Extending a Control System Model for Rate-based Congestion Control

by

Stephen J. O'Halek

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May, 1996

Author ....[illegible]........
Department of Electrical Engineering and Computer Science
May 10, 1996

Certified by ....[illegible]........
James Mills
Visiting Scientist
Thesis Supervisor

Accepted by ....[illegible]........
F. R. Morgenthaler
Chairman, Department Committee on Graduate Students

Eng.

Extending a Control System Model for Rate-based Congestion Control

Stephen J. O'Halek

Submitted to the Department of Electrical Engineering and Computer Science on May 10, 1996, in partial fulfillment of the requirements for the degree of Master of Science

Abstract

The Asynchronous Transfer Mode (ATM) is a network standard currently under analysis and development. The ATM Forum, a consortium undertaking the majority of the development of ATM protocols, proposed guidelines for congestion control which include a rate-based control mechanism. Some of the early work in this field made use of linear control theory to develop rate-based congestion control algorithms. The models established in this early work are inadequate for networks comprised of multiple switch elements.

This thesis extends the model to more complicated networks. The effect of adding link delays to a one switch system is first analyzed. Then, the model for a network of switches is created. This model requires the representation of several dynamics not present in a one switch system. These dynamics result from the influences that networked switches have on one another. For example, the queue of any switch can impact the flow of cells throughout the network because queues cause delays and restrict the flow of cell traffic. Another previously unmodeled dynamic results from the way in which the rate-based congestion control algorithm is implemented. Each switch calculates a feedback rate based on its queue size to limit congestion in that queue. The cells of a source may traverse many switches in route to their destination, and contribute to congestion in each of them. In order to control congestion in the overall system, it is necessary for a source to change to the lowest rate that any switch in the network suggests to it. These new dynamics did not come in to play in the earlier work which did not examine a multi-switch system. A comprehensive model of these dynamics improves the ability to determine the behavior of the system. When all such dynamics are represented as part of the block diagram model of the system, the network conditions can be more accurately determined.

Thesis Supervisor: James Mills
Title: Visiting Scientist

Table of Contents

Chapter 1

Introduction

1.1 Background

The Asynchronous Transfer Mode (ATM) is a network standard currently under analysis and development. ATM was accepted as the preferred transfer mode for the broadband integrated services digital network (B-ISDN) by the International Telecommunication and Telegraph Consultative Committee (CCITT) in 1988[1]. B-ISDN is being developed as a high speed network capable of supporting a wide variety of applications ranging from voice to video all on a single network[5]. In ATM, all information entering the network must be fragmented into fixed size cells. The cells from various applications are multiplexed together based on their service requirements and the network resources available. In the beginning, the integration of various types of traffic was a primary focus during the development of ATM standards. At the present time, it appears that ATM will first be adopted in high speed network applications, such as interconnecting high speed LAN's. The majority of the current development of ATM protocols has taken place within the ATM Forum, a consortium consisting principally of equipment vendors attempting to accelerate the adoption of ATM products and services.

The traffic on an ATM network can be divided into two main classes: guaranteed and best effort. Guaranteed traffic has strict requirements on available bandwidth and delay in the network. An example of guaranteed traffic which has strict real time requirements is voice traffic. ATM supports guaranteed traffic with the constant bit rate (CBR) and variable bit rate (VBR) service classes. Best-effort traffic has less stringent network service requirements. The ATM service classes for this traffic are referred to as available bit rate

(ABR) and Unspecified Bit Rate (UBR) service. For this type of traffic, the available bandwidth is dynamically allocated among all the best-effort traffic[6].

ATM networks are connection-oriented. In order for a source to send a message over the network, it must first inform the network as to the type of traffic it wants to send. If the resources are available to meet the requirements of that traffic, then the network establishes a path from the source to the destination. This path is called a virtual circuit (VC) in an ATM network.

Traffic management is necessary to ensure that the network does not become congested due to excessive data cell flow into the network. The different classes of traffic in ATM networks require different approaches to traffic management. Guaranteed traffic is managed via admission control and traffic shaping. During connection set up, information concerning the bandwidth and delay requirements is provided to the network. The connection is refused if the network does not have the necessary resources available. After accepting a connection, the network periodically checks for user compliance with negotiated parameters. Since it is virtually impossible to establish strict service parameters for ABR traffic, a different method of traffic control is needed. A closed-loop feedback control scheme has been proposed for ABR traffic. In such a scheme the network relays congestion information to the data sources, which must then adjust their behavior.

The ATM Forum has selected a rate-based congestion control mechanism. In a rate-based scheme, data sources modify their data cell transmission rates based on feedback from the network. The ATM Forum has proposed guidelines as to how the network should convey congestion information to the sources and how the sources should react to information they receive. The Forum will not propose a fixed algorithm; rather, it will leave this up to the individual vendors. Sample algorithms have been provided as examples of possible implementations. The algorithms presented provide heuristic approaches. These algorithms contain significant non-linearities which makes detailed analysis difficult.

Figure 1.1 shows the relationship between a network and a source connected to that network. When the source has cells to send it transmits them to a destination through the network. The network infers the source cell rate from the queue size. It determines the congestion levels that occur due to the source cell rate, and sends rate control information back to the source via resource management cells. The source then has a maximum rate at which it may transmit cells in the forward direction, and the network provides feedback to control future source rates. Thus there exists a signal to control -- the rate, a method of observing the signal -- the queue size, and a feedback mechanism -- RM cells. Therefore, linear control theory can be applied to this system.

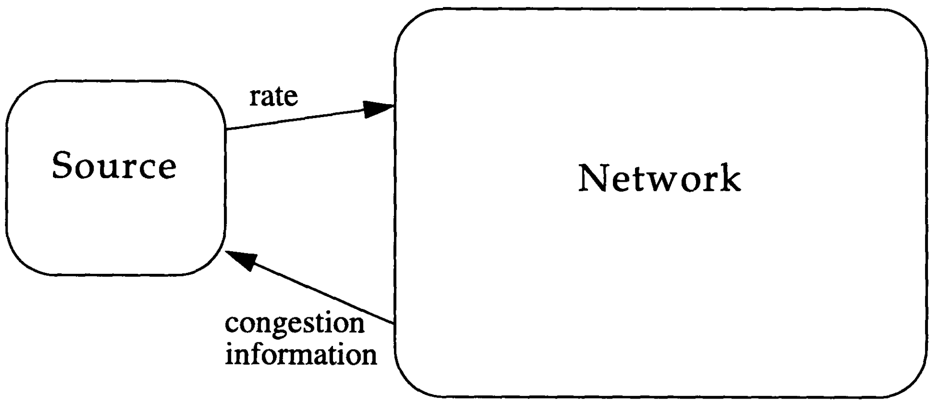

Figure 1.1: Rate-based feedback system

1.2 Goals of Rate-based Congestion Control

Four main guidelines must be considered in the development of a congestion control scheme for ABR traffic:

(1) Use available bandwidth to the fullest possible extent.

(2) Keep queue sizes as small as possible.

(3) Treat all sources "fairly."

(4) Keep the algorithm as simple as possible.

Often, these conditions must be traded-off of one another. Specific congestion control algorithms may emphasize some of these guidelines over others.

Utilizing the full amount of available bandwidth can reduce the probability of congestion. If a cell arrives at a queue that is full, some cell will have to be discarded. If the dropped cell is part of a packet from a higher layer protocol like TCP/IP, then the many cells which composed that packet must be retransmitted. This can lead to more cell loss and more congestion. A goal of ABR service is to ensure that the probability of a cell being discarded is low. Full utilization of the bandwidth helps to keep cell loss due to buffer overflow to a minimum by allowing the maximum amount of buffer space to remain free. If the network operates a rate lower than the maximum rate at any time when there are cells to service, then the average number of queued cells will increase, and therefore the probability of cell loss will increase. Keeping the queue sizes small also reduces the queueing delay that cells face, and therefore reduces the round trip delay in the network.

A congestion control scheme should allocate the available bandwidth fairly among all the users of the network. Fairness is often meant to indicate two things. First, all sources are entitled to an equal share of the bandwidth of the links they traverse. But they are restricted to the share determined by the most congested queue in the path of their cells. Second, sources are entitled to a quick response from the network. A given link is referred to as a bottleneck if the cell rate is limited by congestion at that link and not at any other link. The first point above requires that all bottleneck links at a given switch get an equal share of the available bandwidth. The second point may be a primary consideration for sources just entering the network, since it may take a while to receive a fair portion of the resources. There are other schemes for fair allocation of network resources, and what is to be considered fair is generally the subject of negotiation between the user and the network.

The high speeds at which ATM networks operate require the amount of processing per cell to be minimal. Development of a simple congestion control algorithm is essential in an ATM network. Congestion control schemes that require detailed statistical information to be kept are not acceptable. For example, maintaining statistics and separate queues for each source/destination pair would require excessive processing for an ATM system. This limitation reduces the number of algorithms from which to choose, and is one reason that the ATM Forum failed to adopt a credit-based approach in which the sources modify the number of cells they transmit into the network in response to congestion information relayed as credits.

1.3 Reference Work

Previous work made use of linear control theory to develop rate-based congestion control algorithms that meet the four main criteria discussed in the preceding section[3,10]. As is discussed in Chapter 2, the algorithms and structures developed by the ATM Forum contain significant non-linearities. These algorithms were modified only as necessary to achieve a simpler, more directly analyzable network model via classical control techniques[11]. A binary algorithm proved to contain limitations difficult to overcome while using linear control theory[10]. An explicit rate algorithm was developed that led to modeling and analysis of a one-switch system[3]. Unfortunately, the topics discussed in [3] and [10] do not include all of the modeling issues for a multi-switch system. In particular, they do not discuss link delays, interactions of source cells and the influences of switches on one another.

In this thesis, the work of [3] and [10] is extended to consider the effects of link delay, multiple switches and different source descriptions. In considering these issues, previously unmodeled effects are uncovered and related to the classical control theory model. At first, the network model studied earlier is further analyzed with the complication of link delay added to the model to test its robustness and to search for any unmodeled behavior. Once that model is fully understood, the system under analysis is expanded from one switch to a more realistic network of switches.

The principle tools used will be Matlab[1] and OPNET[2]. Matlab is used in the analysis of the system. It is used to create Bode plots and to predict the response of the system via equation-level simulations. Equation-level simulations are based on block diagram models constructed to approximate the behavior of the network. OPNET is a network simulation tool which is used to create discrete-event simulations of the system. OPNET results are the true or real outputs of the system with which the equation-level results are compared. These tools are discussed in greater detail in Appendix B.

1.4 Outline of Thesis

This thesis extends the use of control theory to model more complicated networks than previously investigated. To this end, the chapters focus on eliciting the dynamics of the system and converting them into equations that can be analyzed as part of the block diagram of a discrete-time control system. Chapter 2 reviews ATM congestion control schemes and establishes the basis for the work of this thesis. Chapter 3 explores the effect of adding link delays to the previously studied one switch system model. Chapter 4 examines the results achieved when switches are networked in the simplest possible way. Chapter 5 discusses the system behavior when switches are networked and more than one

1. Matlab 4.2, copyright 1984-1994, The Mathworks, Inc.
2. OPNET 2.5.A, copyright 1994, MIL3, Inc.

switch serves as the network entry point for one or more of the sources. Chapter 6 brings together all of the elements discussed in the earlier chapters to form a single, comprehensive model for analysis. A summary of the results and conclusions of this work and a discussion of future directions of research is presented in the final chapter.

Chapter 2

Review of ATM Forum Congestion Control Schemes

2.1 Background

The ATM Forum develops recommendations within which any proposed rate-based congestion control algorithm must operate. The Forum does not define a specific algorithm that must be followed. Rather, the Forum focuses on creating functional specifications with many possible implementations. This allows some flexibility in selecting a particular switch algorithm while guaranteeing the compatibility between different switch algorithms. The ATM Forum also tests and compares various rate-based congestion control schemes to check for the viability of proposed algorithms.

Two main approaches to rate-based control have been considered. These are the binary approach and the explicit rate approach. In the binary approach, each source receives one bit of information indicating whether or not congestion has been encountered in the network. In the explicit rate approach, the source receives feedback from the network that explicitly indicates the rate at which a source should operate so that it will not contribute to a congestion problem. The binary approach allows the algorithm to be simpler while the explicit rate approach achieves better performance.

Based on the information gathered thus far, the ATM Forum has established a set of general standards as a starting point for future endeavors[1]. Among the issues for which basic standards have been established are the manner in which congestion information is conveyed through the network and the role sources, destinations and switches play in the communication of information. The standard methods by which congestion information is communicated are discussed in considerable detail. Only general guidelines as to how the switch should react are provided since this may need to vary considerably based on the intended application.

Two ways of conveying congestion control information through the network have been developed. One method uses resource management (RM) cells. These special ATM cells are periodically generated by each source. Typically, one in thirty-two cells generated is an RM cell. They contain information concerning the current cell rate of their source and the minimum cell rate that the source will accept. The RM cells also contain a congestion indicator (CI) bit which is used to relay binary congestion information, and an explicit rate (ER) field which is used in explicit cell rate algorithms. The RM cells follow the same path through the network as data cells in the forward direction. When they reach their destination they are sent back to their source. As the RM cells flow through the network, switches can consult the information already contained in them as well as other local information such as the size of the local queues, and then determine what sort of information to enter into either the CI or ER fields.

The second method for communicating congestion information is applicable for binary schemes only. In this scheme, the explicit forward congestion indicator (EFCI) bit which resides in the header of all ATM cells is exploited. A switch can set this bit if it recognizes local congestion. The destination monitors all of the incoming data cells on a virtual circuit to see if the EFCI bit is set. If so, it notifies the source by using the CI bit in the subsequent RM cell.

The following sections explore ATM binary and explicit rate algorithms in greater detail. These sections more clearly describe the behavior of the source, switches, and destinations. A section of this chapter examines the aspects of the protocol that the ATM Forum has moved toward standardizing. The final section provides the conclusions of the earlier work which is the basis of this thesis.

2.2 Binary Algorithms

Binary algorithms were the first rate-based algorithms considered by the ATM Forum. In a binary algorithm the feedback consists of a single bit which indicates only whether or not congestion is present. A switch determines whether it is congested by examining its queue to see if it has exceeded a certain threshold. If congested, the switch communicates this fact to the source of data cells. The two methods considered were forward explicit congestion notification (FECN) and backward explicit congestion notification (BECN). In FECN, if congested the switch sets the EFCI bit in all data cells passing through it. The destination checks the EFCI bit of every cell that enters and if congestion is indicated, it sets the CI bit in the next RM cell returning to the data source. In BECN, the congested switch sets the CI bit in RM cells returning to congestion causing sources. This second method reduces feedback delay, but requires a more complicated algorithm.

The algorithms for how a source should react to feedback information have evolved over recent years. An early rate control scheme had the source increase its rate unless it received an RM cell with the CI bit set[5]. Only after receiving notification of congestion would the source reduce its rate. Serious problems arose as this scheme was developed. If the RM cells are delayed or lost, then the source would continue to increase its rate until it received the proper feedback information, worsening the congestion problem. It this situation were to persist, the network would collapse. To remedy this problem, the algorithm was altered so that the source would continually decrease its rate. When it received notification that congestion was not present, it could increase its rate. This is referred to as a positive feedback approach.

The ATM Forum is now considering a more complete structure based on the binary algorithm called the proportional rate control algorithm (PRCA)[6]. PRCA is compatible with either FECN or BECN and it uses positive feedback. In PRCA, the source is required to send an RM cell for every Nrm data cells that it sends. When each RM is sent, the source multiplicatively decreases its rate. If it receives an RM cell with the CI bit not set then the source removes the effect of the previous decrease and increases its rate by an additive factor. If the source receives an RM cell with the CI bit set then it takes no additional action.

The method of frequent rate adjustment used in PRCA makes an analysis of this algorithm difficult. Additive increases occur at a rate proportional to the rate of returning RM cells. Multiplicative decreases occur at a rate proportional to the rate that the source sends RM cells. The rates at which RM cells are sent and received need not be the same at any instant since returning RM cells were generated at some past instant when the source's rate may have been lower or higher. Further, the time instants at which adjustments in the rate are made depend on the rate itself. Since there is not a time scale on which increases or decreases periodically occur, any sort of detailed analysis of the system is a challenge.

A typical network is shown in Fig. 2.1. The sources, switches and destinations are connected by links that allow cell traffic to flow in either direction. The two sources always have data to send and they both begin transmission of cells at the same rate. Source cells enter the switch at separate input ports but compete for the same output port which has a bandwidth of 3.538×10^5 cells/sec. Cells are then passed on to the destination from the switch. Resource management cells are transmitted from the destination to the sources to control the rate of data cells entering the network. The details of the operations of the components of the network are further discussed in Appendix A.

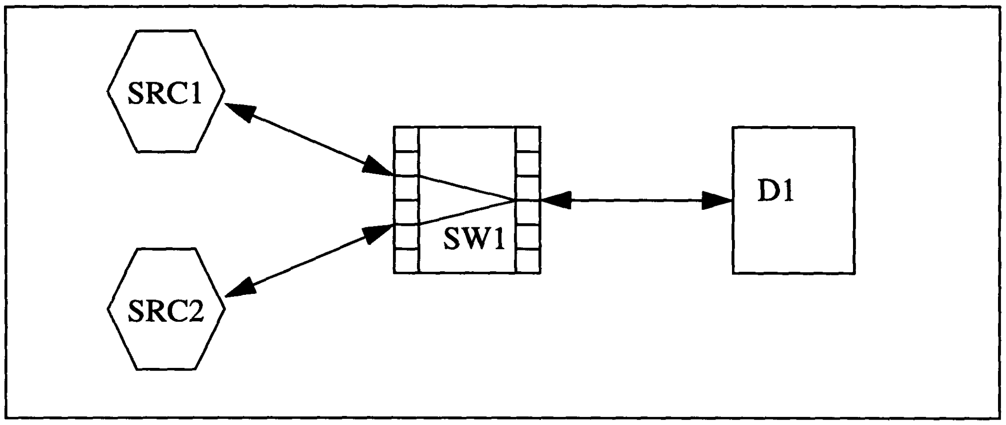

Figure 2.1: Network Topology for Simulation

Simulation results of a PRCA algorithm using BECN based on the network topology shown in Fig. 2.1 are displayed in Figs. 2.2 and 2.3. The parameters used in the algorithm are the same as in results recently presented to the ATM Forum[1]. Ideally, each source would be allowed to transmit at half of the total available bandwidth or 1.769×10^5 cells/sec. Rather than converging to this value, both sources oscillate approximately in unison between about 5.0×10^4 cells/second and 3.0×10^5 cells/sec. The threshold level for congestion to be indicated in the queue is set to 500 cells, and the queue oscillates between 300 and 600 cells. The amplitudes of the oscillations can be changed by tuning the parameters of the algorithm, but due to the non-linearities of this algorithm and the variable time scale, a detailed analysis of the effect of varying parameters is difficult.

The CI values in the RM cells received by source 1 are also shown in Fig. 2.3. Every time an RM cell returns to the source, the CI value is collected and a point is plotted. The CI values are actually equal to either zero to indicate no congestion or one to indicate congestion in the network. Here the CI values have been scaled by 700 so that they may be displayed along with the corresponding queue size. In this figure the CI values appear along the top and bottom of the plot due to the scaling. Comparing this plot to the rate plot reveals an interesting aspect of this algorithm. As the source reaches its peak attainable value, it is still receiving RM cells with CI=0 indicating no congestion, yet its rate levels off anyway. During this period, the source rate is high and new outgoing RM cells are being generated much faster than RM cells are returning. The result is that more of the slowdowns associated with sending RM cells occur than are being compensated for by the additive increases associated with receiving RM cells. This unexpected behavior stabilizes the system, but makes analysis difficult.

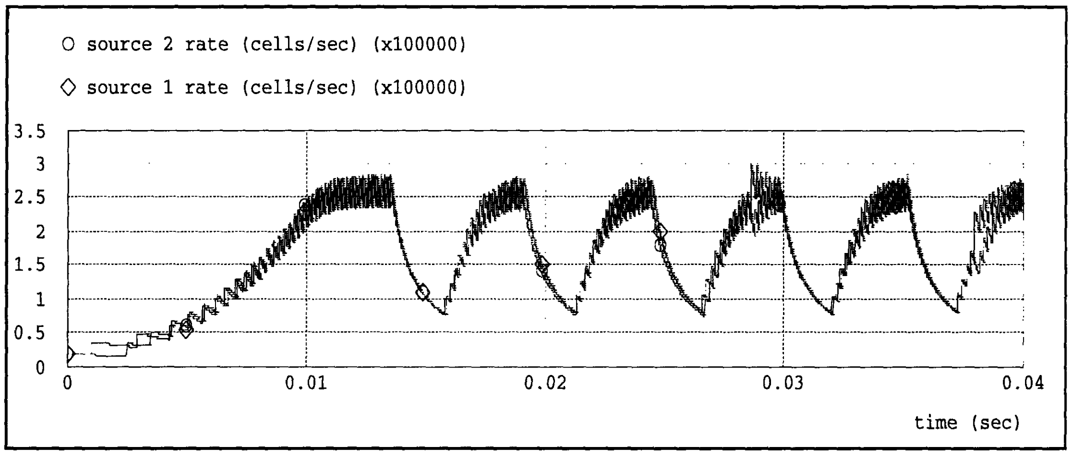

Figure 2.2: Rates of the two sources

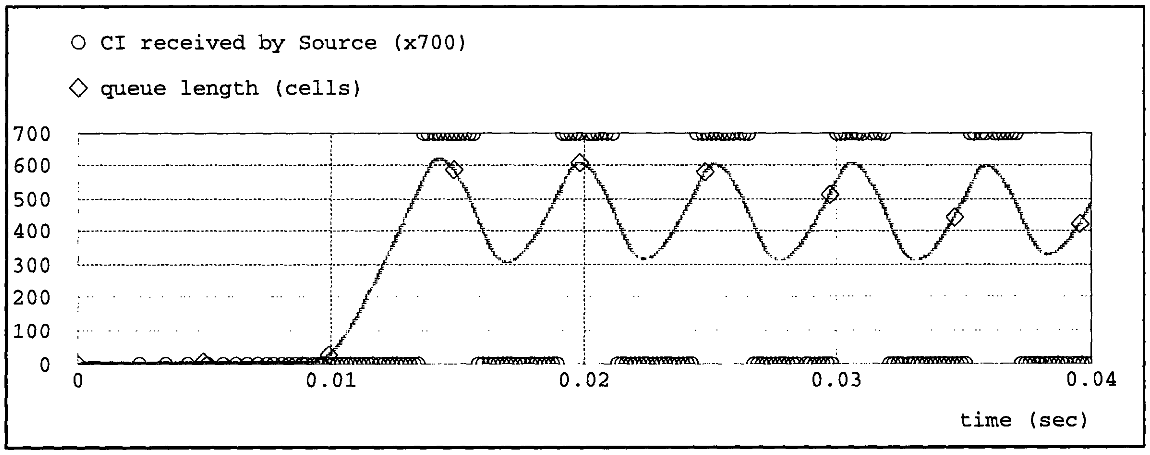

Figure 2.3: Queue size and congestion indication seen by source 1

If the rate changes are assumed to occur at fixed intervals, then it is possible to undertake some analysis of the system. To remove the unexpected contention between increases and decreases in rate discussed above, it is possible for the sources to keep track of the number of increases and decreases taking place during a given interval Δ. In this case the rate, $R[n]$, is given by

$$R[n] = (1-\alpha)^{C_1}R[n-1] + C_0\beta \qquad (2.1)$$

where C_0 is the number of RM cells with CI=0 received during the Δ time interval, C_1 is the number of RM cells with CI=1 received during the Δ time interval, α is the multiplicative decrease factor and β is the additive increase factor. An analysis of an algorithm similar to this one has shown that even with the rate increases and decreases better regulated, the system still exhibits steady-state oscillations similar to those shown in Fig. 2.2. A recent paper on the subject has shown that the cycles are fundamentally caused by the threshold non-linearity used to generate the binary feedback[8].

This non-linearity creates a control system which exhibits large limit cycles. When the queue size grows beyond the threshold level, the input rates will be larger than the service rate or output bandwidth of the queue. The queue will continue to grow until this situation changes. Once the switch enters the congested state, the congestion indicators will cause the source rates to begin to decrease, but the queue size will not decrease until the input rates fall below the output bandwidth. The cycle will then reverse. The two source rates are virtually synchronized in their oscillation since the non-linear mapping of the single queue is driving them simultaneously in the same direction. The oscillations become even larger when delay is added to the feedback loop.

In a more complex network, such as the one shown in Fig. 2.4, some sources may be treated unfairly with respect to others when algorithms like PRCA are used. In source/destination pair 1, the data cells have to traverse more switches than other source/destination pairs. Data cells from such sources are more likely to encounter congestion along their path than data cells which travel across fewer switches. Therefore, the RM cells from VCs going through more switches have a higher probability of being marked as contributing to congestion than those going through fewer switches. These sources do not receive a fair share of the bandwidth. This is known as the beat-down effect.

