes in network traffic. As a result, we can see a sharp increase in cell loss rate at this critical point.

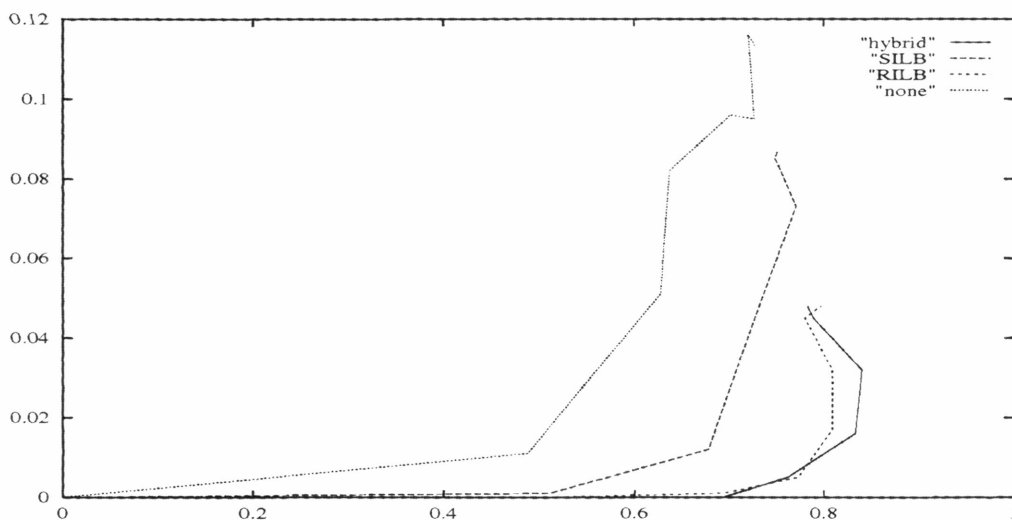


Figure 8 Cell Loss Probability vs Link Utilization

The cell loss probability of RILB is represented by the next curve. As we can see, RILB out performs both SILB and no load balancing at all conditions. A different approach has been used to determine the conditions that RILB will be triggered. In the simulation program, RILB is at an inactive stage when the heaviest loaded output link is operating under 85% capacity, and the lightest loaded

output link is working above 40%. These numbers are chosen arbitrarily, but if all buffers are operating under these conditions, the load is relatively balanced. As soon as any output buffers reaches over 80% of its capacity, every output buffer that is working below 40% capacity will request more packets from the more heavily loaded link. Therefore, a heavily loaded link can potentially have several of its virtual circuits rerouted to other output links. Since there is a well defined floor (40% capacity) and ceiling, (80% buffer capacity in this case), the output buffers tend to be forced to operate within this range. Under most of the conditions this type of load balancing performs very well. There is, however, a draw back to this method of load balancing. In the RILB, there is a delay before a packet is rerouted. Packets may be dropped during the rerouting process.

Finally, the last curve represents the cell loss probability for the hybrid of both SILB and RILB. It is no surprise that the hybrid method out performs both SILB and RILB in terms of the low probability of a packet being dropped. The reason for that is obvious. When the load is light, the RILB portion of the load balancing is very much idle, and everything is handled by the SILB. As soon as the load reaches the point that SILB can no longer efficiently handled the rerouting of the packets, the RILB takes over and begin doing most of the load balancing.

## 6.2 Delay Variance

The delay variance, as mentioned earlier in the paper, is the time difference between the arrival of ATM packets, and it can be directly impacted by three different factors. The first happens when a cell is dropped. Since the ATM cells are delivered periodically, the delay time will fluctuate when packets are dropped. The second factor that affect the delay variance is a change in network traffic. If the traffic pattern of the network fluctuates greatly, this will also be reflected in the values of delay variance. Finally, the third contributing factor is simply the number

of times a virtual circuit get rerouted. Since we need to insure the packet order, an artificial delay has been used to accomplish this objective. Due to this design constraint, the number rerouted circuits will have a direct impact on the delay variance. When we compare the results shown in Table 2, interestingly enough, the performance in terms of delay variance for each load balancing scheme is essentially the same.

user switches	<i>Hybrid</i>	<i>SILB</i>	<i>RILB</i>	<i>None</i>
120 or less	0.124	0.120	0.123	0.135
121 - 140	0.174	0.166	0.193	0.219
141 - 160	0.233	0.253	0.234	0.315
161 - 180	0.326	0.293	0.299	0.360
181 - 200	0.398	0.353	0.393	0.384
201 - 220	0.614	0.593	0.609	0.642
221 - 240	0.674	0.671	0.701	0.701

Table 2 Delay Variance

However, if we plot the delay variance against the link utilization, we begin to see the performance difference for each load balancing scheme.

Similar to the performance difference in terms of cell loss probability. The performance plot of the delay variance and link utilizations exhibits similar graphical behavior. Specifically, the delay variance graph is relatively flat until the link utilization reaches a certain point. Beyond this critical point, there is a sharp increase in terms of delay variance. The significance of this graph is that it gives us an indication of the maximum amount of traffic that each load balancing scheme can handle.