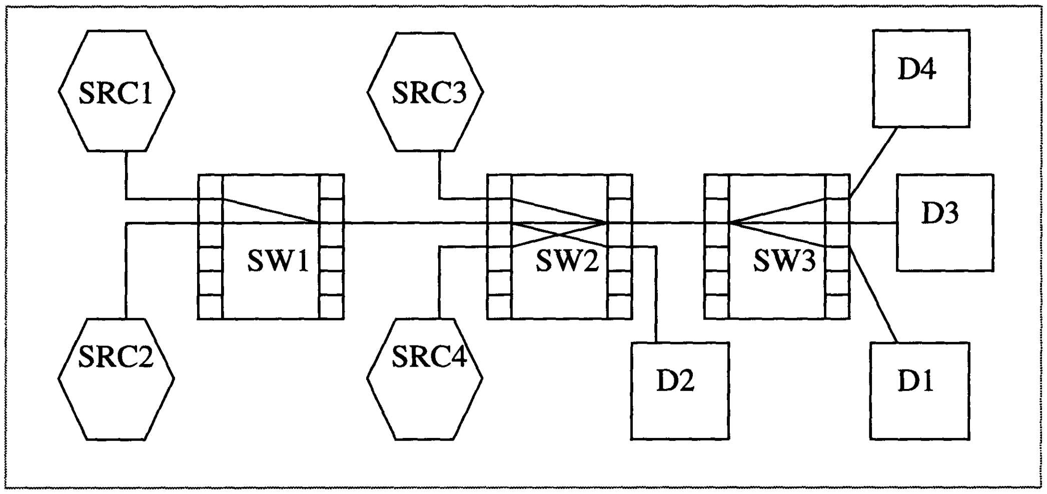

Figure 2.4: Larger Network Model for Illustration of the Beat-down Effect

In PRCA, the beat-down problem is due to the fact that the switches cannot differentiate between the different VCs when marking cells to indicate congestion. A proposed solution to the beat-down problem is to have the switch employ 'intelligent marking.' This means that a congested switch should only mark cells from those VCs that are using more than their fair share of the bandwidth. This requires that the switch determine the fair-share for each VC. If the algorithm is complicated so that the switch can calculate this information, it seems logical for this information to be fully utilized. To do this, a switch can explicitly communicate the calculated rate to the source. This is the goal of explicit rate algorithms.

2.3 Explicit Rate Algorithms

Binary algorithms can only indicate whether a source should increase or decrease its rate. This limits the speed with which a binary algorithm can achieve a fair allocation of rates and is the principle cause of oscillations in PRCA systems. Explicit rate algorithms are intended to allow more detailed information to reach the source. The switch explicitly communicates to the source the desired data cell rate. This allows the system to react more quickly and have less oscillations than the binary algorithms. The major concern in this case, as with intelligent marking, is how to calculate the fair rate to indicate to each source.

Charny developed one of the first viable rate-based algorithms[4]. This algorithm was shown to converge to a max-min fair rate allocation. This scheme calculated the fair rate using the following

$$\text{Fair Rate} = \frac{B - B'}{f - f'} \tag{2.2}$$

where B is the link bandwidth, B' is the bandwidth used by all VCs whose rates are less than the fair rate, f is the number of active VCs, and f' is the number of active VCs whose rates are less than the fair rate. The fair rate proposed by Charny's algorithm is the average rate of all the VCs which have not been constrained elsewhere. Therefore the local switch is the bottleneck for these VCs. If the cell rate of a source is greater than the fair rate, then the switch sends back the fair rate and the source adjusts its rate to that value. This algorithm requires that the switch store the rates of all active VCs. The computation of the fair rate also requires order n operations, where n is the number of active VCs. These considerations complicate the switch implementation, and a simpler algorithm is desired for ATM networks.

The enhanced proportional rate control algorithm (EPRCA) and a related algorithm proposed by Siu and Tzeng[12] both try to emulate Charny's algorithm while using fewer calculations and no per-VC accounting. These algorithms attempt to estimate the fair rate by an exponential average of the rates given in the RM cells entering the switch. This average is called the Mean Allowed Cell Rate (MACR). The Siu algorithm averages all the rates when the switch is not congested, but averages only those rates less than the current MACR when the switch is congested. EPRCA averages those rates greater than a fraction of the MACR (7/8 is suggested) when uncongested, and those rates less than MACR when congested. In the Siu algorithm, when a switch is congested it informs those VCs whose rates are greater than the MACR that they must reduce their rates to the MACR. In EPRCA, a value slightly less than the MACR is indicated to the sources to attempt to maintain the source rates at a level just under MACR.

When a fair allocation of rates is achieved, then the average of all the rates entering the switch is less than the average of all the bottleneck VCs. If the switch is congested, then this average must be too high, and so averaging only those rates less than the MACR decreases the average. If the switch is not congested, the rates are allowed to increase. As the rates increase, the MACR will also increase. In this way, the estimate of the fair rate oscillates around its true value. The EPRCA only averages those VCs whose rates are greater than a fraction of the MACR when uncongested. This is done with the assumption that those VCs whose rates are much less than the MACR are probably constrained elsewhere. These algorithms also contain a condition such that if the queue is very congested, then the rates are set to a fraction of the fair rate. Thus the rate drops rapidly if major congestion occurs. Simulation results of these schemes show that they solve the beat-down problem of the PRCA[12].

Other explicit rate schemes have been proposed to improve on the performance exhibited by EPRCA. An algorithm proposed by Jain[8] claims to reach its target rate ten to twenty times faster than EPRCA, but requires per-VC accounting. Another algorithm proposed by Barnhart[2], does not require per-VC accounting and does not oscillate in steady-state. Both of these algorithms set the target bandwidth slightly less than the available bandwidth. In this way, the queue sizes are kept near zero which helps speed up the system response. Both of these algorithms contain significant non-linearities, which makes any analysis of them difficult.

Jain's algorithm measures the time, T, until the Nth cell arrives at the output link of a switch. The input rate is then calculated as N/T. The target cell rate for that link is set slightly below the available bandwidth and used to calculate an *Overload Factor* in the following way

$$\text{Overload Factor} = \frac{\text{Input Rate}}{\text{Target Rate}} \tag{2.3}$$

where *Input Rate* represents the sum of the input rates destined for a specific output port. The *overload factor* is used to calculate an *explicit rate based on load (erbl)* for each VC. This is found as follows

$$\text{erbl} = \frac{\text{VC Rate}}{\text{Overload Factor}} \tag{2.4}$$

If the input rate is larger than the target rate, the *Overload Factor* will be greater than one, and dividing by the *Overload Factor* will reduce the rates. The opposite effect occurs if the input rate is low. The algorithm also requires that the switches compute a *Fair Share* for the VCs. This is calculated as

$$\text{Fair Share} = \frac{\text{Target Rate}}{N} \tag{2.5}$$

where N is the number of active VCs. The *explicit rate* sent to each source is the maximum of the erbl and the *Fair Share*.

$$\text{explicit rate} = \max(\text{erbl}, \text{Fair Share}) \tag{2.6}$$

This quantity must be stored for each VC and is sent back in all RM cells seen for a particular VC during the next calculation interval. The *Fair Share* is included in the algorithm to ensure that VCs that start low are able to increase their rates quickly.

Barnhart's algorithm also specifies a target rate that is slightly less than the available bandwidth and an input rate that is calculated as in Jain's algorithm. These are used to calculate an explicit rate for the queue (ERQ) by the following formula

$$\text{ERQ}[n+1] = \text{ERQ}[n]\left(1 + \left(1 - \frac{\text{Input Rate}}{\text{Target Rate}}\right)\text{Gain}\right) \tag{2.7}$$

The gain term is a constant less than one (suggested values are 0.1 -- 0.5). If the *Input Rate* is larger than the *Target Rate*, then this will cause ERQ to decrease, and if the *Input Rate* is less than the *Target Rate*, then ERQ will increase. The switch is also required to keep an exponential average of the rates coming in (MACR) as in EPRCA. The switch uses ERQ as the explicit rate that it feeds back as long as this value is between 0.75*MACR and 1.25*MACR. If ERQ is beyond of one of these bounds, then the bound is used instead. These bounds act in a similar role to the fair rate in Jain's algorithm.

Both of these algorithms represent a shift in emphasis for rate-based schemes. In previous schemes, the response was primarily based on the instantaneous queue length. When a certain threshold was crossed to indicate congestion, the algorithm responded to alleviate this problem. The latest algorithms base their feedback on the rate that data is entering the queue, and they try to keep this value lower than the available bandwidth. This helps to maintain queue length near zero and thus attempts to prevent congestion before it occurs.

2.4 Classical Control Model for a Single Switch

Recently, Rohrs, *et al.*[10] proposed a new perspective on rate-based congestion control. Rohrs couches the problem as a linear feedback control problem. The block diagram model that resulted from this work is shown below as Fig. 2.5. The conclusions reached in [10] are described below. This work was used as a starting point for the efforts in this thesis. In many cases, it was possible to either use these results directly, or extend them to achieve a model for a more complex system. The results concerning explicit rate feedback schemes are principally the results of interest for this thesis.

It was concluded in the past work that if the source and switch algorithms are well chosen, it is possible to achieve source rates and queue sizes that settle to expected steady-state values, given certain system constraints. The discrete-time control model resulting from Fig. 2.1 is shown in Fig. 2.5. In this figure, the controller located in the switch calculates and sends to the sources the common rate, $R_i[n]$, and all $N \geq 1$ sources under the control of that switch must transmit at no more than that rate. The total rate of cells entering a given queue is $R[n]$. In the feedback loop, the controller is a proportional gain constant G.

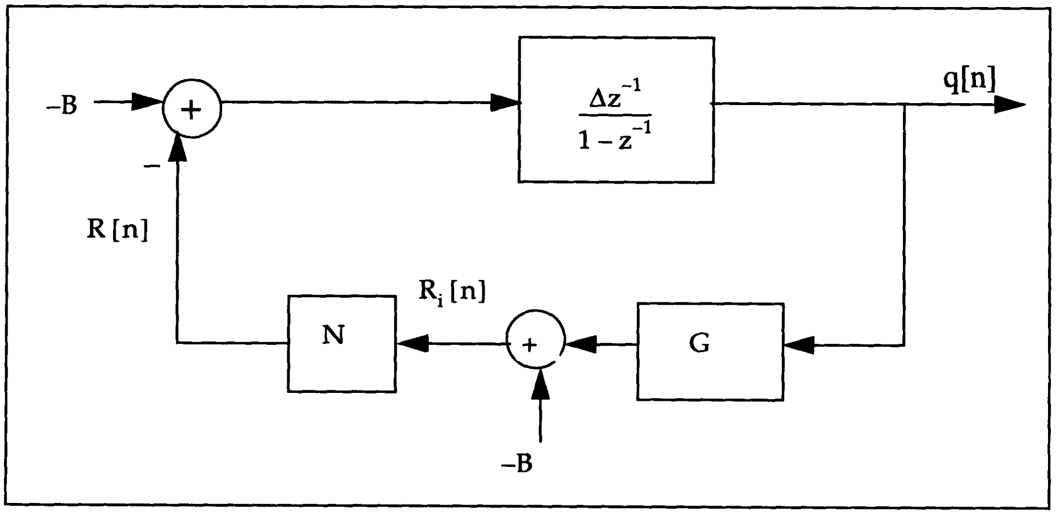

Figure 2.5: Original block diagram model

In this discrete-time system, the queue size, $Q[n]$, is determined every Δ seconds and its value is given by the following equation

$$Q[n] = Q[n-1] + \Delta(R[n-1] - B) \tag{2.8}$$

where B is the bandwidth of the system or the total rate at which a queue can service the cells that it receives and $R[n]$ is the total rate of cells entering the queue. When using a proportional compensator in the feedback loop, the rate at which each source is told to operate, $R_i[n]$, is given by

$$R_i[n] = \max(B - GQ[n], 0) \tag{2.9}$$

Here B is an offset term used in the switch algorithm to make sure that the rate for each source is set to a positive value less than the bandwidth, which is the maximum rate any source should consider. In this equation, G is the proportional gain constant in the feedback loop.

From Eq. (2.9), the steady-state rate $\bar{R_i}$ of each source is given by

$$\bar{R_i} = B - G\bar{Q} \tag{2.10}$$

where $\bar{Q}$ is the steady-state value of the queue. Since all sources are given the same rate, the steady-state rate is given by

$$\bar{R_i} = B/N \tag{2.11}$$

where $N \geq 1$ is the number of sources under the control of a given switch. Solving for the steady-state queue using these two relations yields

$$\bar{Q} = \frac{B(N-1)}{GN} \tag{2.12}$$

Taking the derivative of Eq. (2.12) with respect to N yields

$$\frac{d\bar{Q}}{dN} = \frac{BG}{(NG)^2} \tag{2.13}$$

Since this is always positive, as the number of sources in the network increases, the steady-state queue size will increase. Furthermore,

$$\lim_{N \to \infty} \bar{Q} = \frac{B}{G} \tag{2.14}$$

Therefore, the steady-state queue size is bounded between 0 and B/G.

The loop gain, $L(z)$, of the closed loop system is given by

$$L(z) = \frac{GN\Delta z^{-1}}{1 - z^{-1}} \tag{2.15}$$

Based on this formulation Rohrs, *et al.* employed the tools of classical control theory to predict the behavior of the system. Bode plots were used to determine the stability of the system as a function of G and N. This early work provides evidence as to the practicality of using analysis tools to predict the behavior of a system.

2.5 ATM Specifications for Congestion Control

The ATM Forum is presently developing its standards for congestion control algorithms. The current framework the Forum is operating under would allow switches to use either a binary or an explicit rate algorithm, or a combination of the two to achieve congestion control. This section discusses the principle features of the proposed standard provided by the ATM Forum. These standards and specifications are still under development.

When a connection is set-up, the source must specify a minimum cell rate (MCR) and a peak cell rate (PCR) for the connection. An initial cell rate (ICR) at which the source begins to transmit is also specified. The source is required to transmit an RM cell before transmitting any data and transmit one RM cell for every N_{rm} data cells after that. Before sending an RM cell, if X_{rm} forward RM cells have been sent since the last backward RM cell was received, then the source must lower its rate to a value no lower than MCR. When the source receives a returning RM cell with its CI bit set the source initially considers reducing its rate by a multiplicative decrease factor. If a returning RM cell is received without the CI bit set then the source initially considers additively increasing its rate. After computing the new possible rate, the source must check to make sure this rate is less than both the ER in the last RM cell and the PCR. If not, then the rate is set to the lesser of these two values. The new rate must also be greater than the MCR, or it is set to this value. In this way the source algorithm can be thought of as essentially the same as the PRCA source behavior, except the ER provides an adjustable ceiling for the allowed rates. If the size of the additive increase is large, and the CI bit is never set, then this algorithm will usually set the source rate to the ER.

A switch can mark the EFCI bit in data cells, mark the CI bit in forward or reverse RM cells, lower the ER field in forward or reverse RM cells, or perform a combination of these functions. The manner in which a switch determines which cells to mark or what to put in the ER field is not specified. The various switches in a network may employ different feedback techniques. The destination is required to return to the source any RM cells it receives. It must set the CI bit if it has received any data cells with EFCI set, and it may lower the ER field if it is congested.

Chapter 3

Link Delays for Single Switch Model

3.1 Introduction

It is critically important for a general purpose data communications system to be able to handle delay and remain stable. Delay can take many forms in a network. There are propagation delays, processing delays, transmissions delays, and queueing delays. One of the simplest forms of delay to understand is the delay due to the fact that data cells must travel at a finite speed over network links. The delay due to the traveling time of a cell is referred to as propagation delay. This delay is encountered by cells travelling both to and from each source. Thus the total round trip delay is equal to twice the delay in one direction. In this thesis, propagation delays will be modeled as link delays in the networks to be analyzed. In this chapter, representations for these link delays shall be discussed.

A link delay will usually be represented in terms of the number of sample intervals to which it corresponds. It may also be useful to correlate the delay added to the model to the round trip propagation delay of the data cells of a given source. Each sample interval, Δ, worth of delay can be translated into the terms of the distance between the source and the switch. If cells travel at a rate of $0.75 \times c$, where c is 3×10^8 m/sec or the speed of light, then the distance in kilometers between a source and the first switch for each Δ of delay is

$$\frac{0.75c}{1000} = 203\text{Km}$$

Since the distance between sources and switches in ATM may be several thousand kilometers, it will be necessary for a proposed system to be stable in the face of delays which may be tens of Δ intervals long.

In the network model shown in Fig. 3.1, delay can be modeled by placing a delay on the link connecting a source to the switch. In this model, cells are generated at source 1 and source 2, then they are sent out on the links which connect these sources to the switch. Cells reach switch 1 and are processed there. Cells from both sources are then transmitted on to the destination labeled D1.

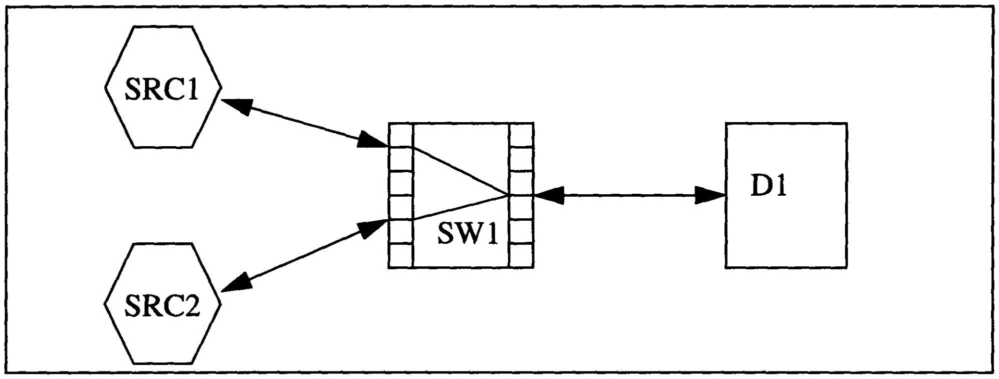

Figure 3.1: Example Network Model

This chapter discusses the modelling of propagation delay. A block diagram for the work that includes the link delay is developed and system outputs are predicted. Examples are provided to establish the accuracy of the block diagram model.

3.2 Modeling Link Delay for a Network

In continuous-time, the impulse response of a delay is given by an impulse shifted by an amount corresponding to the value of the delay, $h(t) = \delta(t-D)$, where D is any real value. The frequency domain representation of link delay is $H(\Omega) = e^{-j\Omega D}$. This term has a constant magnitude and a linear phase in the frequency domain.

The analysis in this thesis is developed from a discrete-time control system perspective. The nature of this system is discrete since it is based on an interrupt which occurs every Δ seconds. This is the time interval between the instances at which the values of source rate and queue size are collected.

The exact model of an arbitrary delay in a discrete-time system involves an infinite sum of sinc functions[9]. That is, if the system function corresponds to a delay, then it takes the form

$$H(e^{j\omega}) = e^{-j\omega d} \tag{3.1}$$

and the output, $y[n]$, is obtained from

$$y[n] = \sum_{k=-\infty}^{\infty} x[k] \frac{\sin[\pi(n-d-k)]}{\pi(n-d-k)} \tag{3.2}$$

where the input is $x[n]$. Equation (3.2) is a non-causal, infinite support filter. If the arbitrary delay term d above takes on only integer values, then Eq. (3.2) reduces to

$$y[n] = x[n-d] \tag{3.3}$$

The Z-transform representation for Eq. (3.3) is z^{-d}.

A linear approximation to Eq. (3.2) may be appropriate when D is not an integer. This approximation may be better understood in terms of continuous-time processing of a discrete-time signal. That is, if the discrete-time input signal is put through a digital-to-analog converter and then delayed and integrated as the block diagram calls for, and then finally sampled again to find the discrete-time output, the output is a linear approximation of the effect of the delay on the input. Such an approximation recreates the delay term based on a linear interpolation between the actual discrete-time sample points at which the value of the function is known exactly.

The response to a discrete-time unit step of a system with a non-integer D quantifies this discussion. The digital-to-analog conversion, the analog processing, and the sampling of the resultant signal are shown in Fig. 3.2. Here, the zero order hold function converts a discrete-time step into a continuous-time step. The processing is done in continuous-time, and then the output is sampled every Δ seconds. Let $D = k\Delta + D_p$, where k is the integral delay component of D and D_p is the partial delay component of the delay D. The discrete-time response of the system based on Fig. 3.2 can be modeled by the following equations

$$y[n\Delta] = 0 \qquad n \le k \tag{3.4}$$

$$y[n\Delta] = \Delta - D_p \qquad n=k+1 \tag{3.5}$$

$$y[n\Delta] = y[(n-1)\Delta] + \Delta \qquad n \ge k+2 \tag{3.6}$$

or

$$y[n\Delta] = (n-k)\Delta - D_p \qquad n > k \tag{3.7}$$

From this, $Y(z)$ can be calculated as

$$Y(z) = z^{-k}\left[\frac{\Delta}{(1-z^{-1})^2} - \frac{D_p}{1-z^{-1}}\right] \tag{3.8}$$

To calculate $H(z)$, it is necessary to multiply Eq. (3.8) by $(1-z^{-1})$ twice, once to change $Y(z)$ from a step response to an impulse response, and once to remove the effect of the integrator from the system. This yields

$$H(z) = z^{-k}\left[(\Delta-D_p) + D_p z^{-1}\right] \tag{3.9}$$

It is possible to use this equation, which is a weighted interpolation between two integer delay samples, to find a value for the arbitrary delay

.

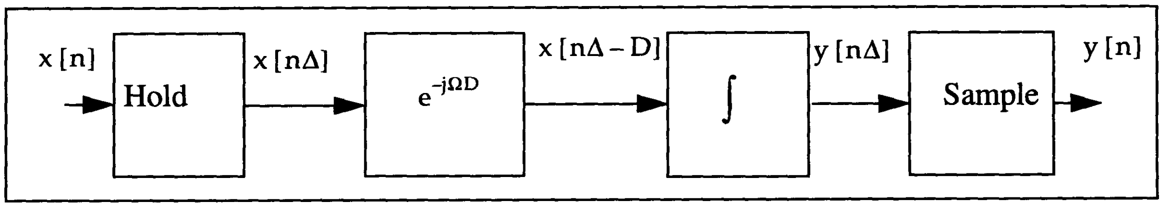

Figure 3.2: Model used to determine linear approximation for arbitrary delay

Unfortunately this modeling of an arbitrary delay term causes some problems in the analysis of its effect on the system response. The problem is that this linear approximation for the delay causes a serious inaccuracy in the frequency domain. All delays should have a constant magnitude and linear phase, but the $H(z)$ above has neither. Shown in Figs. 3.3 and 3.4 are the Bode magnitude and phase response plots for the approximation to the arbitrary delay term. For these plots, $k = 0$ and $D_p = \Delta/5$ have been chosen. In Fig. 3.3, the magnitude is not a constant. In Fig. 3.4, the phase is not linear. Due to these facts, the approximation used to derive this result is not an appropriate one for the purposes of this thesis since it serves to confuse the modeling issues of stability and analyzability. Therefore, this partial delay model will not be used.

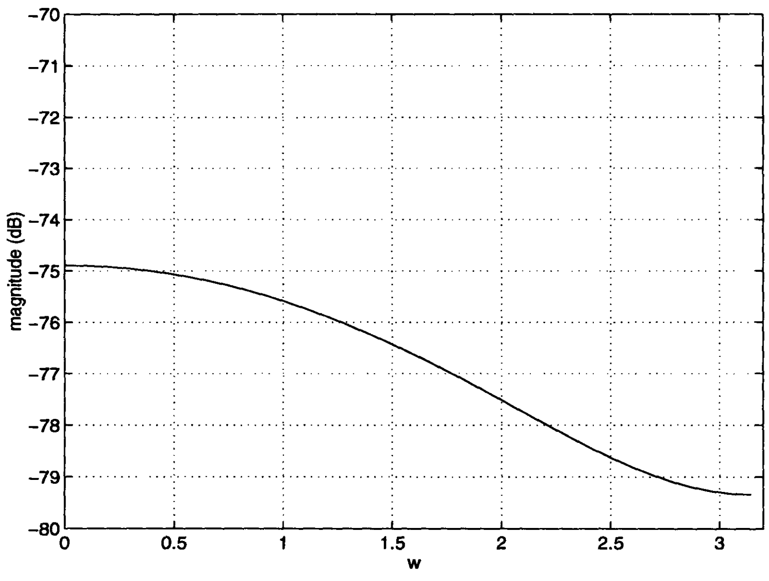

Figure 3.3: Magnitude plot for partial delay term

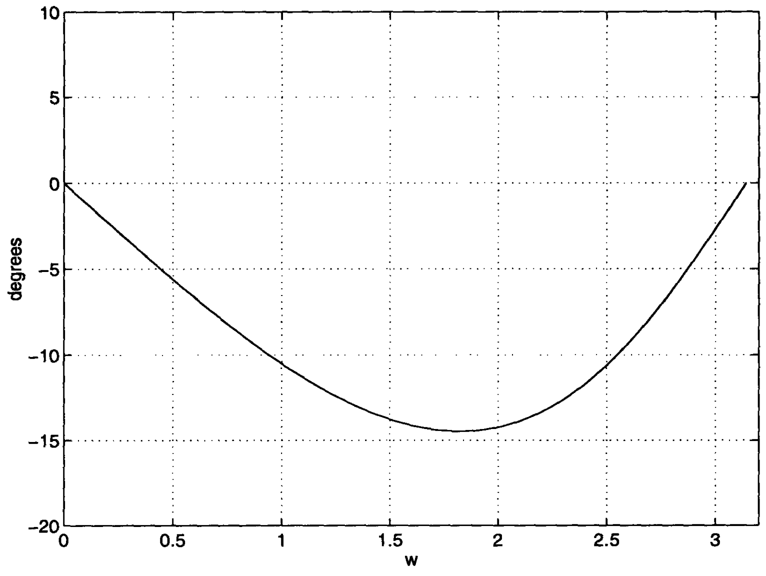

Figure 3.4: Phase plot for partial delay term

Thus, since attempts at modeling arbitrary delay in a discrete-time system have proven either too inaccurate or too complex, delay shall be restricted to integral Δ amounts. This can be represented exactly in discrete-time, and since no interpolation is involved, no infinite sum of sinc functions is necessary. As stated above, the exact integral Δ representation for delay in discrete-time is z^{-d}, where the integral value of delay is d. The Bode plots of Figs. 3.5 and 3.6 verify that an integral delay in a discrete-time system has the expected magnitude and phase responses. Fig. 3.5 indicates a constant magnitude and Fig. 3.6 indicates a linear phase response. A block diagram can now be constructed on the basis that the modeling of link delay shall be confined to integral Δ delays.

A new model which extends the number of sources to an arbitrary number is shown as Fig. 3.7. A block diagram which has been derived based on Fig. 3.7 and extended from Fig. 2.5 is shown as Fig. 3.8. Cells of each of the N sources in the block diagram can incur a different delay. Therefore, the block diagram has N different feedback branches. The integral Δ delays encountered by the cells of each source take on the values d_i, $i \in [1, N]$. Therefore, the exponents in the block diagram are d_1 through d_N. This figure can be used to analyze and predict the behavior of the system in Fig. 3.1.

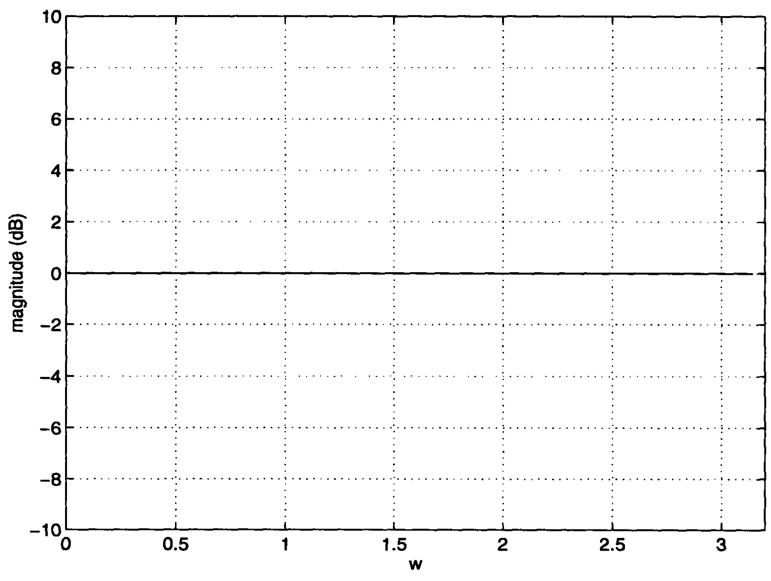

Figure 3.5: Magnitude plot for integral delay

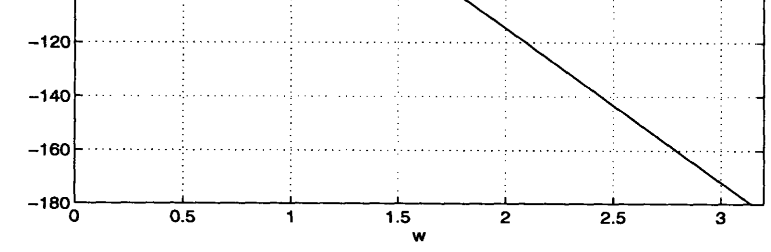

Figure 3.6: Phase plot for integral delay term

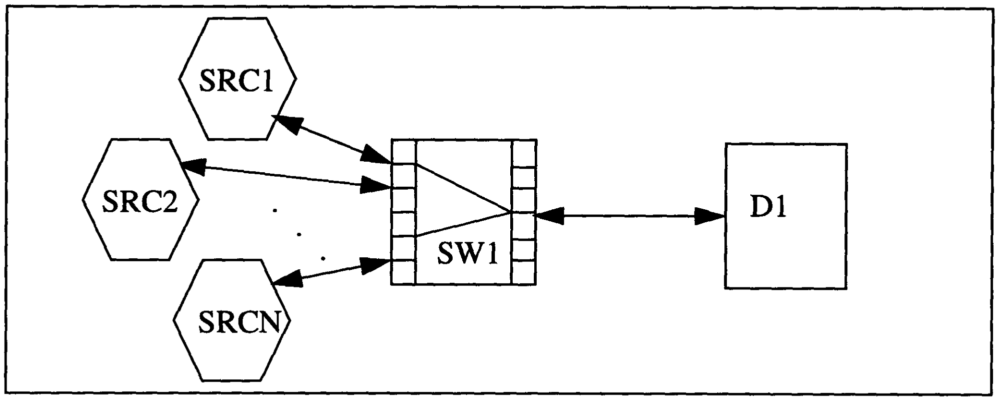

Figure 3.7: Network with one switch and an arbitrary number of sources

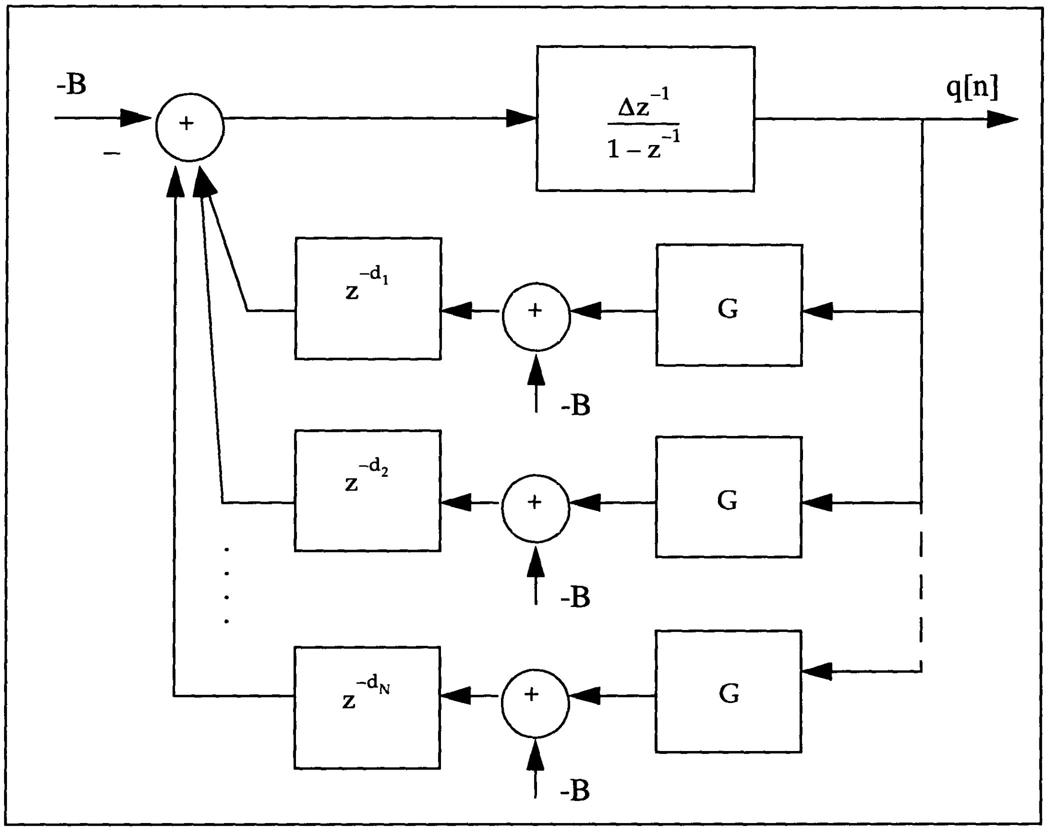

Figure 3.8: Block diagram corresponding to the one-switch network

3.3 Prediction of Behavior Due to Delay

It is possible to analyze the block diagram of Fig. 3.8 to predict the system behavior. However, the analysis becomes more direct if that block diagram is redrawn to combine the delay terms in the feedback path. The block diagram that results from this procedure is shown in Fig. 3.9. The delay terms have been combined into one polynomial in z. The value d_{max} is the largest delay that any cells encounter in the system; that is,

$$d_{max} = \max[d_1, d_2, \ldots, d_N] \tag{3.10}$$

In this polynomial, each coefficient a_0 through $a_{d_{max}}$ must be an integer greater than or equal to zero and less than or equal to N, the number of sources. Also, the sum of the coefficients must be N.

$$\sum_{i=0}^{d_{max}} a_i = N \tag{3.11}$$

The block diagram may be analyzed via Bode plots.

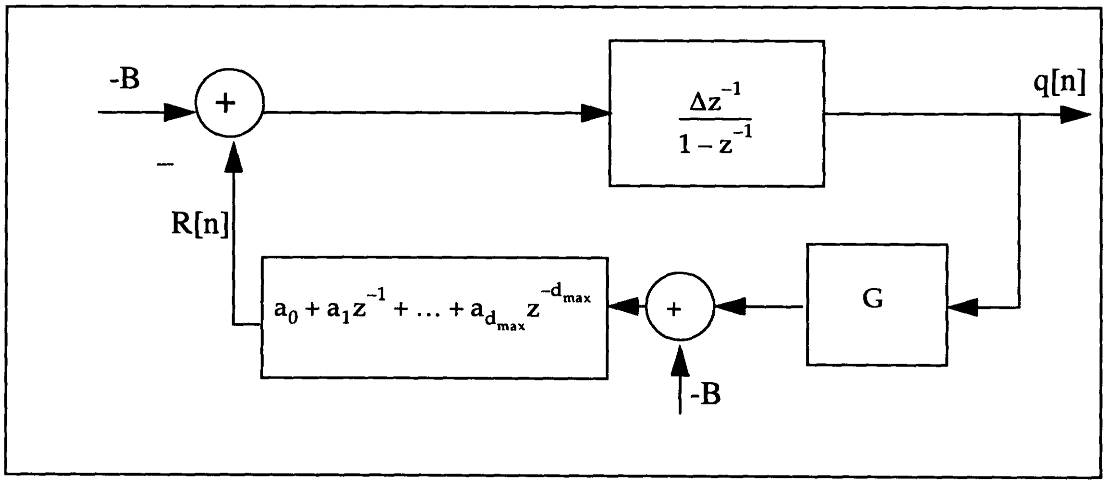
-B
+
–
R[n]
$\frac{\Delta z^{-1}}{1-z^{-1}}$
q[n]
G
+
-B
$a_0 + a_1 z^{-1} + \ldots + a_{d_{max}} z^{-d_{max}}$

Figure 3.9: Compound model

If the delay polynomial shown in Fig. 3.9 is ignored so that the basic system behavior can be discovered, the loop gain equation is

$$L(z) = \frac{\Delta N G z^{-1}}{1 - z^{-1}} \tag{3.12}$$

Parameter values of $G = 100$ and $\Delta = 0.0009$ are used.

The plots resulting from the Bode analysis of this equation are shown as Figs. 3.10 and 3.11. Bode plots based on a delay term can be added to the plots of Figs. 3.10 and 3.11 to form Bode plots for the complete system. Thus, it is possible to discern information concerning the response of a system without link delay, then add the effect of delay indicated by the Bode plots of the delay term. In this way, the response of the system to various amounts of delay can be determined.

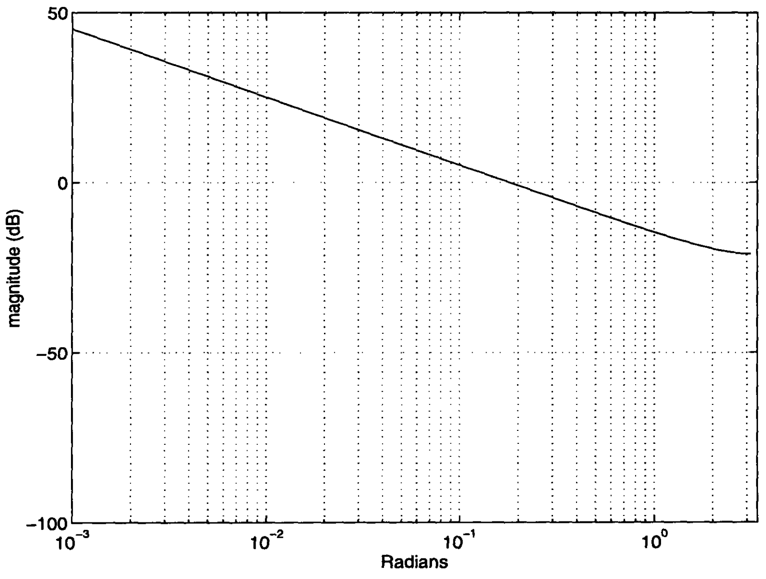

Figure 3.10: Magnitude of loop gain without delay

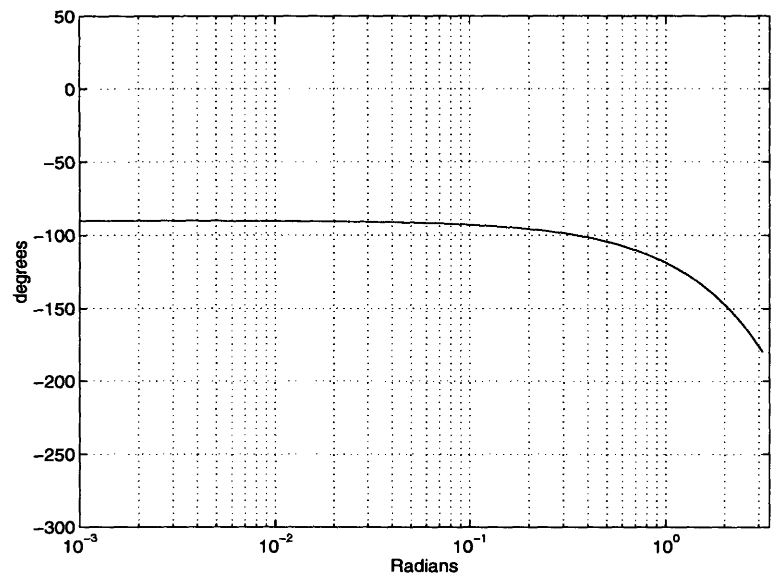

Figure 3.11: Phase of loop gain without delay

For the first analysis of the system with added delay, link delays of 0.0018 seconds, or 2Δ, are added to the links connecting source 1 and source 2 to switch 1. The round trip delay has a value of 4Δ for each of the two sources. For these values, the delay polynomial takes on a value of $2z^{-4}$. The Bode magnitude and phase plots for $2z^{-4}$ are shown as Figs. 3.12 and 3.13. These Bode plots can be added to the Bode plots for the loop gain equation to determine the overall behavior of the system. The combination of plots indicates that any additional delay terms serve to erode the phase margin, a figure of merit for the stability of the system. When the phase margin is below about $45°$, the system is not robustly stable and oscillations in the system step response are likely[11]. This particular delay term reduces the phase margin from about $85°$ in the system without delay to about $35°$ when the delay is added. Therefore, when this delay term is included in the block diagram, the step response of the system is likely to be oscillatory.

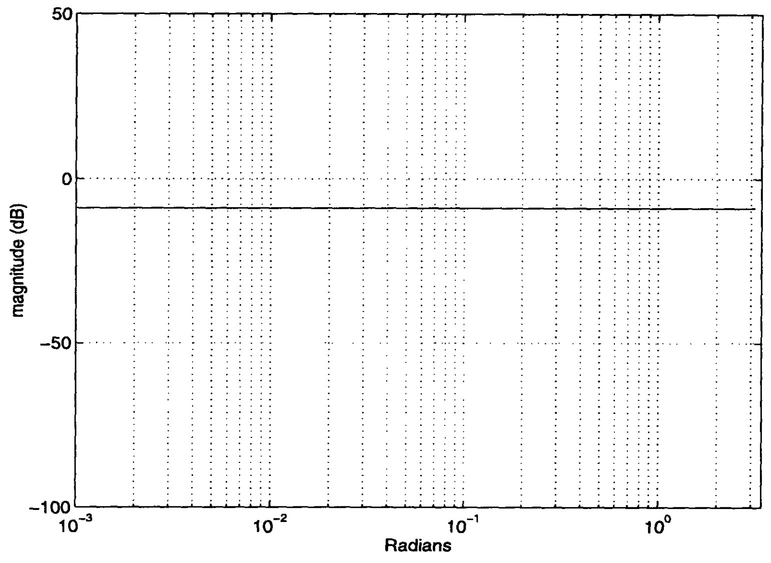

Figure 3.12: Magnitude plot for example with $2z^{-4}$

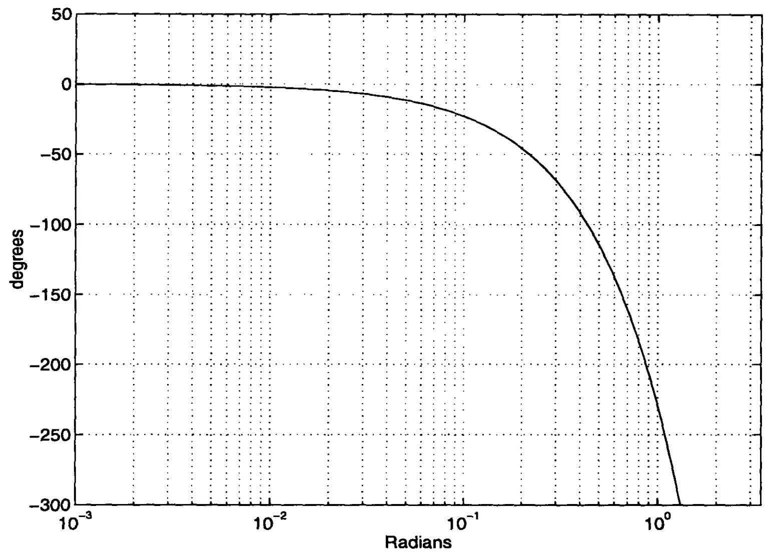

Figure 3.13: Phase plot for example with $2z^{-4}$

It is also worthwhile to show the Bode plots for a different value of the delay polynomial to give another indication of the type of responses that are possible. The next pair of Bode plots are based on a delay polynomial which takes the form $z^{-2} + z^{-4}$. In this case, source 1 cells incur a delay of Δ in each direction and source 2 cells incur a delay of 2Δ in each direction. The Bode plots that result from this delay term are shown as Figs. 3.14 and

3.15. When this delay term is added to the block diagram which is described by the loop gain of Eq. (3.12), then the phase margin is reduced from about 85° to about 45° when the delay is added. This amount of phase margin indicates that the system step response may or may not be oscillatory. Therefore, at the very least, the step response of the system from the second delay example will be less oscillatory than the system from the previous delay example for which the phase margin was less.

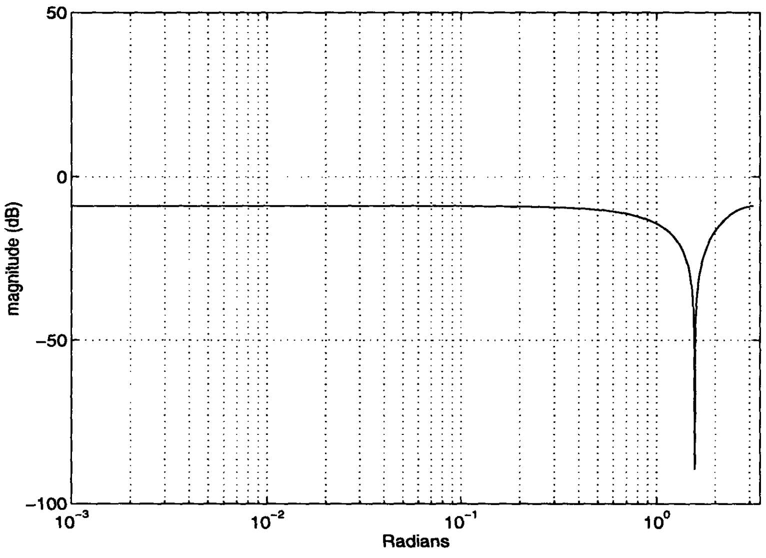

Figure 3.14: Magnitude plot for $z^{-2} + z^{-4}$

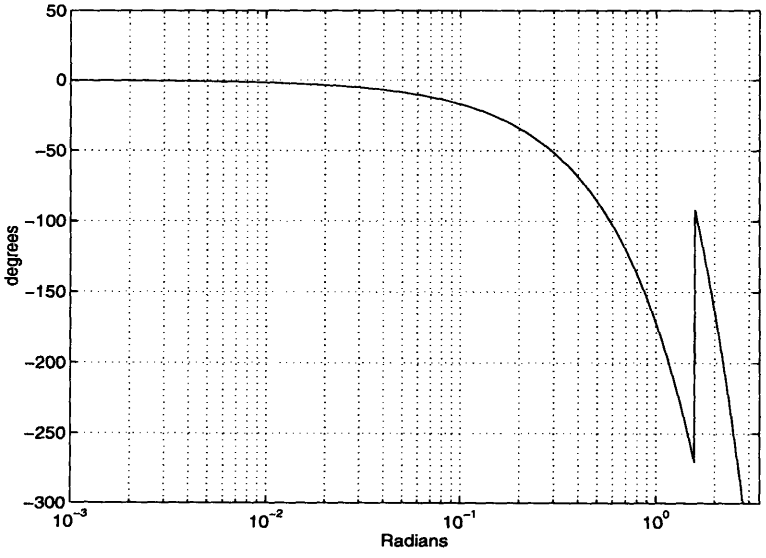

Figure 3.15: Phase plot for $z^{-2} + z^{-4}$

Bode plots can provide a suggestion of the behavior of the system. For example, they provide information as to the stability of a system. To gain more detailed information about the system, equations can be developed to provide a clearer picture. Such equations are derived from the block diagram of the system. These equations take the form of Z-transforms. When the inverse Z-transform is taken, aspects of the system such as the period of oscillation, the overshoot, the peak value and the settling time of the system can be determined. An example of the procedure used to find these equations shall be provided to verify the possibility of accurately estimating the properties of the system response given just a simplified block diagram model.

The following analysis is developed for the example where the delay polynomial takes the form $2z^{-4}$. The transforms of the queue and rate signals of Fig. 3.9 are given by

$$Q(z) = \frac{-(B(z) + R(z))\Delta z^{-1}}{1 - z^{-1}} \tag{3.13}$$

$$R(z) = (GQ(z) - B(z))2z^{-4} \tag{3.14}$$

Combining Eqs. (3.13) and (3.14) yields

$$Q(z) = \frac{(2 - z^{4})B\Delta}{z^{5} - z^{4} + 2\Delta G} \tag{3.15}$$

A similar equation can be developed in terms of the rate, but they each provide the same information.

When the dominator of the above equation is factored, the poles are found to be $0.8698 \pm 0.2388i$, $-0.0794 \pm 0.6120i$, and -0.5809. The system response will be oscillatory due to this set of poles. The inverse transform of Eq. (3.15) provides a complete description of the queue size for all $n \geq 0$. However, in this case, the system can be described as dominantly second order with the dominant pole pair being $0.8698 \pm 0.2388i$. This pole pair contributes a time domain term of the form $0.9^{n}\cos(0.26n)$. Since this mode is the dominant term, much of the behavior of the system can be determined from it alone. For example, this mode has a period of $n = 2\pi/0.26 \approx 25$. Each n is equivalent to a Δ interval of time. Therefore, the period of oscillation of the response will be approximately $25\Delta \approx 0.023$ seconds.