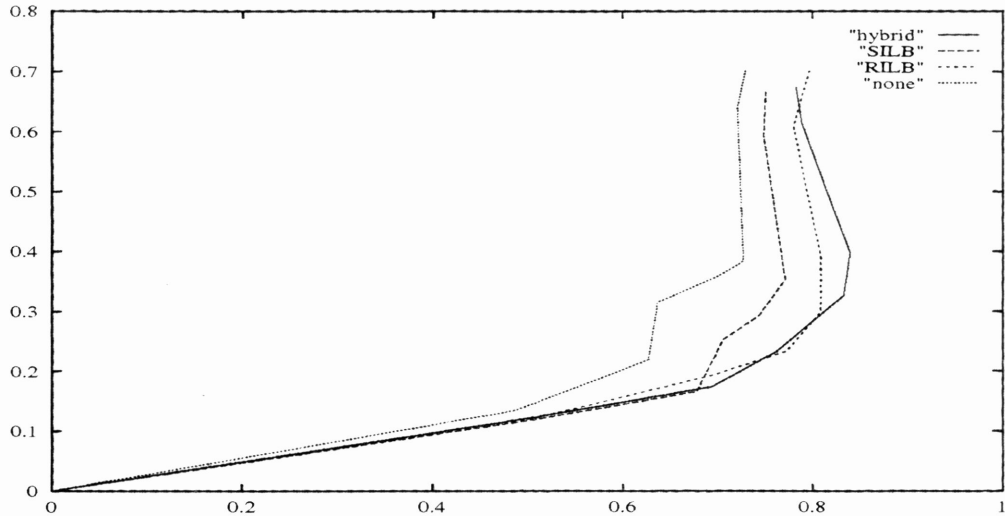


Figure 9 Delay variance vs Link Utilization

According to the above graph, the performance of a multi-trunk ATM network without any load balancing scheme started to decrease rapidly at about 70% link utilization. With SILB, we are able to increase the link utilization to about 75%. With RILB, the load that the network is able to handle increases another 5% to about 80%. Finally, in the hybrid of SILB and RILB, we can handle yet another 5 percent of additional network traffic. By analyzing the data graphically, we are able to determine a significant increase in performance in terms of delay variance.

### 6.3 Link Utilization

The link utilization of the network tells us two things about the network. First, under light to moderate load, the link utilization gives a good indication of the

network traffic load. This information is usually used in conjunction with other data gathered in the simulation to obtain performance comparisons under various network loads. The second piece of information that the link utilization gives us is the effectiveness of the load balancing mechanism. By comparing the link utilization of different load balancing schemes, we can obtain a rough sketch of the effectiveness of the different load balancing schemes under various load conditions. By analyzing the data obtained in table 3, we can further understand the performance of our load balancing schemes.

user switches	<i>Hybrid</i>	<i>SILB</i>	<i>RILB</i>	<i>None</i>
120 or less	0.5114	0.5107	0.5111	0.4881
121 - 140	0.6944	0.6782	0.6938	0.6266
141 - 160	0.7623	0.7056	0.7727	0.6371
161 - 180	0.8331	0.7438	0.8088	0.7020
181 - 200	0.8398	0.7710	0.8093	0.7271
201 - 220	0.7886	0.7493	0.7804	0.7213
221 - 240	0.7833	0.7508	0.7972	0.7293

Table 3 Link Utilization

The link utilization data shown in table 3 supplies us with information regarding to the effectiveness of different load balancing scheme under various load. In general, the data from the various load balancing scheme follow a similar trend. When the network traffic is light, we see a relatively low link utilization. This is simply because there is not enough network traffic to keep all of the links busy. As the network traffic increase, the link utilization increase as well. This trend continues until the link utilization reaches a critical point where the network traffic gets sufficiently high that the load balancing schemes can no longer function

effectively. At this point, we can expect to see the link utilization level off and sometimes it even drops slightly.

Based on the link utilization of different load balancing scheme simulated, we can conclude that any one of the three types of load balancing will give us large improvements over no load balancing at all. We also find that the load balancing schemes that forces each link to have a preset range of load is much more effective than trying to balancing the load at every small increments of load change. Namely, the result of our simulation strongly indicates that RILB and the hybrid method are much more effective than the SILB in terms of link utilization.

## 7 Conclusions

For this research project, our goal is to find a cost effective method to increase throughput of a computer network. We approach this problem by using multi-trunk ATM network and attempting to transmit network traffic from different virtual path in parallel. In order to further increase the performance of the network, we design a load balancing mechanism for the multi-trunk ATM network and test the feasibility of this design through simulations. Among the load balancing schemes under consideration are SILB, RILB and the hybrid of both methods. After the design and extensive software simulation, we have found that the multi-trunk ATM network with load balancing mechanism exhibits great performance improvements in terms of cell loss probability and limited improvements in delay variances and link utilizations. Base on the performance difference and the results we obtain from simulating multi-trunk ATM networks without any load balancing, we conclude that a load balancing mechanism must be in implemented in order to make the multi-trunk ATM network a feasible solution. This particular research project may be concluding; however, research in this area is far from over. There are many open issues that need to be addressed as the demand for faster networks are yet to be satisfied.



## 8 References

- [1] Song Chong, San-qi Li, and Joydeep Ghosh, "Predictive Dynamic Bandwidth Allocation for Efficient Transport of Real-Time VBR Video over ATM," *IEEE Journal on Selected Communications*, Vol 13, No. 1, January 1991, pp. 12-23.
- [2] Johna Till Johnson, "ATM Networking Gear: Welcome to the Real World," *Data Communications*, October 1993, pp. 66-86.
- [3] Jean-Yves Le Boudec, "The Asynchronous Transfer Mode: A Tutorial," *Communications*, May 30, 1991.
- [4] Jyh-Charn Liu, Kang G. Shin, Charles C. Chang, "Prevention of Congestion in Packet-Switched Multistage Interconnection Networks," *NFS Grant MIP-9203895*, May 6, 1994.
- [5] Jerome R. Cox Jr., Michel E. Gaddis, and Jonathan S. Turner, "Project Zeus," *IEEE Network*, March 1993, pp. 20-30.
- [6] Jim Kurose, "Open Issues and Challenges in Providing Quality of Service Guarantees in High-Speed Networks," *ACM SIGComm*, Nov 1992.