The unit step response will settle after about five time constants. The value n corresponding to one time constant is found from

$$\frac{1}{e} = 0.9^n$$

from which n is found to be approximately 9.5 and the settling time is $5n \approx 47$. This corresponds to $47\Delta \approx 0.042$ seconds after the system begins to oscillate. The system begins to oscillate only after the stream of RM cells begins to return to each source. A more detailed evaluation of the system is possible. It would involve analysis of the behavior of the system function due to all of the poles. If the model is accurate, then these values will closely match the actual system response. When the equation-level simulation and the discrete-event simulation results are calculated, they will be compared with these results.

By Eq. (2.12), the predicted steady-state value of the queue size is

$$\bar{Q} = \frac{B(N-1)}{GN} = 1769 \tag{3.16}$$

By Eq. (2.10), the predicted steady-state value of the source rates is

$$\bar{R_i} = B - G\bar{Q} = 1.769\times10^5 \tag{3.17}$$

All of this information is available from analysis of the block diagram. It is possible to run an equation-level simulation which combines all of the information that may be elicited from the above equations into a queue size plot or a rate plot. These plots are compared in the next section to the results from a discrete-event simulation of the system to determine the accuracy of the underlying model.

3.4 Equation-level and Discrete-event Simulation Results

The above analysis will be correct if the block diagram captures all of the essential behaviors that occur in the network it represents. A direct comparison of the results of a simulation based on the block diagram model versus the actual responses of the network to the same inputs will reveal how well the block diagram and its associated equations model the behavior of the network. Figures 3.16 - 3.18 show the equation-level simulation and discrete-event simulation step responses. To produce all equation-level simulation rate responses, the initial cell rate from the discrete-event simulations was used as the first point, and the time at which the discrete-event plots first changed from the initial cell rate due to RM feedback was used as the time at which the equation-level simulation rates first changed as well.

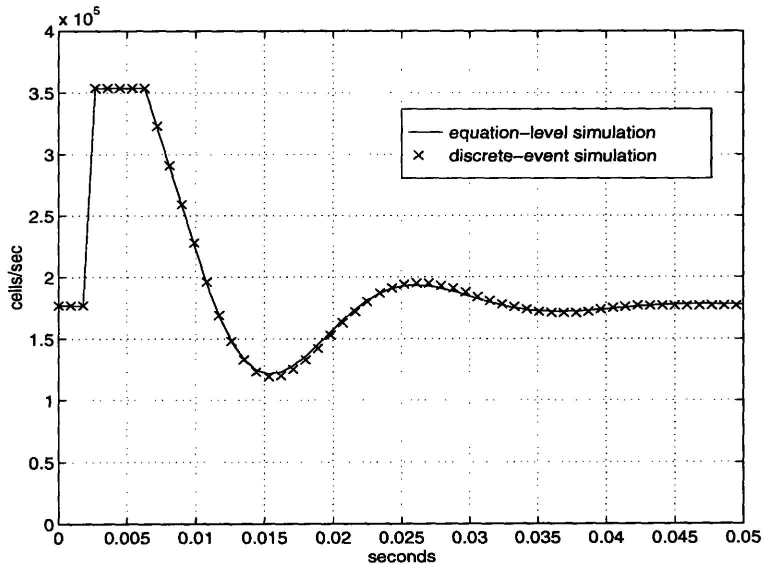

Figure 3.16: Comparison of source 1 rates

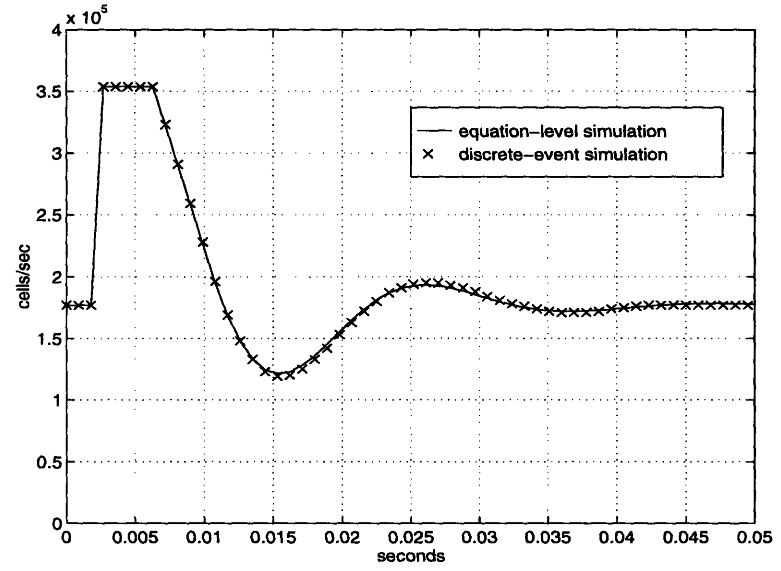

Figure 3.17: Comparison of source 2 rates

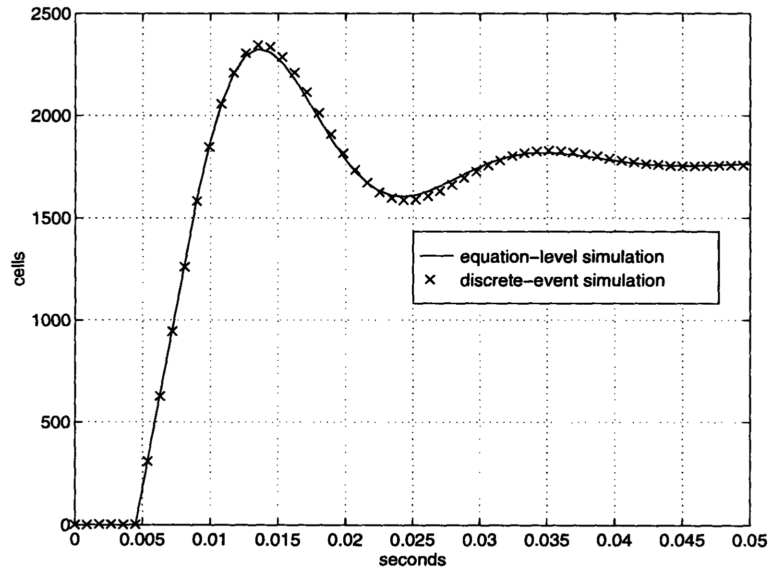

Figure 3.18: Comparison of queue sizes

Fig. 3.16 shows the plots of the equation-level simulation source 1 rate versus the discrete-event simulation source 1 rate. The source rates start at the predetermined initial cell rate, and then shortly thereafter step up to the maximum cell rate of 3.538×10^5 cells/second due to the first feedback from switch 1. The sources then wait for more feedback from queue 1 located in switch 1. Once the queue starts to grow, the sources receive feedback indicating that a rate slowdown is required. Fig. 3.17 indicates that the source 2 rate behaves in exactly the same way as the source 1 rate for both the discrete-event and the equation-level simulations since the two sources follow identical paths to the same destination.

The equation-level and discrete-event size of the queue 1 plots are shown in Fig. 3.18. After the brief initialization phase, the switch starts to receive a steady stream of RM cells returning to the sources which it can use to control their rates. The switch controls the sources by transmitting to them the source rate it calculates based on the size of the queue. Changes in the queue size cause changes in the source rates. Peaks in the queue size are correlated to troughs in the source rates.

The equation-level simulation and the discrete-event simulation source rates and queue sizes all have a period of oscillation of about 0.022 seconds relative to the value of 0.023 seconds predicted by the analysis above. The rate and queue step responses settle at about 0.043 seconds. This may be compared to 0.042 seconds, the value estimated by the approximate analysis above. There is little difference between the equation-level and discrete-event results in each of the three figures. Therefore, estimated equations are useful for quickly determining the general behavior of the system, but even better results are obtained when predicted values are obtained from an equation-level simulation of the block diagram model. These results verify a high degree of accuracy for the block diagram model of the system.

The minor differences between the equation-level and discrete-event results of each figure can be quantified by a squared error analysis. To find this value, for each point of the plots, the difference is found and squared. Figure 3.19 shows the squared error versus time plot for one pair of source rates. It can be applied to both source rates since Fig. 3.17 is identical to Fig. 3.16. Figure 3.20 shows the squared error plot for the queue size plots.

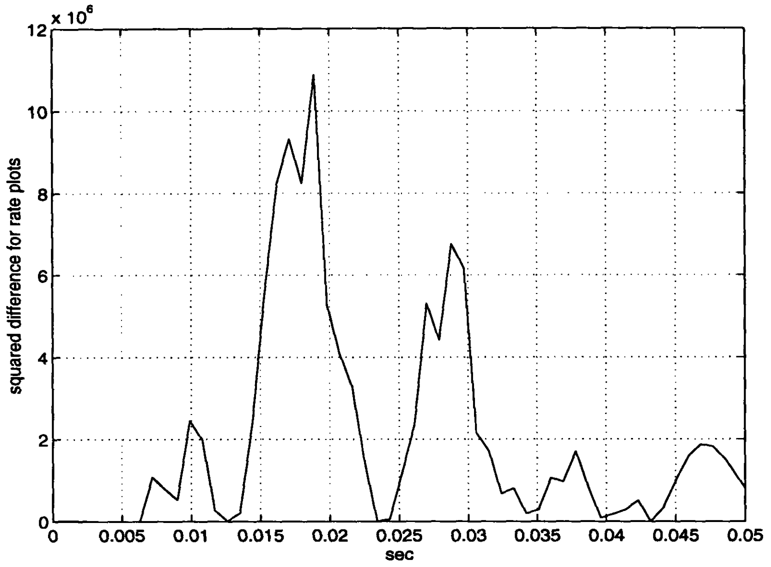

Figure 3.19: Comparison of the equation-level and discrete-event rate plots

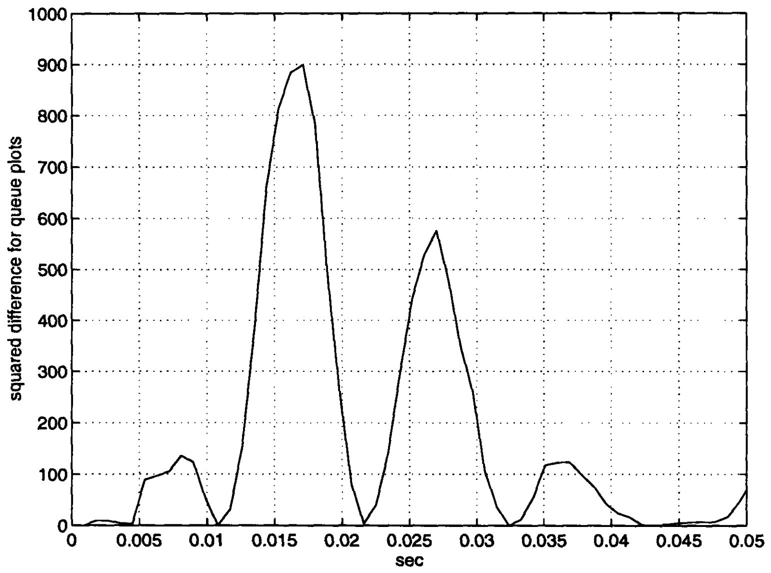

Figure 3.20: Comparison of the equation-level and discrete-event queue size plots

In most cases, plots of these values do not significantly add to the understanding of the results. Therefore, the root mean square (RMS) difference value shall be computed, and the ratio of the RMS difference value as a percentage of the steady-state value ($\mathrm{RMS}_{\%}$) shall be recorded. The $\mathrm{RMS}_{\%}$ for the source rates is 0.2% and the $\mathrm{RMS}_{\%}$ error for the queue size is 2.4%. These values indicate that the plots are very well matched. Since the pairs of plots match closely, it is possible to conclude that the block diagram used to model the system does take into account all major factors that have an influence in this model.

3.5 Conclusions

The block diagram model of Fig. 3.9 allows prediction of the behavior of a single-switch system with added link delay. The block diagram can be converted to equations relating present values of the rates and the queue sizes to previous values of rates and queue sizes. Using these equations, a simulation can be constructed to predict the behavior of the network. If such equation-level simulation results match the discrete-event simulation results, then the block diagram model can be confidently asserted to be a good representation of the system. The match between equation-level and discrete-event results to this point indicates that the significant facets of the behavior of the system have been modeled.

Chapter 4

New Dynamics Associated with Multi-Switch Systems

4.1 Introduction

Figure 4.1 shows an example of a network comprised of more than one switch. In this figure, two sources share the same output port of switch 1. The cells from the two sources then share the same link in transit to switch 2. Switch 2 directs the sources to their respective destinations. The destinations transmit RM cells back to the sources that created them. Several new factors play a role in the behavior of such a system. In this chapter, the new factors are discussed. The details of the operation of the new dynamics are provided through three examples using the network of Fig. 4.1.

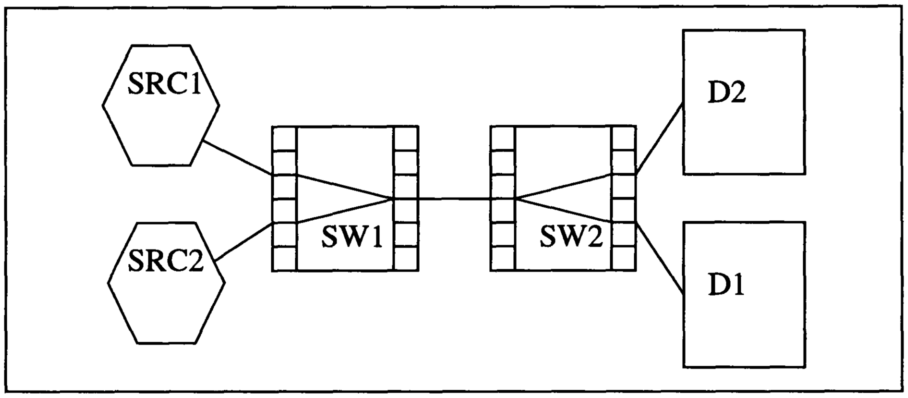

Figure 4.1: Topology to Illustrate Effect of Networking Switches

4.2 New Dynamics

In the previous chapter and in previous work[10], the role of the switch in a one-switch system has been modeled by a linear discrete-time control system. These results still hold for switch 1 in Fig. 4.1. But the switch 2 model is not the same. The example system of Fig. 4.1 is sufficient to illustrate two previously unmodeled effects that occur in a two switch system. These effects occur as the cells travel through switch 1 to switch 2.

First, queue 1, the queue located in switch 1, limits the total rate at which cells may depart switch 1 to the value of $B = 3.538 \times 10^5$ cells/second, the service rate of the queue. Therefore, when the total rate of the source cells entering this queue is greater than this value, the queue size increases which creates a non-zero queueing delay. This indicates that any change in the source cell rate will impact switch 1 immediately, but will impact switch 2 only after a length of time equivalent to the queueing delay.

Second, when cells from the two sources of Fig. 4.1 enter switch 1, they are both routed toward the same output port and are placed in queue 1 until the output port is available. Cells from the two sources take turns entering the queue. Thus, source cells are "blended" together as they enter queue 1. These dynamics indicate that at any instant cells from a particular source may be entering switch 1 and switch 2 at different rates because cells enter switch 1 directly, but they must pass through queue 1 before they enter switch 2.

These unmodeled dynamics have an impact on the results based on the example network of Fig. 4.1. The rate blending and the queueing delay indicate that the model used for switch 1 does not hold for switch 2. The equation-level simulation results based on using the same model for switch 1 as for switch 2 are shown in the next three figures. The discrete-event simulation results are plotted along with the equation-level results. The queue plots for switch 1 are shown in Fig. 4.2, and the plots for the source 1 rate at switch 1 and at switch 2 are shown in Fig. 4.3 and Fig. 4.4, respectively. The rates for source 2 are identical to those for source 1 and are not shown separately.

Figure 4.2 indicates that queue 1 is as predicted by earlier work[10]. It rises without oscillation or overshoot to the steady-state queue size of 1769 cells predicted by Eq.

(2.12). Figure 4.3 also indicates a match between the equation-level simulation values and the discrete-event simulation results. The sources initially transmit at a rate of B/2 and then step up to a rate of B when brought under control by the feedback from switch 1 which is based on the size of queue 1. The source 1 rates settle to their expected steady-state value of 1.769×10^5 cells/second as predicted by Eq. (2.11) without overshoot or oscillation. Matches occur in Figs. 4.2 and 4.3 because the sources enter the network at switch 1 in this example. As such, the modeling of switch 1 is unaffected by the new dynamics.

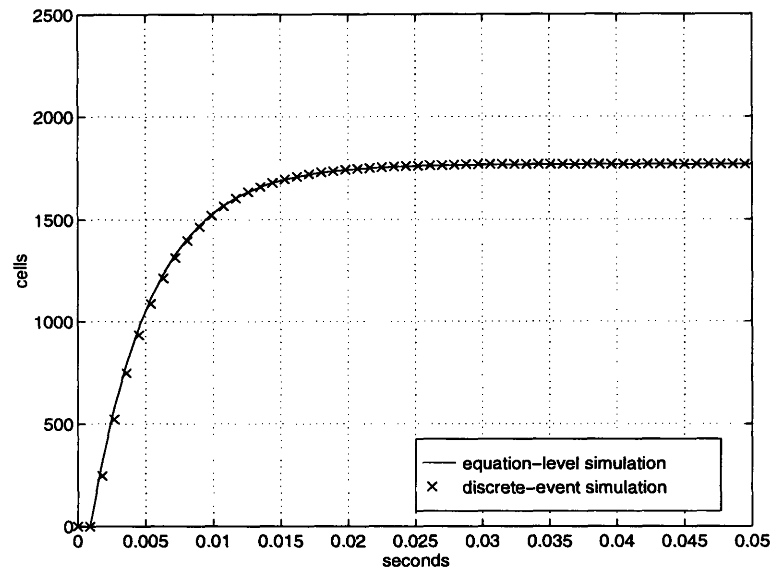

Figure 4.2: Queue 1 size

Figure 4.4 is an example where the equation-level simulation results and discrete-event simulation results do not match. In this case, the assumption was made that switch 2 could be accurately modelled by Fig. 2.5 since this has been shown to be a good model for switch 1. But the discrete-event simulation rate of source 1 cells entering switch 2 is not the same as the rate of source 1 cells entering switch 1. So it is possible to conclude that there are unmodeled dynamics when Fig. 2.5 is used to represent switch 2. Cells from the two sources are blended in queue 1, and the result is that cells flow into switch 2 at a rate that is always approximately B/2. Cells entering switch 2 have also been delayed by queue 1. Figure 2.5 can not be used to represent switch 2. A model for switch 2 incorporating the new dynamics shall be formulated in the next section.

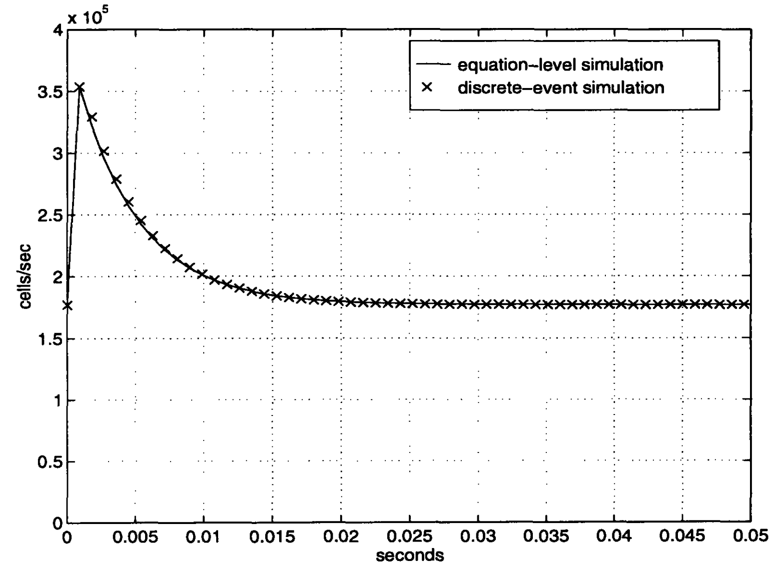

Figure 4.3: Source 1 rate seen by switch 1 (source 2 is identical)

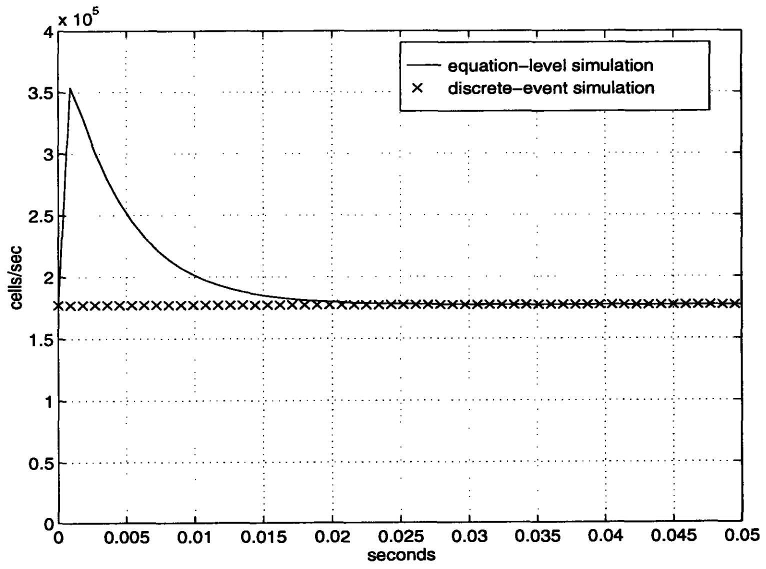

Figure 4.4: Source 1 rate seen by switch 2 (source 2 is identical)

4.3 Improved Model

4.3.1 Queueing Delay

Study of the role that queue 1 plays in the network of Fig. 4.1 indicates that a previously unmodeled delay exists in the network. In this figure, once cells are generated by source 1 or source 2, they are transmitted to switch 1. If cells arrive at switch 1 at a rate greater than B, the service rate of the queue, then that switch develops a non-zero queue. Whenever queue 1 is non-zero, a given cell that enters switch 1 is delayed by the length of time it takes queue 1 to service all cells ahead of that cell in the queue. The value of the queueing delay will be modeled as an integer number of Δ intervals for use in the discrete-time system. The value of the queueing delay for a cell entering queue 1 at sample time n, referred to as $d_{q_1}[n]$, is given by

$$d_{q_1}[n] = (q_1[n])/(B\Delta) \tag{4.1}$$

In Fig. 4.2, the steady-state queue size is 1769 cells. Therefore, from Eq. (4.1), $d_{q_1}[n] \approx 6\Delta$ intervals or 5 milliseconds in steady-state. This delay is not negligible since, as may be noted from Figs. 3.16 - 3.18, a lesser amount of delay causes oscillations in a system step response. Therefore, this delay must be carefully modeled so that system behavior can be predicted as closely as possible.

4.3.2. Blending

In the network of Fig. 4.1, cells from source 1 are "blended" with cells from source 2. To understand this behavior so that it may be modeled, two main points must be carefully considered. First, the sequence of cells entering queue 1 at a given time will be composed of a mix of source 1 cells and source 2 cells. The total cell rate entering queue 1 can exceed B. When that sequence of cells exits queue 1, the relative ratios of source 1 and source 2 cells will remain the same, but the total rate will be limited to B. Second, even if the blending effect could be ignored, the queueing delay, discussed in the previous section, can cause the rate of source 1 cells entering queue 1 to be very different from the rate of source 1 cells exiting.

In order to determine the exact rate of source 1 cells flowing to queue 2, it is necessary to calculate the ratio of cells flowing out of queue 1 that come from source 1. As long as queue 1 is non-zero, cells flow out of it at a rate of B. It is necessary to examine the source rates that entered the switch at the proper time in the past to determine the value of the source cell rates departing the switch. If the sum of the cells produced by the sources during the interval under consideration is less than the fixed number that queue 1 can service during an interval of length Δ, then cells with a new set of rates for source 1 and source 2 will start to flow out of queue 1. In fact, many sets of source rates may flow out in an interval, and the value of $r_1^{(2)}[n]$, the rate of source 1 cells entering switch 2 between sample instants $n-1$ and n, is based on all of these rates. The sum of the source rates will give an indication of how many cells were produced at that rate.

In order to find $r_1^{(2)}[n]$, it is necessary to implement an algorithm to keep track of the cells in the system, and the rates at which those cells are produced. Finding the exact rate at which cells from a source depart the queue during an interval requires finding the rates associated with the cells at the head of the queue. Therefore, the first step in the algorithm is to determine which cells are at the head of the queue. To make this determination, the algorithm calculates the queue size, then counts cells from the back of the queue, where cells have just entered, up to the head of the queue. By keeping track of the number of cells produced at each rate, it is possible to determine the interval during which the cells now leaving the queue were produced and the rate at which those cells were produced.

In the second step, the algorithm works back through the queue starting at the head. It counts cells until the count reaches the number which corresponds to the number of cells that the queue can service during an interval. At this point, the intervals of cells flowing out of the queue and their corresponding rates have been determined. As a third step, the algorithm calculates for each source the number of cells produced at the rate associated with each interval of cells. With this information, the average rate at which cells from a given source depart the queue can be calculated. This is the source cell rate entering the subsequent switch.

This process is simplified when cells which were created during the same interval all exit the queue during the same interval. That is, if the first cell at the head of the queue was the first one produced at a given rate, and the last cell to be serviced during an interval was the last one produced at that rate, then cells can be accounted for in blocks corresponding to intervals. Generally, however, the cell at the head of the queue is not the first of the set with that rate, and the last to exit is not the last of the set with that rate. To ensure that the average rate departing the queue is calculated correctly, it is necessary to keep track of all cells individually. This makes it possible to know what fractions of the first and last intervals of cells contribute to the average rate. When all of this data is kept, the algorithm is essentially a discrete-event simulation of the system.

Since it is desired to develop an equation-level simulation which has the advantage of less complexity than the discrete-event simulation, an approximation to this algorithm is implemented. This approximation assumes that the interval corresponding to the cells at the head of the queue can be calculated based on the number of intervals worth of cells currently in the queue. The approximate algorithm also assumes the interval of cells at the head of the queue is a full interval. The algorithm does allow that the final interval of cells may be a partial interval. This rate blending algorithm calculates the number of intervals, the cells in each interval, and the rates at which those cells were produced to determine the average cell rate of source cells departing the queue and entering the next switch. This rate blending algorithm is used for the equation-level simulations of this thesis.

The following pseudo-code executes blending

1. Find cells and their rate at the head of queue i, $r_x^{(i)}[n-M]$ for all of the sources x, by setting $M = d_{q_i}[n]$.

2. Calculate the total number of intervals of cells exiting the queue during the current interval by counting cells up to $B\Delta$ starting with cells at the head of the queue.

3. Calculate the fraction of the last interval of cells needed so that the total cells exiting is equal to $B\Delta$.

4. For a given source, find the contribution to the average cell rate for each interval determined in step 2 starting with the interval at the head of the queue by calculating cells produced in the interval, dividing by $B\Delta$, and multiplying by the corresponding rate.

5. Sum the rate contributions due to each interval from step 2 to find the average cell rate departing the queue.

4.3.3 Extended Model

A new version of the block diagram which includes blocks relating to both the queueing delay and to the blending of source rates is shown in Fig. 4.5. The source rates are shown on different branches so that their combination in the rate blending function can be illustrated. The queueing delay has a significance that is equivalent to that of any other form of delay. The rate blending function is described in Section 4.3.2. From this diagram, another attempt can be made at predicting the behavior of the system shown in Fig. 4.1.

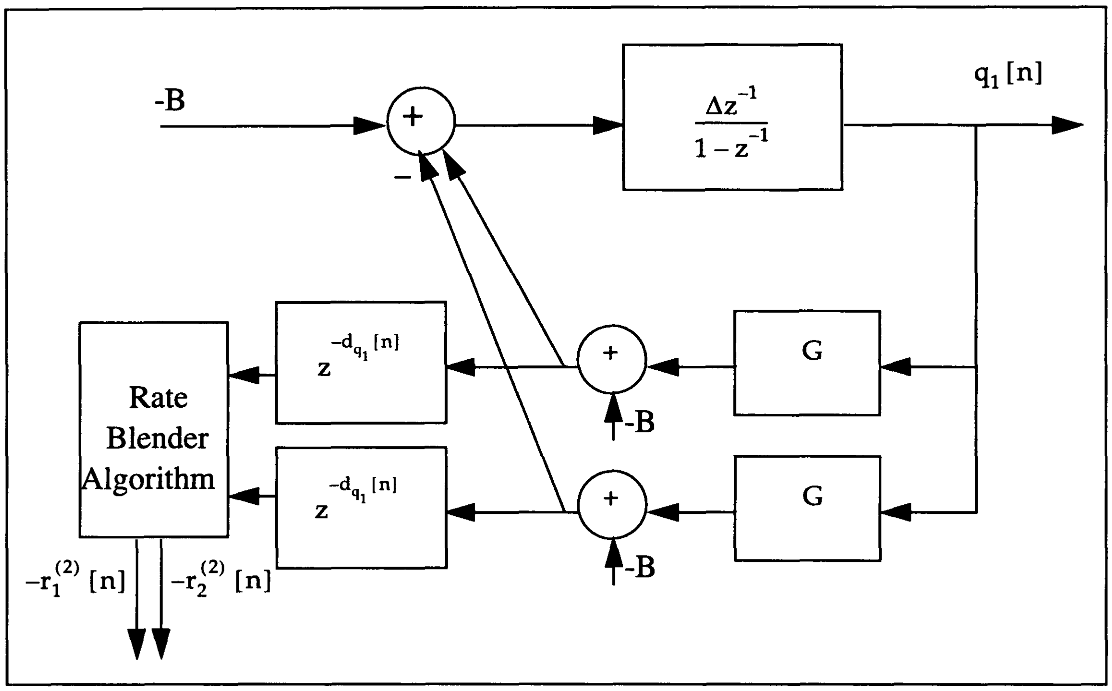

Figure 4.5: Updated Block Diagram Model

4.4 Analytical Predictions

The above block diagram model yields the same loop gain equation as is given by Eq. (3.12), and the Bode plots of Figs. 3.10 and 3.11 apply as well. They indicate that the system response will be without oscillation or overshoot. From Eq. (2.12) the steady-state value of the queue size is 1769 cells and from Eq. (2.11) the steady-state value of the source rates is 1.769×10^5 cells/second for both source 1 and source 2.

Because the system has been modeled in block diagram form in Fig. 4.5, it is possible to predict the behavior of the system analytically in as much detail as desired. In this way, equations for the step response of the system can be developed and thus very precise predictions can be made. This is the utility of applying control systems analysis to the problem of data networks. From the block diagram, the following equations can be derived

$$Q(z) = \frac{-(B(z) - R(z))\Delta z^{-1}}{1-z^{-1}} \qquad (4.2)$$

$$R(z) = 2(-GQ(z) + B(z)) \tag{4.3}$$

where $R(z)$ is the Z-transform of the total rate entering the switch. Combining these equations, and letting the inputs be discrete step inputs of height B, the result is

$$Q(z) = \frac{\Delta B z^{-1}}{(1-0.82z^{-1})(1-z^{-1})} \tag{4.4}$$

The inverse Z-transform of which is

$$q[n] = 1769\,u[n] - 1769(0.82)^n u[n] \tag{4.5}$$

where $u[n] = 1$ for $n \geq 0$, and 0 otherwise.

This equation indicates that there will be no oscillations in the system step response since there are no sinusoids present. The output will settle after about five time constants. One time constant is approximately $5\ \Delta$ intervals which corresponds to a settling time of approximately 25Δ or 0.023 seconds.

4.5 Experimental Results

Since no changes to the modeling of switch 1 have been added, the simulation results in Figs 4.2 and 4.3 are the same as would be generated by the new model. The rate blending and queueing delay affect the modeling of the inputs to switch 2. In this example, the sources are in sync, which implies that the source rates seen by switch 2 will always be $B/2$.

The effect of the queueing delay will be that the source cell rate entering switch 2 will be a delayed version of the blended rate. This delay depends directly on the size of queue 1. Queue 1 grows until it reaches steady-state, so the delay grows as well. However, since the blended rate takes on a constant value, it is not possible to detect whether a plot of the rate has been delayed since it is the same no matter when it is examined. Therefore, the effect of the queueing delay will not be obvious in the outputs of this example.

The predictions of the previous section can be compared with equation-level simulation results and discrete-event simulation results. The equation-level source rate results are shown in Fig. 4.6 along with the corresponding discrete-event simulation results.

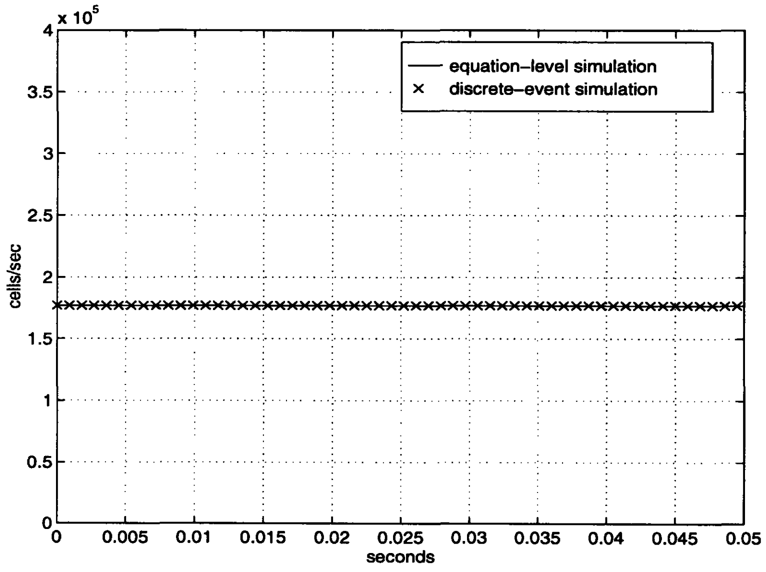

Figure 4.6: Source 1 rate at switch 2 (source 2 rate is identical)

This figure shows that the equation-level simulation results now match the discrete-event simulation data. The major improvement that has been made is that the cell rates entering switch 2, shown in Fig. 4.6, are now correctly modelled. It is possible to quantify the differences between this pair of plots by finding the squared difference as a function of time, and then finding the $\mathrm{RMS}_{\%}$ for this figure. The value for the $\mathrm{RMS}_{\%}$ is calculated to be 0.06% for the rates of source 1 and source 2 at switch 2. Due to this small value, it is possible to conclude that the models for the dynamics added to the block diagram are correct.

Another situation which provides a test of the modeling of rate blending, occurs when source 2 is delayed in start up with respect to source 1. In this example, the start up delay for source 2 is 0.02 seconds. As will be seen, this example illustrates the manner in which cells from the two sources blend together, and shows that this effect is most significant during periods in which new sources first enter the network.

The analysis of this example can be broken into two periods: the time before source 2 begins to transmit and the time after source 2 begins to transmit. Before source 2 starts to transmit, rates and sources reach steady-state values. The first cell returns to source 1 in less than an interrupt time. At this time, the following conditions exist

$$R_1[n] = B \tag{4.6}$$

$$R_2[n] = 0 \tag{4.7}$$

$$q_1[n] = q_2[n] = 0 \tag{4.8}$$

The second group of equations for the interval after source 2 has turned on, are identical to those derived earlier, Eqs. (4.2) - (4.5), since the network is now identical to the previous network. Thus the predictions made earlier still hold. The difference between the two examples is the time at which source 2 start up occurs. In this example, rate blending has a noticeable impact throughout the transient interval.

The next five figures contrast the equation-level simulation results with the discrete-event simulation results. Figure 4.7 shows that the discrete-event and equation-level queue 1 sizes have the same form. The queue remains 0 until both sources have activated because the service rate of the queue is equal to the maximum possible value that the source 1 rate can achieve. When source 2 turns on, the queue begins to grow. It reaches the steady-state value of 1769 cells predicted by Eq. (2.12). The $RMS_{\%}$ value which compares the equation-level and discrete-event queue 1 plots is 1.2%.

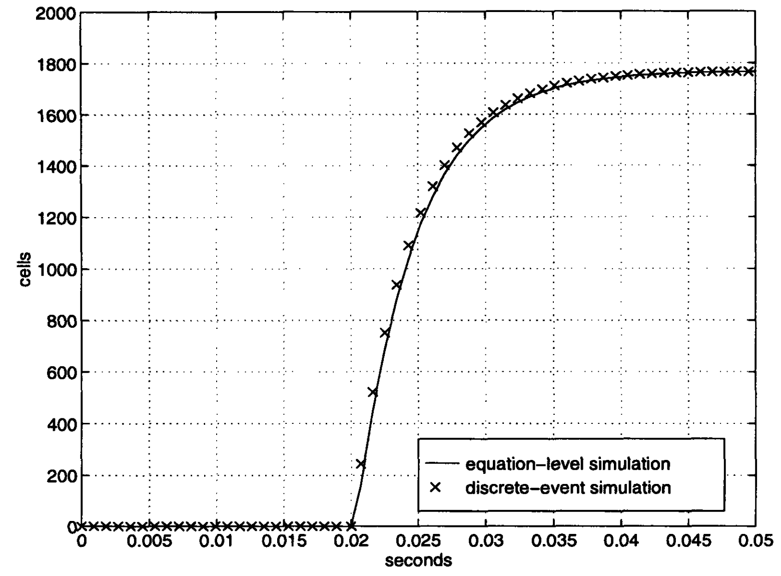

Figure 4.7: Comparison of queue 1 size values

Figures 4.8 and 4.9 show the rate of source 1 cells arriving at switch 1 and the rate of source 2 cells arriving at switch 1, respectively. These figures serve primarily to validate the analytical predictions made in this section and the previous section. Source 1 starts at B/2 and then immediately steps up to B and remains at that value until source 2 starts transmitting. When source 2 starts transmitting it attempts to step to B, since queue 1 has a size of zero and both of the switches indicate that sources may transmit at full rate. Then, both source 1 and source 2 reduce their rates toward their expected steady-state values of B/2. Since these rate values are generated in response to the size of queue 1, the relationship that exists between the queue 1 plot and the rates plots is expected.

In both figures, the equation-level simulation and discrete-event simulation plots are well matched. The slight discrepancy around $t = 0.02$ seconds in Fig. 4.9 can be attributed to the manner in which the equation-level data points were collected and plotted. A data point for the discrete-event simulation was taken just after $t = 0.02$ but the first equation-level simulation point was not taken until about half of a millisecond later. If points had been taken at exactly the same time, the peaks would match better. The small difference in the times at which the points were taken exaggerates the difference between the plots. The $RMS_{\%}$ values corresponding to these plots are 1.3% for source 1 rate at switch 1 and 2.6% for source 2 rate at switch 1. These values verify that the predictions based on the block diagram model are accurate.

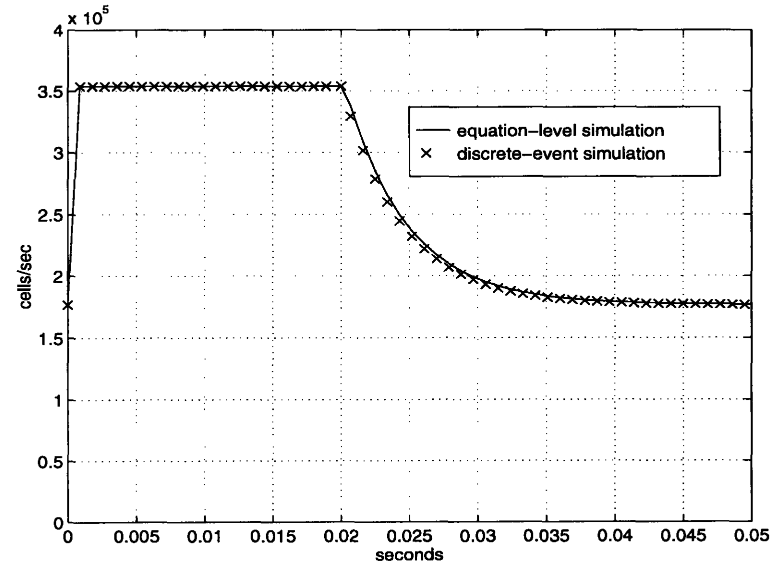

Figure 4.8: Comparison of source 1 rates at switch 1

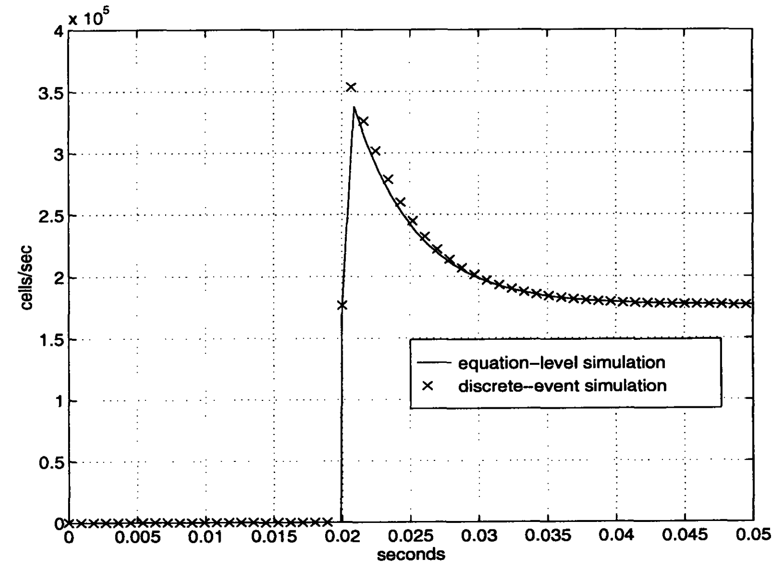

Figure 4.9: Comparison of source 2 rates at switch 1

Figures 4.10 and 4.11 show the source rates arriving at switch 2. Figure 4.10 shows that switch 2 receives source 1 cells first at a rate of B, then at a rate of B/2 as soon as source 2 turns on. Figure 4.11 shows that source 2 approximately steps from zero to a cell rate of B/2 from the point of view of switch 2, although cells are actually being generated at a rate of B by source 2.

Figure 4.12 shows the rate of source 1 cells arriving at switch 1 and at switch 2. Figure 4.13 shows the rate of source 2 cells arriving at switch 1 and switch 2. Both figures indicate that the source rate at switch 2 changes in a more dramatic fashion than the source rate at switch 1. This illustrates that the cell rate a switch receives from a source may be very different from what that source is actually producing due to the manner in which the queue constricts the total amount of traffic that may flow through the queue.

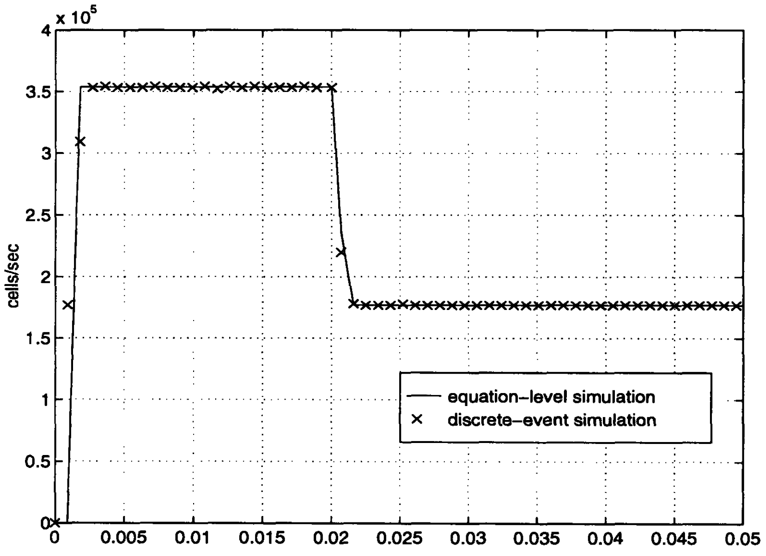

Figure 4.10: Comparison of source 1 rates at switch 2

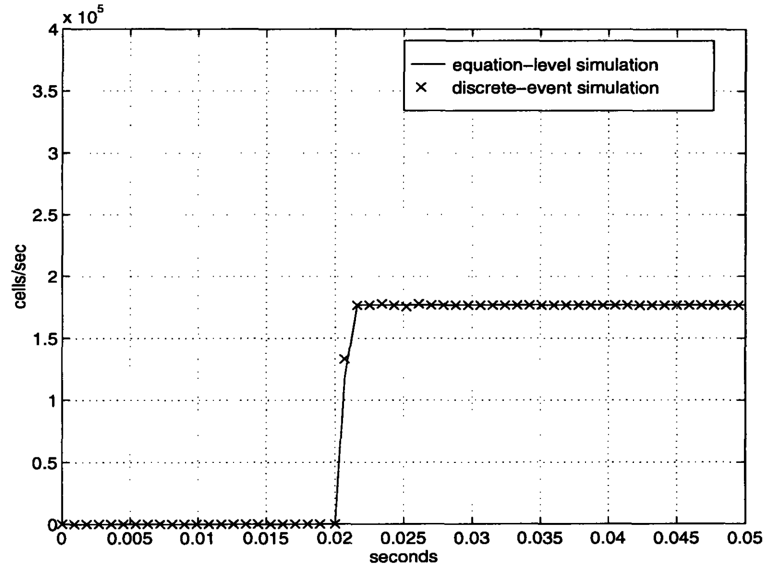

Figure 4.11: Comparison of source 2 rates at switch 2

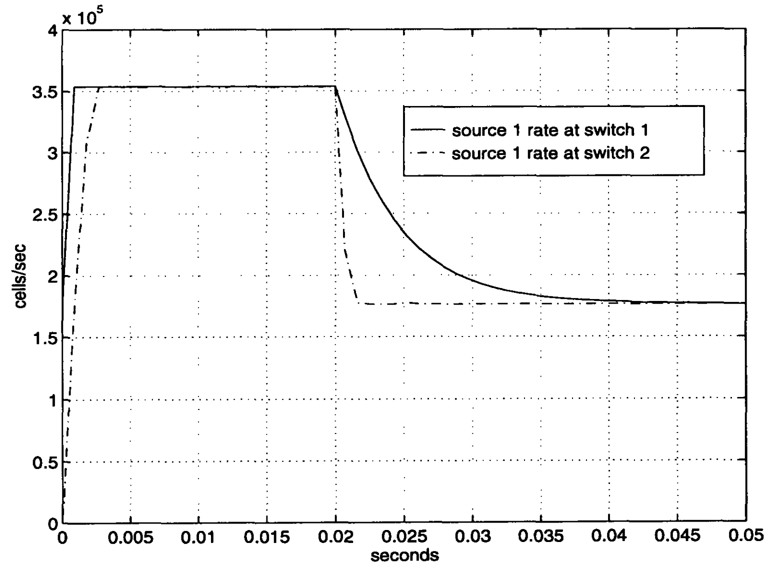

Figure 4.12: Comparison of source 1 rate at switch 1 and at switch 2

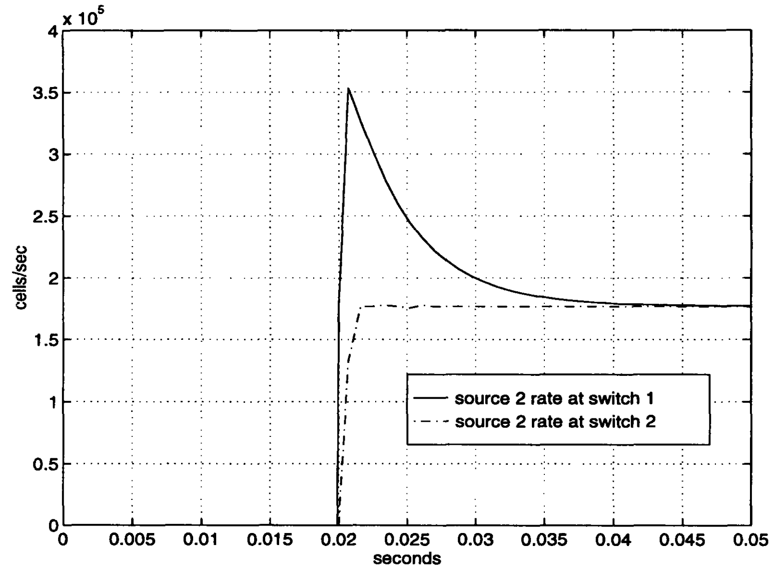

Figure 4.13: Comparison of source 2 rate at switch 1 and at switch 2

The equation-level simulation results based on Eqs. (4.6) - (4.8) and on Eqs. (4.2) - (4.5) and the discrete-event simulation results in each of these figures are very similar to one another. The peaks have the same value, the steady-state values are identical and changes occur on the same time scale. The $\mathrm{RMS}_{\%}$ values are 3.4% for source 1 rate at switch 2 and 1.1% for source 2 rate at switch 2. The block diagram model accurately predicts the system behavior.

Queue 1 takes on a non-zero value in the network just modeled. Cells are delayed due to waiting time in the queue. But the delay due to queueing is not evident in the rate outputs. The queue does not form until after source 2 has started up. After source 2 starts to transmit, the rates at switch 2 instantly reach their steady-state values of $B/2$. Although a queueing delay does take place, it is masked by the fact that the blending causes the rates to have a constant value. Therefore, in a two source system of this type, the queueing delay does not manifest itself. The effects of queueing delay on a network will be further explored in the next example.

To illustrate the effect of the queueing delay a third source can be added to the model of Fig. 4.1. The new system to be modeled is shown below as Fig. 4.14. A third source is added in parallel to source 1 and source 2. It will share the same output port of switch 1 with source 1 and source 2. Source 1 and source 2 start to transmit at $t = 0$ and achieve steady-state by $t = 0.02$ seconds. Source 3 begins to transmit at $t = 0.02$ seconds.

The difference that adding a third source makes is that the queue at switch 1 will be non-zero when source 3 begins to transmit. Having queue 1 be non-zero when source 3 starts to transmit allows the queueing delay to be noticeable in the output rate results. Since a queue develops before source 3 begins to transmit, every source 3 cell incurs a delay before it enters switch 2.

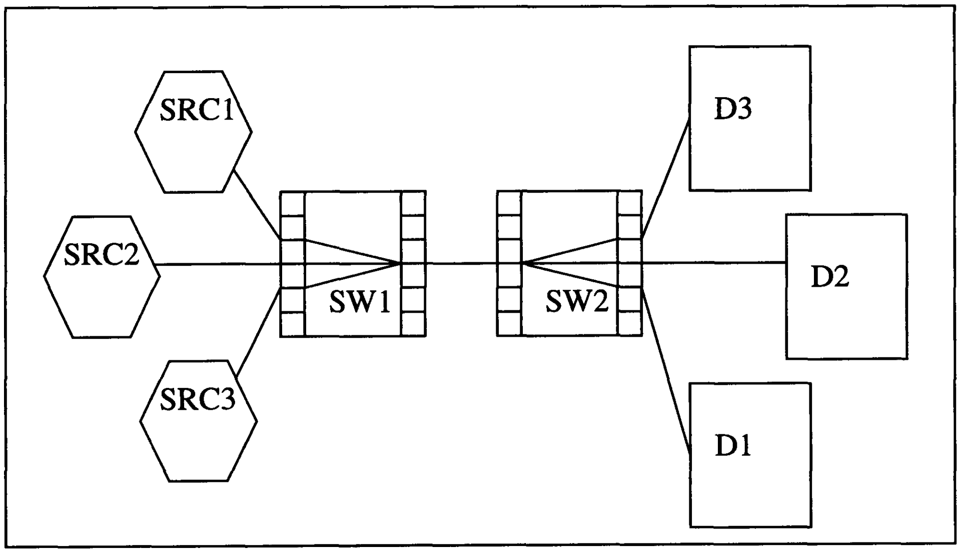

Figure 4.14: Larger network model to explore effect of queueing delay

Several equations can be developed to quantify this analysis of the system without delay. An equation for the size of queue 1 provides information about the system response. The source rates are directly related to the queue size. During the interval for which source

3 is off, the system is the two source system analyzed earlier. Therefore, Eq. (4.5) holds for the interval up to 0.02 seconds. Specifically, before time 0.02 seconds

$$q[n] = 1769\,u[n] - 1769\,(0.82)^{n}\,u[n] \tag{4.9}$$

After source 3 starts to transmit, the system becomes a three source system. The zero state part of the queue response after time 0.02 seconds or 22Δ can be found from

$$Q_{zs}(z) = \frac{-(B(z) - R(z))\Delta z^{-1}}{1 - z^{-1}} \tag{4.10}$$

$$R(z) = 3(-GQ_{zs}(z) + B(z)) \tag{4.11}$$

Combining these equations, substituting in for G and Δ, and recognizing that

$$B(z) = \frac{1}{1 - z^{-1}} \tag{4.12}$$

results in

$$Q_{zs}(z) = \frac{636.8z^{-1}}{(1 - 0.73z^{-1})(1 - z^{-1})} \tag{4.13}$$

The inverse Z-transform of Eq. (4.13) is

$$q_{zs}[n] = 2353\,u[n-22] - 2353\,(0.73)^{(n-22)}\,u[n-22] \tag{4.14}$$

where the appropriate shift is included. The zero input response of the queue due to the initial conditions can be added to this equation to determine the complete response of the queue after time 0.02 seconds. The response due to the initial condition is

$$q_{zi}[n] = 1747\,(0.73)^{(n-22)}\,u[n-22] \tag{4.15}$$

Combining these equations, the total queue size after source 3 initialization is

$$q[n] = 2353\,u[n-22] - 562\,(0.73)^{(n-22)}\,u[n-22] \tag{4.16}$$

Equations (4.9) and (4.16) provide detailed information about the behavior of the system. For example, it is possible to conclude that the system is stable since it converges to a finite steady-state value, that it is not oscillatory since there are no sinusoidal terms, and the system time constant is approximately $n = 3.2$ which corresponds to a settling time of about 0.014 seconds after source 3 starts up.

The amount of delay due to waiting time in queue 1 can also be estimated. When source 3 starts to transmit, queue 1 will have a size of approximately 1769 cells. From Eq. (4.1), this corresponds to a delay of approximately 0.005 seconds. This is the expected delay before source 3 can change its rate from the initialization value of $B/2$ and the delay until switch 2 first starts to receive any source 3 cells. Equation-level and discrete-event simulation results can now be developed for comparison.

The equation-level queue 1 size is shown in Fig. 4.15. Queue 1 grows toward the expected steady-state value of a two source system (1769 cells) as soon as source 1 and source 2 begin transmission. As soon as source 3 begins transmission, queue 1 begins to grow at a more rapid rate. It finally reaches the expected steady-state value for the three source network. The steady-state value that it reaches is predicted by Eq. (4.16) to be 2353 cells. The settling time is approximately 0.011 seconds which may be compared to the value of 0.014 seconds calculated above.

Both the equation-level and discrete-event queue 1 plots indicate virtually identical information. The $RMS_{\%}$ value which expresses the difference between the equation-level simulation and discrete-event simulation plots for the queue 1 size is 1.0%. This small error further indicates a match between these plots.

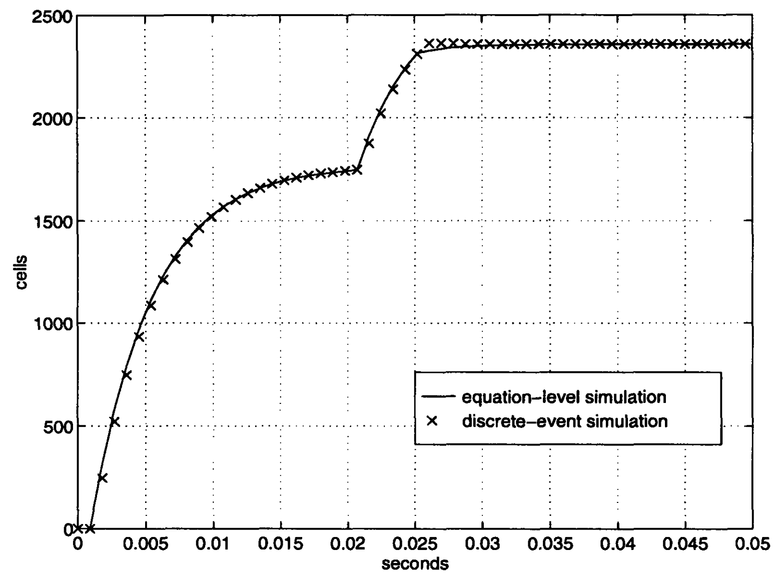

Figure 4.15: Comparison of queue 1 sizes

Figures 4.16 and 4.17 display the equation-level and discrete-event source 1 and source 3 cell rates at switch 1, respectively. The source 1 rate received by switch 1 has the expected steady-state value of 1179 cells/second predicted by Eq. (2.11) and a settling time of approximately 0.011 seconds which is comparable to the value of 0.014 seconds determined above.

Figure 4.17 displays the effect of queueing delay. As predicted above, the first time source 3 receives an RM cell indicating a rate change is at 0.025 seconds, which corresponds to the expected delay of 0.005 seconds after source 3 begins to transmit. Therefore, due to the queueing delay, source 3 transmits at its initial rate until it receives feedback information. Figure 4.18 displays the rate of source 3 cells arriving at switch 1 and at switch 2. This figure indicates that it takes about 0.006 seconds before any source 3 cells arrive at switch 2. This is a direct manifestation of the queueing delay.

The pairs of plots shown in Figs. 4.16 and 4.17 are comparable. The $\mathrm{RMS}_{\%}$ value for the source 1 rate at switch 1 is 1.5% and for source 3 rate at switch 1 it is 2.0%. These low values indicate a successful modeling of the system.

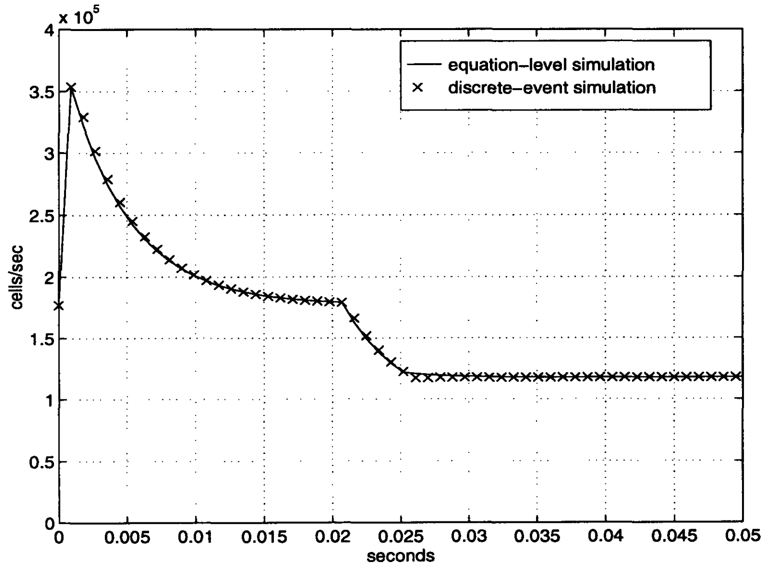

Figure 4.16: Comparison of source 1 rates at switch 1 (source 2 rates are identical)

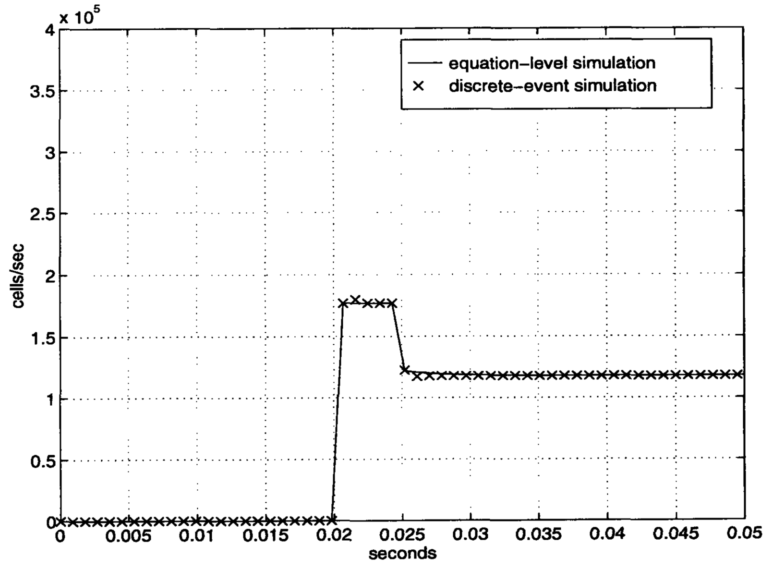

Figure 4.17: Comparison of source 3 rates at switch 1

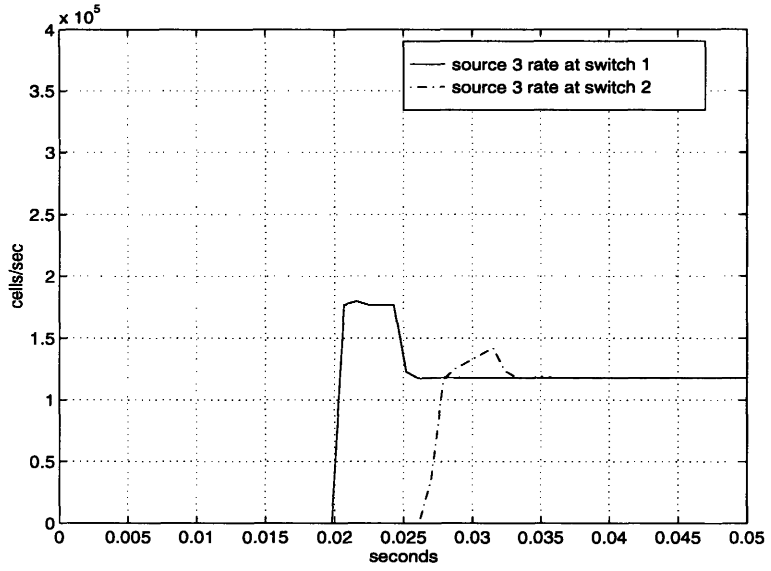

Figure 4.18: Comparison of source 3 rates at the two switches

Figures 4.19 and 4.20 show the results for source 1 at switch 2 and source 3 at switch 2, respectively. The point of interest here is the delay between the start up of source 3 and the arrival of source 3 cells at switch two. This may be recognized from the fact that in Fig. 4.20 the cell rate does not increase from zero until about 6 milliseconds after source 3 has begun transmission.

Also of interest in these plots is the overshoot that the rate of source 3 experiences, and the undershoot that source 1 experiences. This occurs because the source 3 rate starts with a value of B/2, and this value is maintained for a significant period of time. Source 1 and source 2 also have initial rates of B/2, but they already have plenty of RM cells in route back to them, so their rates drop quickly since queue 1 is growing. The feedback information is not received quickly by source 3 since all of its RM cells have to first wait in the queue and then return to switch 1 before they can have a lower rate placed in them. The delay in feeding back the rate information causes the overshoot.

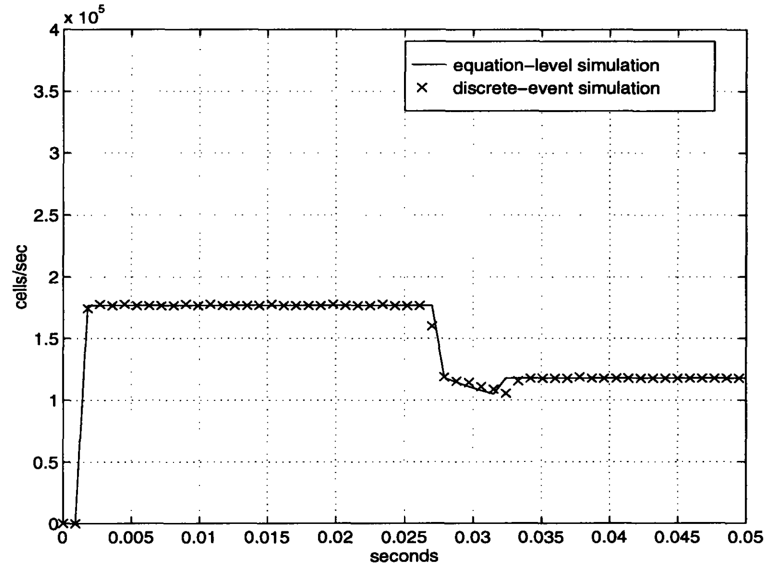

Figure 4.19: Comparison of source 1 rates at switch 2 (source 2 rates are identical)

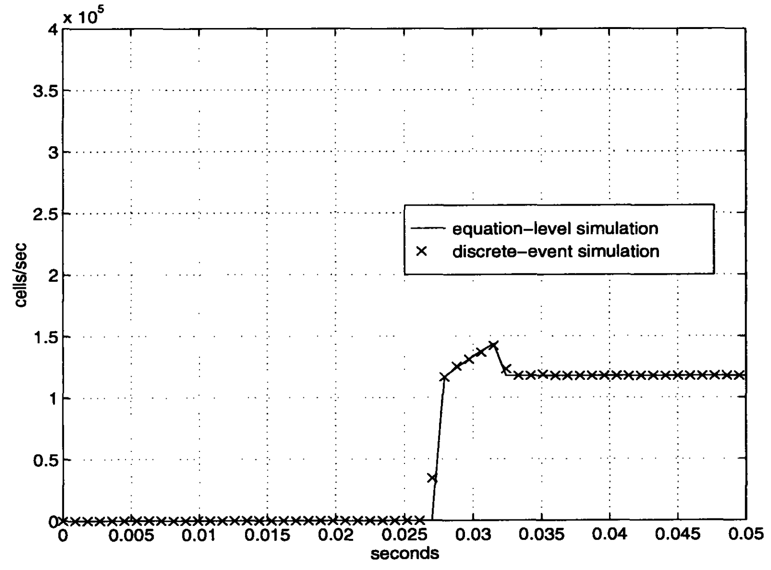

Figure 4.20: Comparison of source 3 rates at switch 2

In both of these figures, the discrete-event and equation-level plots are similar. The $\mathrm{RMS}_{\%}$ value for the source 1 rate at switch 2 is 2.4%, and for source 3 rate at switch 2 it is 3.8%. These values indicate that the model has successfully indicated the appearance of the discrete-event simulation results.

4.6 Conclusions

The original block diagram model used to form equation-level simulation results to be compared with discrete-event simulation results failed to capture two very important elements. One major factor that was lacking in the model was a queueing delay block. If the cells of a source must pass through a non-zero queue in route to their destination, they are delayed by that queue. This can have significant effects on system behavior. The "blending" that occurs anytime sources share a single queue was also previously unmodeled. Blending occurs when different sources share the same output port and the service rate of the queue limits the total rate at which cells can be passed on to subsequent switches. When both of these elements are included in the model, the behavior of the system can be more accurately predicted.

Chapter 5

Controlling Switch

5.1 Introduction

Up to this point, a system with a single switch and a system with two switches have been modeled. Each contributed to the development of a more accurate model of the network. When the transition between these two systems was made, it was discovered that the switches of a system have an on-going influence on one another because of the presence of queueing delays and blending of source cells. As such, it is not possible to de-couple the various switches in a multi-switch network.

A different combination of switches and sources allows discussion of another way in which switches react to one anothers' behavior. When an RM cell is sent back to its source, each switch it passes through has an opportunity to place a source rate value in the RM cell if data cells from that source are contributing to congestion in that switch. If a switch determines that the rate placed in the cell by another switch is less than the value it wishes to feedback, it leaves the cell unaltered. Otherwise, it places its desired source rate in the cell.

In what follows, a switch is referred to as a controlling switch of a source when it provides the most limiting feedback rate information to the source. In the example networks discussed in earlier chapters, switch 1 was the controlling switch for all sources. To model the situation when two or more switches serve as controlling switches in a network, another element is added to the block diagram of the system to indicate the manner in which the switches interact.

5.2 New Dynamics

A new example is shown in Fig. 5.1. In this example, cells from source 1 and source 2 first enter the network at switch 1. Source 3 and source 4 are connected to the network via switch 2. Source 1 cells travel to destination 1, source 2 cells to destination 2, and so forth. A network of this topology allows both queue 1 and queue 2 to grow to non-zero steady-state values. Both switch 1 and switch 2 serve as controlling switches. In this example, source 1 cells compete for an output port of switch 1 with source 2 and for an output port of switch 2 with source 3 and source 4.

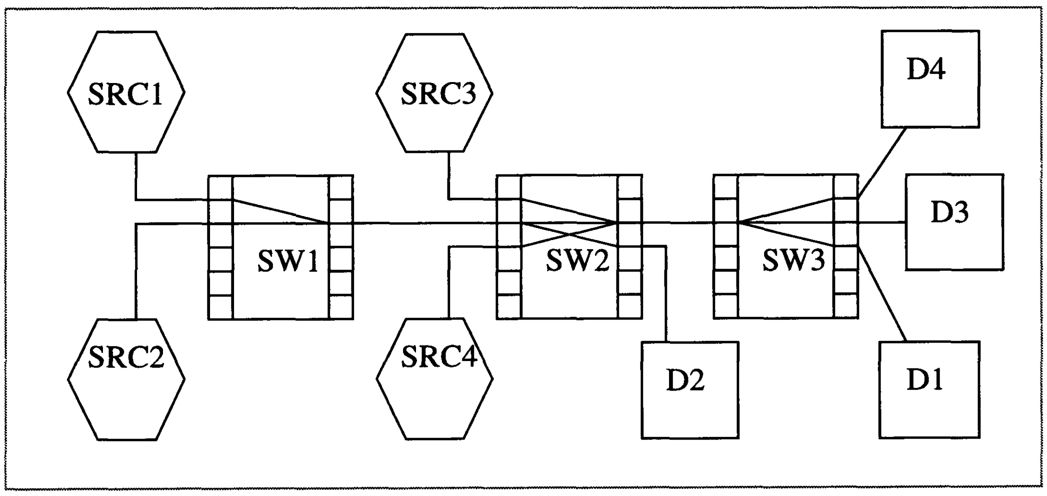

Figure 5.1: Topology to introduce the *min* function

In order to model this system behavior it is necessary to introduce a new type of function. This function models the fact that every time an RM cell passes through a switch on the way back to the source that created it, the switch checks the value stored in the RM cell which corresponds to the feedback rate. The switch also calculates a feedback rate that it wishes to send to the source based on the size of its queue. If the value already in the RM cell is less than the value calculated at the switch, no change is made. If the value calculated at the switch is less, that value is placed in the RM cell as the new feedback rate.

This process occurs in every switch on the way back to the source. The lowest rate calculated by any of the switches is the rate that is actually fed back to the source. The *min* function used in this thesis represents the selection of the lowest rate that any switch suggests.

5.3 Improved Model

Before analysis is attempted, it is useful to consider the role of the *min* function when the system response is broken up into parts; namely, the transient response and steady-state response. During transients, the *min* function must actively select the lowest rate fed back to each switch, and is, in general, time varying. In steady-state, the *min* function produces a constant effect.

A new block diagram structure corresponding to a general switch is shown in Fig. 5.2. This model incorporates the effect of the *min* function. It shows that before the new rate is set, the rates fed back by all switches for each of the sources must be compared to find the minimum value. In this figure, feedback rates from all of the switches through which the cells of a source pass are represented in the form $\tilde{r}_k^{(i)}[n]$, where i is the switch feeding back the rate and k is the source to which the rate is fed back. Every switch, i, in which a source, k, contributes to congestion provides a feedback rate to the *min* function of that source. The number of sources to which the switch shown in Fig. 5.2 sends feedback is represented by L.

Figures 5.3 and 5.4 display models that correspond specifically to switch 1 and switch 2, respectively, for the example illustrated in Fig. 5.1. Source 2 contributes to congestion only in switch 1. As such, the *min* function always selects $r_2^{(1)}[n] = \tilde{r}_2^{(1)}[n]$ which is reflected as unity gain in the feedback path for $r_2^{(1)}[n]$ in Fig. 5.3. Likewise, Fig. 5.4 illustrates that switch 2 is the controlling switch for source 3 and source 4. As in Fig. 5.3, switch 1 and switch 2 control source 1 during different intervals. Figure 5.4 reflects this fact by the introduction of the *min* function block. The rate blender algorithm and queueing delay must also be included in the model for switch 2 since these effects occur when source 1 cells pass through queue 1.

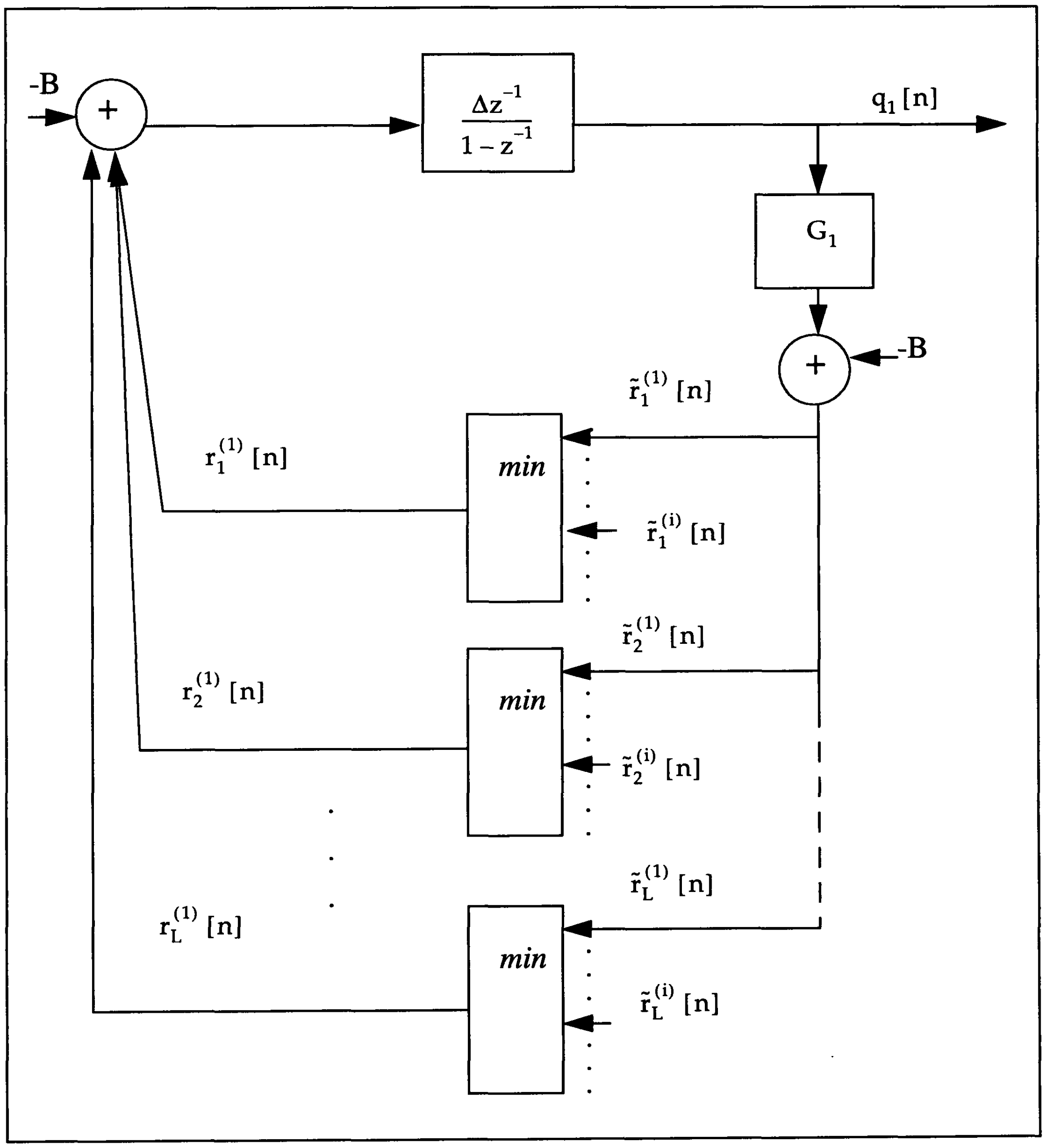

Figure 5.2: General switch model

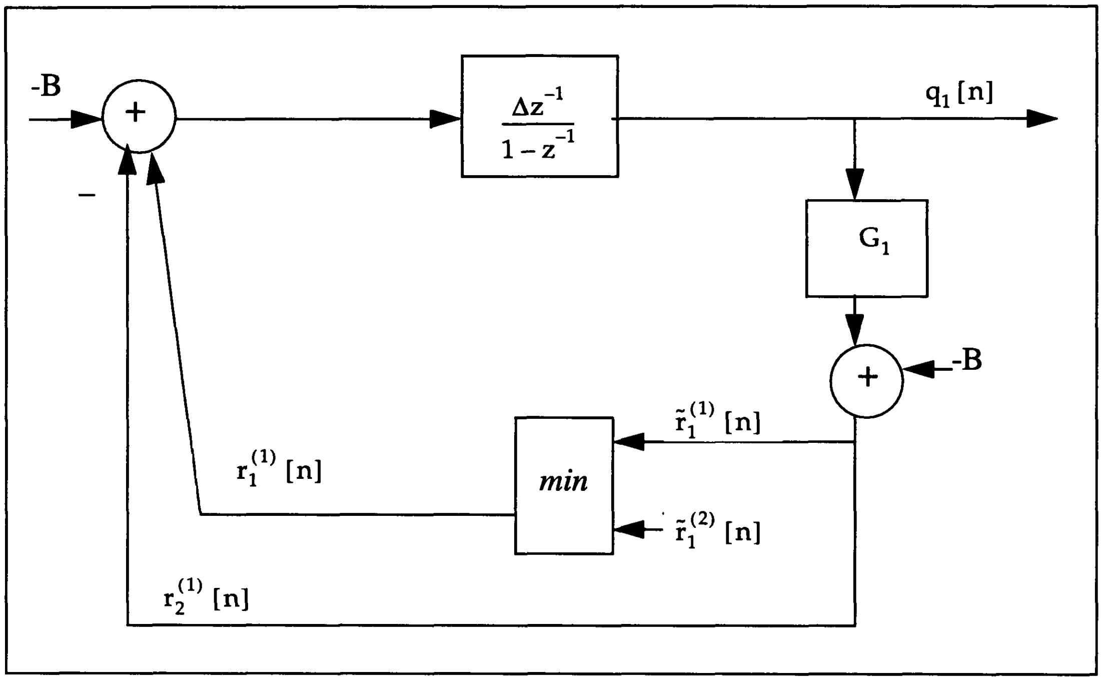

Figure 5.3: Switch 1 model for the example in Fig. 5.1

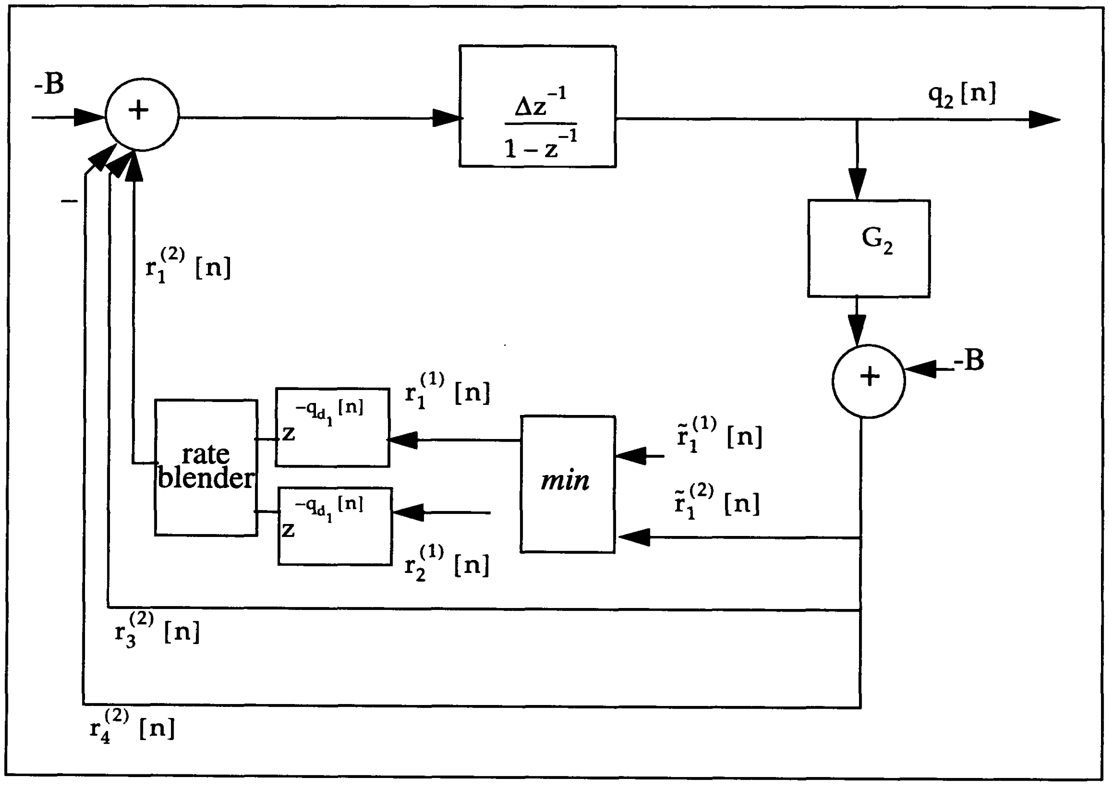

Figure 5.4: Switch 2 model for the example in Fig. 5.1

5.4 Analytical Predictions

The above block diagram models yield the same form of loop gain equation as is given by Eq. (3.12). The Bode plots of Figs. 3.10 and 3.11 indicate that the system response will be without oscillation or overshoot. From Eq. (2.12), the steady-state value of the size of queue 2 is 2359 cells. From Eq. (2.11), for source 1, source 3 and source 4, the steady-state value of the rate is 1.179×10^5 cells/sec. Until switch 1 receives cells at a total rate equal to the bandwidth, it increases the cell rate that it feeds back to source 1 and source 2. Because the steady-state value of the source 1 rate is 1.179×10^5 cells/second, the cell rate fed back by switch 1 increases to 2.359×10^5 cells/second, and that is the rate at which source 2 transmits in steady-state. The steady-state value of the queue 1 size can then be calculated by use of Eq. (2.10) to be 1179 cells.

The equations which describe the responses of the queues and the source rates in this chapter require consideration of the role of the *min* function, the queueing delay, and the rate blending in addition to the effect the rates and queues have on one another in the absence of these elements. Due to the complexity of the combination of these equations, it is necessary to obtain these results algorithmically.

The algorithm has each source rate step up to its start up value at the given time. Every Δ seconds, the queue size is calculated. That size is based on the rates at which cells have entered the queue over the past interval and the previous queue size according to Eq. (2.8). When delay such as queueing delay exists, Eq. (2.8) must be adjusted. Equation (4.1) indicates the amount of queueing delay that source cell rates encounter. This value can be used to determine which past interval should be consulted to determine the source cell rates now entering the queue being examined. In the examples of this thesis, queue 1 is the first queue so cells that approach it have not already encountered queueing delay. Cells approaching queue 2 have passed through queue 1, so they are delayed by an amount that may be calculated from Eq. (4.1). These calculations determine the rate at which cells approach a queue.

To determine the rate at which source cells depart the queues, the rate blending algorithm is implemented. This algorithm takes into account the mixing of cells in a queue and the fact that the queue limits the rate at which cells may pass through a switch.

Finally, the *min* function must be implemented whenever more than one switch forms a non-zero queue. Source rate feedback values are calculated at every switch that the cells of a source traverse. These values are calculated based on the queue size the switch calculates from Eq. (2.9). The algorithm selects the lowest rate calculated by these switches, and makes that value the next source rate. In this way, the equation-level simulation results are produced.

5.5 Experimental Results

A set of equation-level and discrete-event simulation results can be gathered for the case in which no start up delays are used in the system of Fig. 5.1 in order to determine the accuracy of the modeling of the system. These results are shown in Figs. 5.5 - 5.8. Figure 5.5 shows the equation-level simulation and discrete-event simulation results for queue 1, and Fig. 5.6 shows the equation-level and discrete-event simulation results for queue 2. Figure 5.7 shows the rates for source 1, source 3 and source 4 superimposed since cells are generated at all of these sources at the same rate, and each receives RM cell feedback information at the same instant. Figure 5.8 shows the rates for source 2, which differ from the other three source rates since source 2 is controlled by switch 1 whereas the other sources are controlled by switch 2.

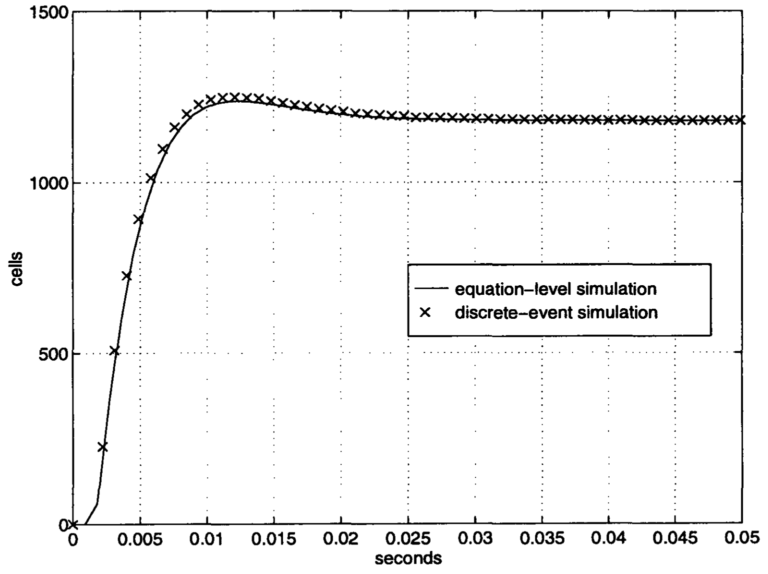

Figure 5.5: Queue 1 responses for the four source system of Fig. 5.1

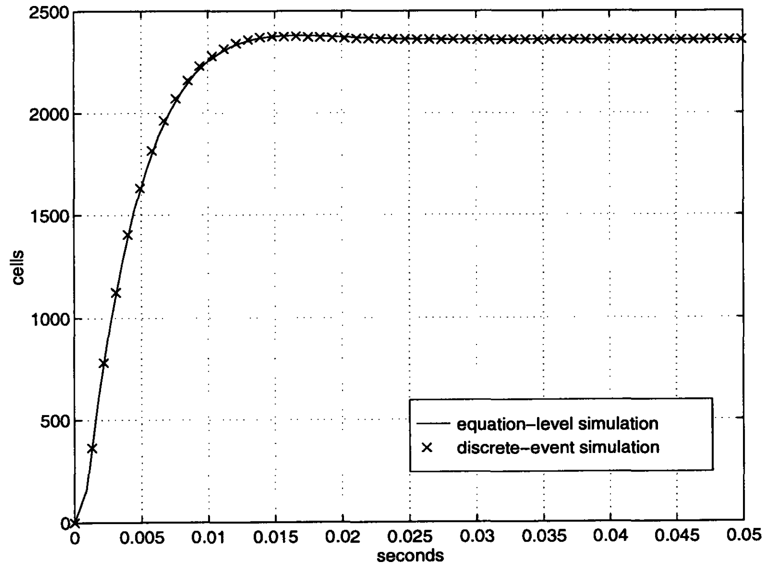

Figure 5.6: Queue 2 responses for the four source system of Fig. 5.1

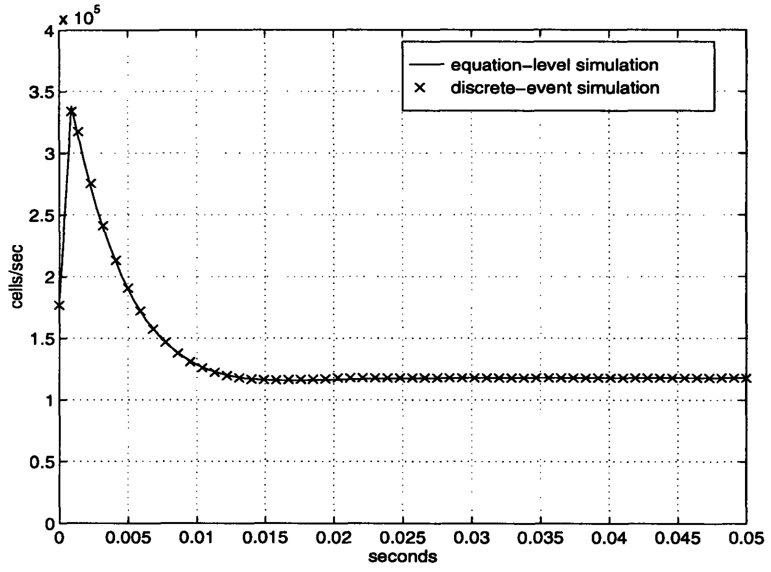

Figure 5.7: Source 1, 3 and 4 responses for the four source system of Fig. 5.1

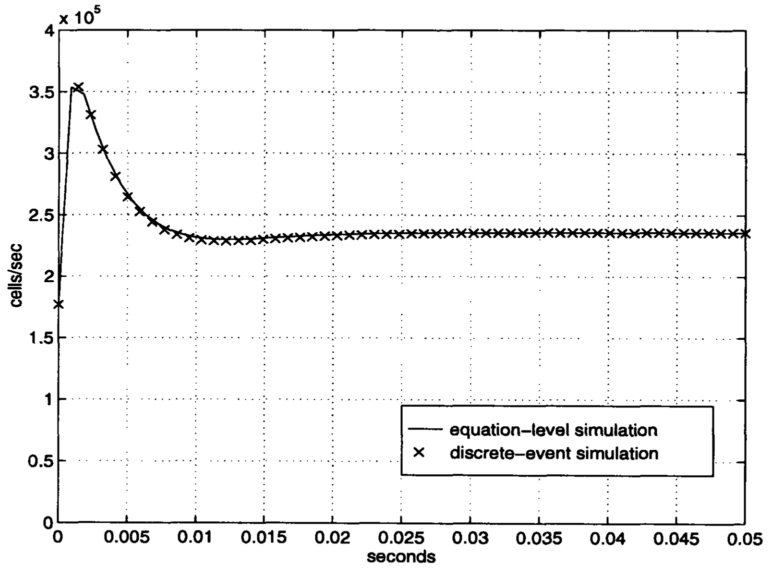

Figure 5.8: Source 2 responses for the four source system of Fig. 5.1

These figures serve as an example in which *min* function behavior is observed. Source 1 receives rate information that is the minimum of the rate fed back by switch 1 and switch 2. The larger the queues in these switches, the lower the source rate that is fed back. Source 3 and source 4 receive feedback information from switch 2 only.

In this example, since three sources vie for the same output port of switch 2 and just two vie for the same output port of switch 1, switch 2 is the more rate-limiting switch. This fact is represented in Fig. 5.9 which shows that queue 2 is always larger than queue 1. Therefore, according to Eq. (2.11), the three sources which receive rate information from switch 2 have the same cell rate of $B/3$. As explained in Section 5.4, source 2 will have a steady-state rate of $2B/3$.

These values are verified by the discrete-event simulation results. Switch 2 generates a feedback cell rate of $B/3$ for sources 1, 3 and 4. Switch 1 sets its allowable cell rate to $2B/3$, since one of the sources is transmitting at a rate of $B/3$ and the switch has a total service capacity of B. Therefore, source 2 settles to a data cell rate of $2B/3$. The $RMS_{\%}$ values for the rate plots are 2.7% for source 1, source 3 and source 4 rates and 1.1% for the source 2 rate. This is a further indication of a match between discrete-event simulation and equation-level simulation results.

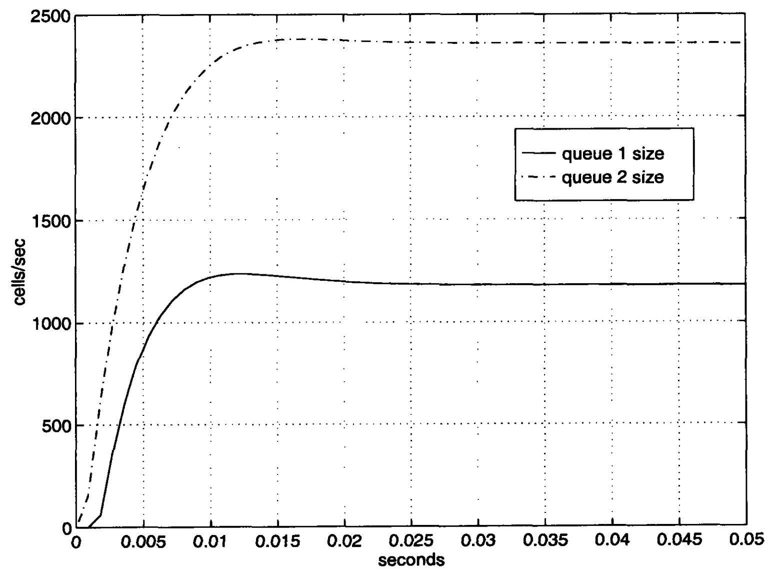

Figure 5.9: Comparison of queue 1 and queue 2 sizes

Figure 5.9 indicates that beyond time $t = 0$, the queue in switch 2 is larger than the queue in switch 1. Due to this fact, switch 2 always dominates the *min* function, the output of which is fed back to source 1. That is, from the onset, source 1 is always under the control of switch 2 since it is always the more limiting switch. From the development in Section 5.4, it is predicted that queue 1 will take on a value of 1179 cells in steady-state and queue 2 will have 2359 cells in steady-state. These are the values the queues take on in steady-state in the discrete-event simulations of Figs. 5.5 and 5.6. The $\mathrm{RMS}_{\%}$ values expressing the difference between the equation-level simulation and the discrete-event simulation for the queue plots are 6.3% for queue 1 and 2.1% for queue 2.

In the next example, source 4 has a start up delay of 0.005 secs with respect to the other sources. With this change, the time varying nature of the *min* function is demonstrated. The *min* function picks the feedback from switch 1 for a while, then it chooses the feedback from switch 2.

As shown in Section 5.4, the steady-state values of queue 1 and queue 2 are predicted to be 1179 cells and 2359 cells, respectively. The steady-state source rates are predicted by Eq. (2.11) to take on values of 1.179×10^5 cells/second for source 1, source 3, and source 4 and 2.539×10^5 cells/second for source 2. From Eq. (4.1), queue 2 contains about 700 cells at the time when source 4 begins to transmit. This indicates a delay due to queueing of approximately 2 milliseconds. Therefore, the first time at which source 4 will be able to change its cell rate from its initial cell rate of $B/2$ is approximately at $t = 0.007$ seconds.

The actual and predicted simulation results allow a more detailed evaluation of the capabilities of the block diagram model. Figures 5.10 and 5.11 show the equation-level simulation and discrete-event simulation queue responses. Figures 5.12 - 5.15 show the rate responses.

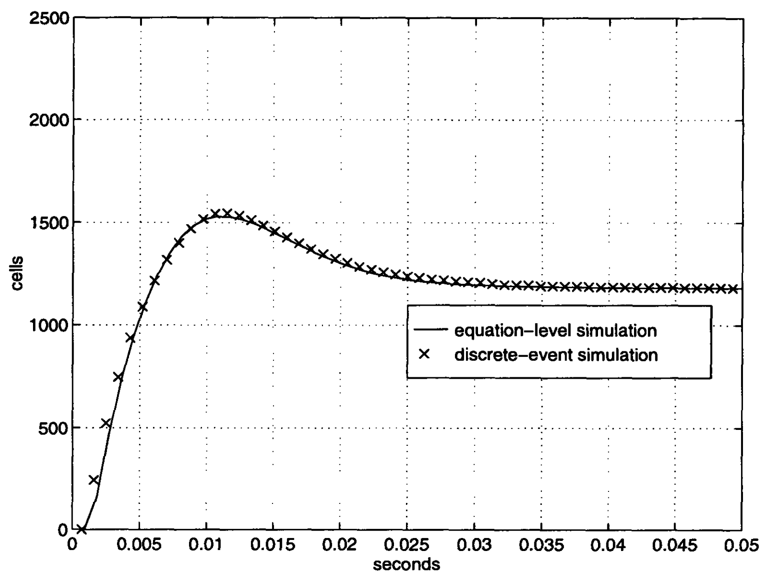

Figure 5.10: Queue 1 sizes when source 4 has a short start up delay

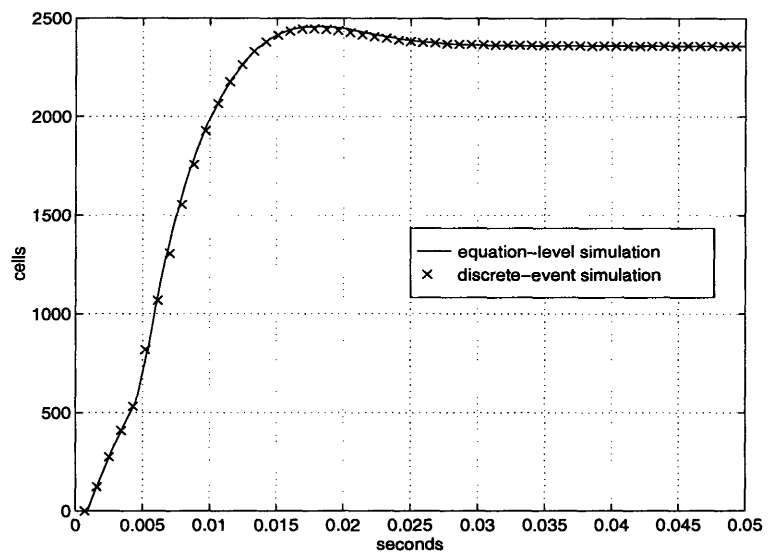

Figure 5.11: Queue 2 sizes when source 4 has a short start up delay

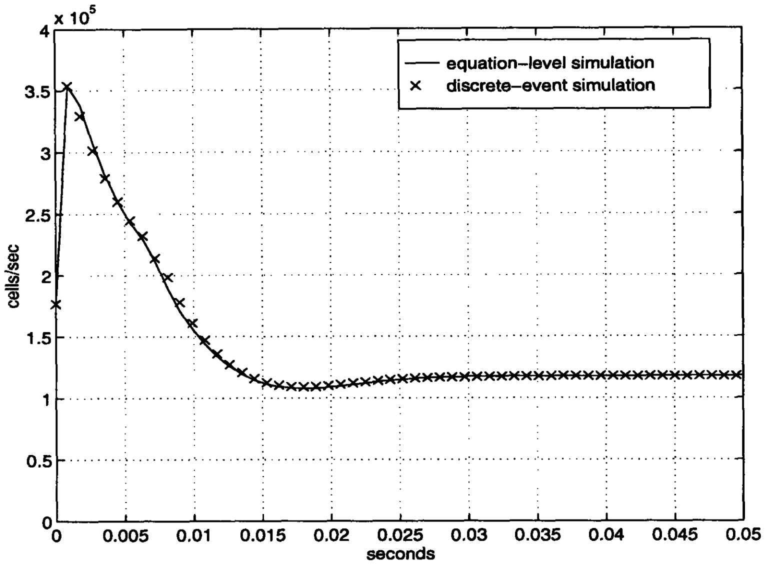

Figure 5.12: Source 1 rates at switch 1 when source 4 has a short start up delay

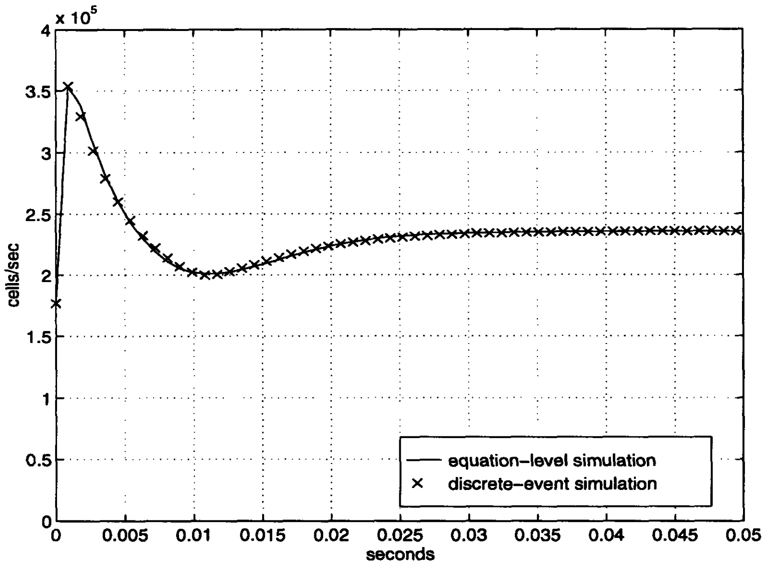

Figure 5.13: Source 2 rates at switch 1 when source 4 has a short start up delay

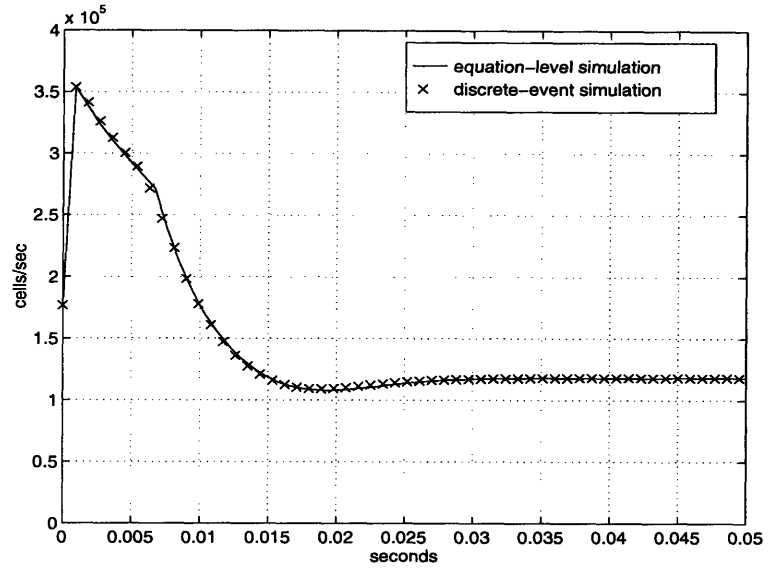

Figure 5.14: Source 3 rates when source 4 has a short start up delay

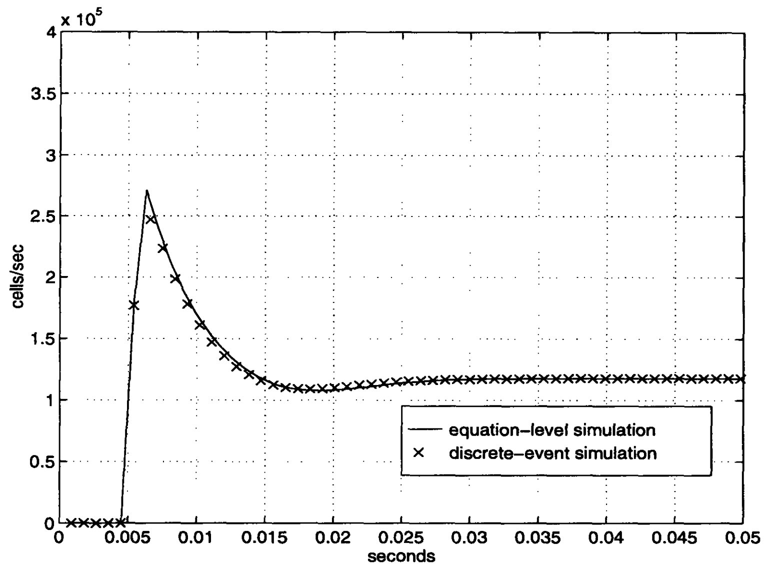

Figure 5.15: Source 4 rates when source 4 has a short start up delay

A few points are notable about this example. First, the *min* function is now more active. This fact becomes clearer by seeing the two queues in direct comparison as is shown in Fig. 5.16. At first queue 1 is larger and therefore controls source 1. Then, shortly after source 4 starts to transmit, queue 2 becomes larger than queue 1 and switch 2 controls the behavior of source 1. The impact of this result can be seen by looking at the source rate plots. Source 1 is different from source 3 until t = 0.007 seconds because it is under the control of a different switch. Source 2 is approaching B/2 at first because source 1 is transmitting data cells at the full rate allowed by switch 1. This source 1 cell rate forces switch 1 to set a feedback cell rate of B/2. Once switch 2 gains control of source 1 and forces its rate to B/3, switch 1 allows a higher cell rate for source 2.

The transmission rate of source 4 during the interval just after transmission begins is noteworthy. All sources initially transmit at B/2 and wait for feedback information. When source 4 starts up, its first RM cell must wait in queue 2. A delay of about 0.002 seconds from t = 0.005 seconds to t = 0.007 seconds before source 4 starts tracking the rates of source 1 and source 3 is illustrated in Fig. 5.15. This amount of delay matches the prediction made based on the size of queue 2.

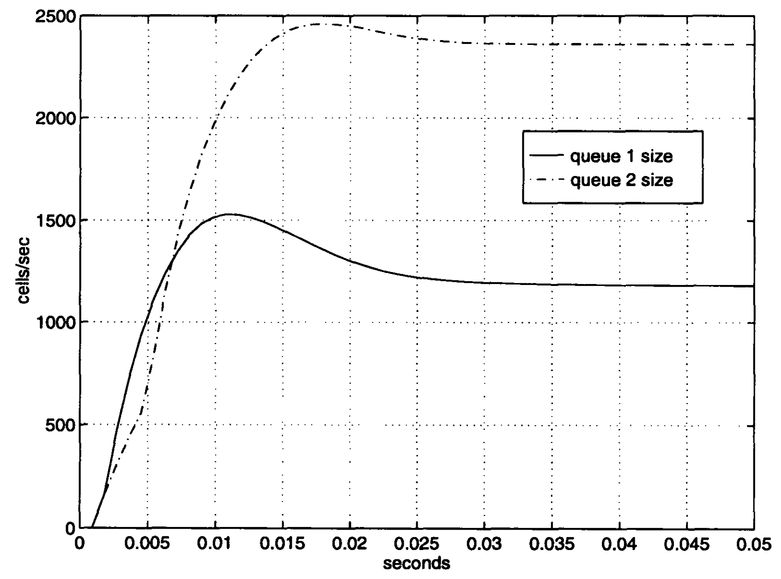

Figure 5.16: Comparison of queue sizes

Several important behaviors that are displayed in the discrete-event output plots of this example have been predicted by the equation-level simulation plots. The queue sizes and source rates take on steady-state values that are approximately equal to the values predicted earlier. These $\mathrm{RMS}_{\%}$ values that correspond to the Figs. 5.10 - 5.15 may be calculated. For queue 1 and queue 2, the values are 4.5% and 2.8%, respectively. The $\mathrm{RMS}_{\%}$ values are 1.6% for source 1, 1.3% for source 2, 2.2% for source 3 and 0.9% for source 4.

The final verification of the modeling of the *min* function can be made by analysis of a case which involves a longer start up delay. In this example, source 4 is delayed in start up by 0.05 seconds relative to the other three sources. According to Eq. (2.11), the predicted steady-state rate values are 1.179×10^5 cells/second for sources 1, 3, and 4, and 2.358×10^5 cells/second for source 2. Section 5.4 explains that the expected steady-state queue values for queue 1 and queue 2 are 1179 cells and 2359 cells, respectively. The step responses of the system are shown in Figs. 5.17 - 5.22.

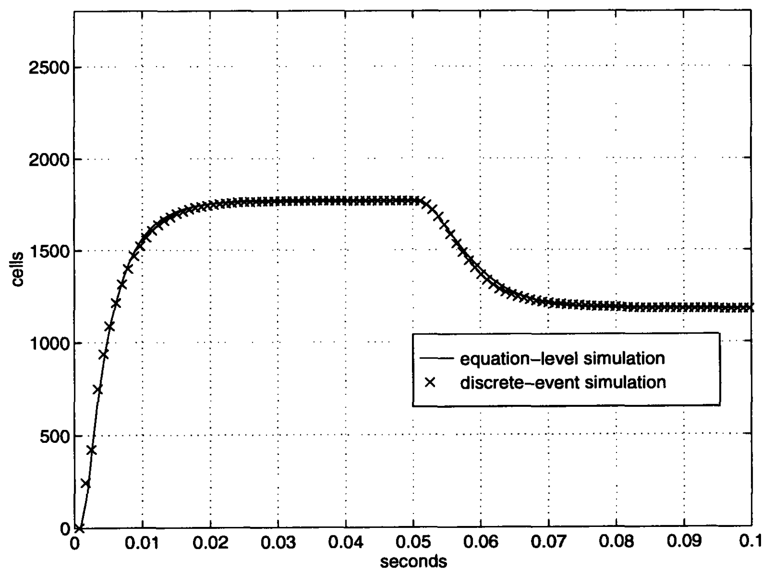

Figure 5.17: Queue 1 sizes for system in which source 4 has a long start up delay

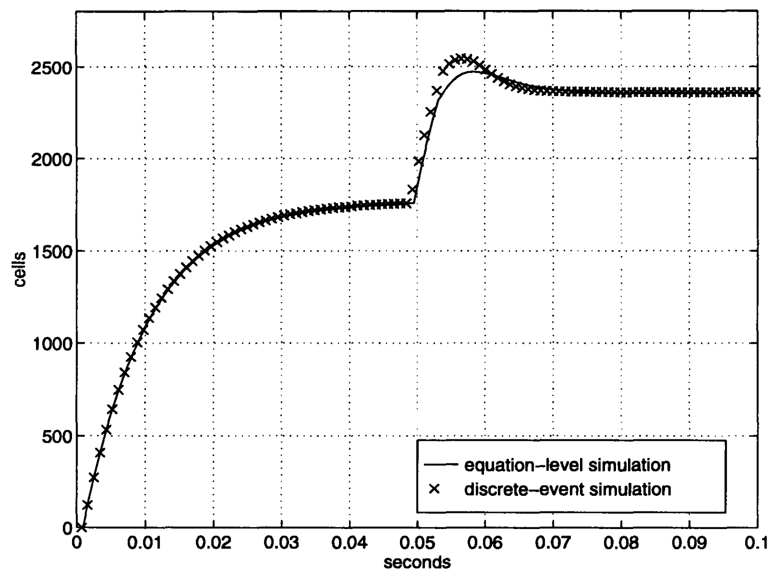

Figure 5.18: Queue 2 sizes for system in which source 4 has a long start up delay

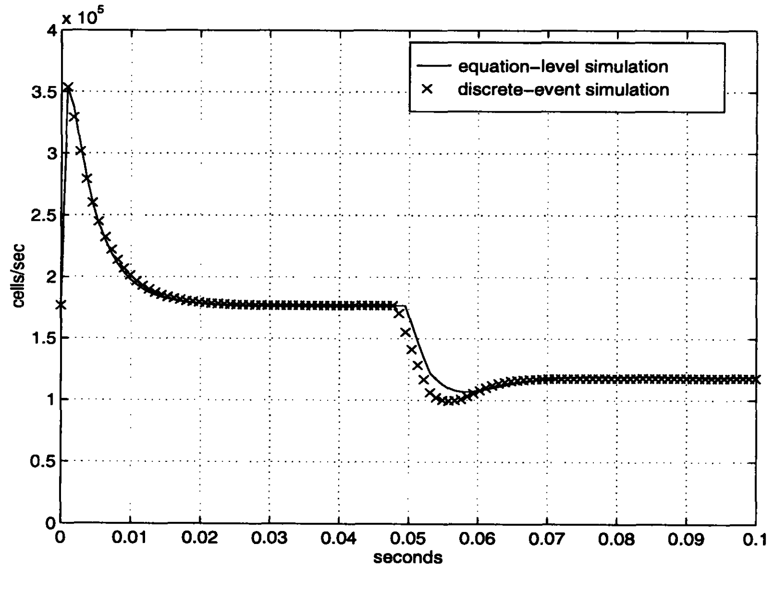

Figure 5.19: Source 1 rates for system in which source 4 has a long start up delay

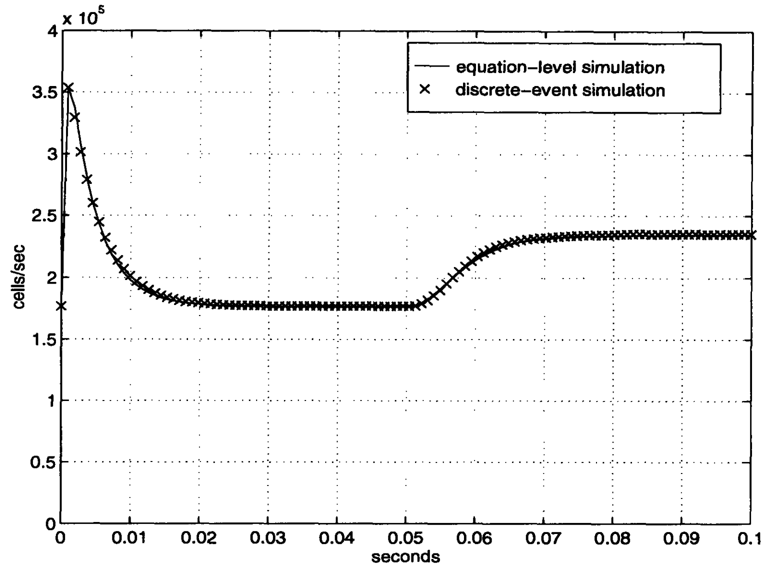

Figure 5.20: Source 2 rates for system in which source 4 has a long start up delay

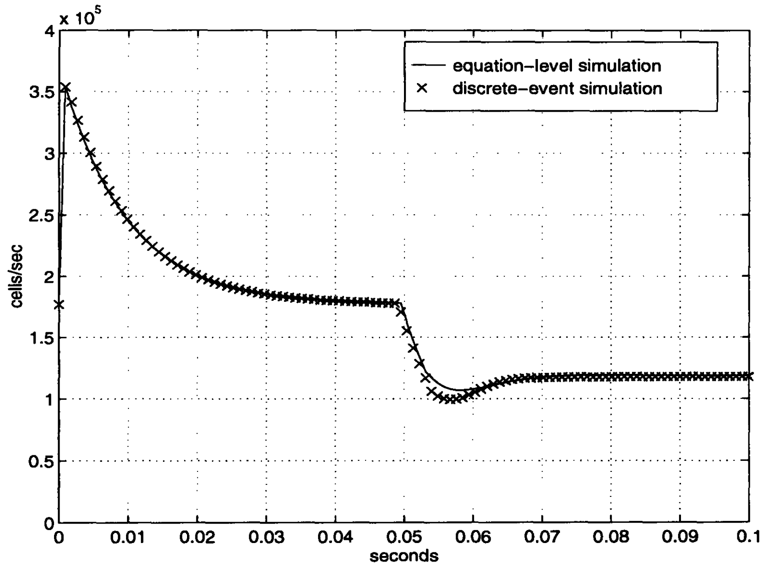

Figure 5.21: Source 3 rates for system in which source 4 has a long start up delay

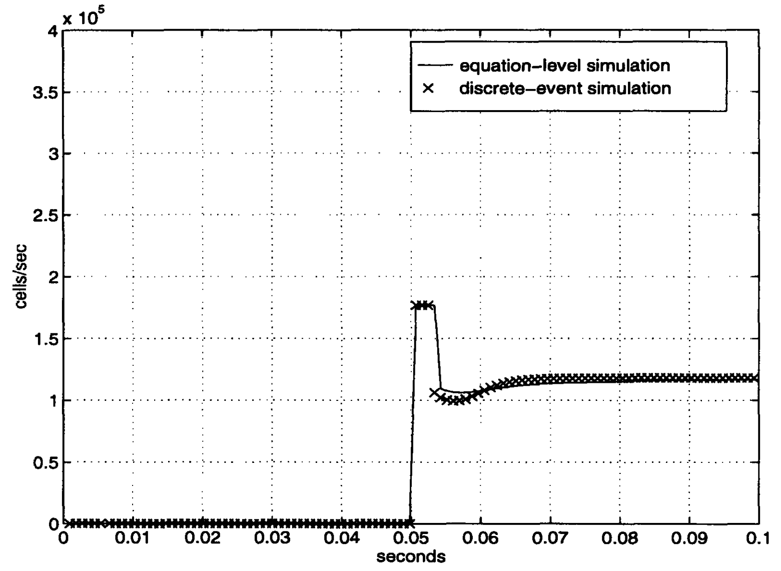

Figure 5.22: Source 4 rates for system in which source 4 has a long start up delay

Like the previous example, this example illustrates how the *min* function selects feedback information from queue 1, then selects information from queue 2 to maintain proper operation of the system. Figure 5.23 shows a comparison of queue 1 and queue 2. As expected, the *min* function selects the minimum of the two source rate suggestions that source 1 receives. Both queues reach the same value before source 4 begins to transmit because they both receive cells from the same number of sources. When source 4 starts up, queue 2 increases in size since cells from three sources are now entering this queue. As a result, source 1, source 3, and source 4 receive slow down feedback information since they are contributing to the most significant congestion problem in the network. Queue 1 decreases in size since the rate of source 1 is decreasing. Queue 1 is starved because source 1 is producing cells at a rate below the rate switch 1 is feeding back. This causes switch 1 to increase its feedback rate until it reaches $2B/3$. As a result, source 2 is permitted to increase its rate to this value.

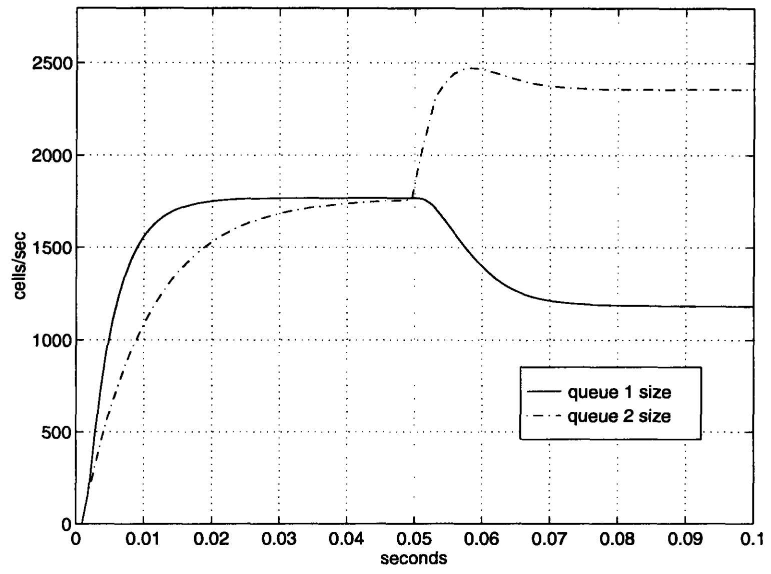

Figure 5.23: Comparison of queue 1 and queue 2

Notable about this system is the fact that the queues have different trajectories after the first three sources have been initialized. During this interval, both switches receive data cells from two sources. Early models[10] of the system did not take into account the influences that switches have on one another in a system in which several non-zero queues exist. Therefore, early models did not predict that these queues would have different trajectories. The new models of the system provide an explanation for the fact that the queues grow at different rates in spite of the fact that both switches receive cells from two sources and all sources are transmitting at the same rate. Source 1 cells are blended with source 2 cells in queue 1, and therefore the rate of source 1 cells flowing into switch 2 is less than the rate of generation of source 1 cells. Therefore, queue 2 grows more slowly at first. As expected, the two queues reach the same steady-state value of 1769 cells predicted by Eq. (2.12). This occurs around the time of 0.02 seconds.

Also notable is that in Fig. 5.22, there is a long time span after time $t = 0.05$ seconds for which the source 4 rate remains at its initial value of 1.769×10^5 cells/second. Since the feedback information must come in an RM cell, and the RM cell generated by source 4 must wait in a queue of more than 1500 cells, the data cell rate of source 4 may not change for approximately 0.004 seconds. This result indicates the ongoing role of queueing delay. An effort could be made to mitigate this result by improving selection of initial cell rate.

The $RMS_{\%}$ values may be calculated for this example to verify the accuracy of the model. For queue 1 the value is 3.5%, for queue 2 it is 2.3%, for source 1 rate it is 1.3%, for source 2 rate it is 0.1%, for source 3 rate it is 0.7%, and for source 4 rate it is 5.6%. Many of the small differences between the plot pairs may be attributed to the limitations of working with a discrete-time system. The block diagram analysis provides a useful, direct approach to predicting the system behavior.

5.6 Conclusions

The *min* function is an important part of the network model in a multi-switch system. The *min* function selects the least of the cell rates that the relevant switches indicate, and transmits that value to the proper source. Whenever the *min* function is operating in a system, it is impossible to completely de-couple the influences of the various switches. When it and all of the other newly discovered dynamics are included in the model, the behavior of the system can be accurately predicted.

Chapter 6

A Complete Example

6.1 Introduction

In this chapter, the affect of adding propagation delay to the example network of Chapter 5 is examined. The network analyzed in Chapter 5 is repeated below as Fig. 6.1 for convenience. This experiment provides an example in which all of the dynamics included in Chapters 3 - 5 are included; namely, queueing delay, rate blending, the *min* function, and propagation delay. Predictions of the system response are made and compared with discrete-event simulation results to determine how closely this more complete example network is modeled by a discrete-time control system.

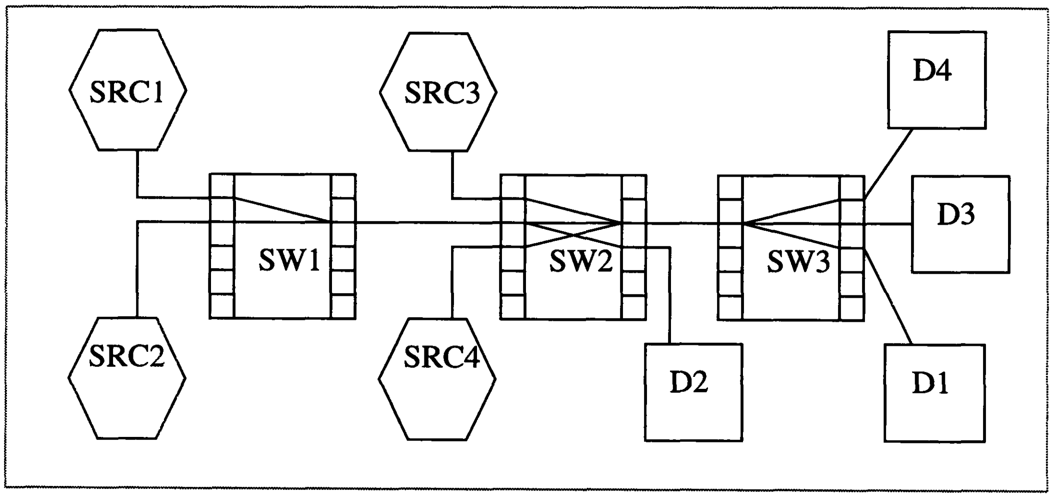

Figure 6.1: Example network for analysis

For the example analyzed in this chapter, 1.5Δ seconds of delay are placed on each of the four source to switch links. This yields a total of 3Δ seconds of round-trip delay for each source. This value is the maximum integer number of Δ's that can be placed on all source to switch links while allowing the system to remain stable. When propagation delay is added to the system, sources are unable to respond to rate control information as quickly as without delay. If the delay is long enough, the lack of timely response to feedback causes overshoots and oscillations in the source rates and the queue sizes of the system. This effect shall be explored and modeled in this chapter.

6.2 Analytical Predictions

Block diagram models which combine the results from Chapter 3 with those from Chapter 5 are shown as Figs. 6.2 - 6.3. These block diagram models are based on the model of Chapter 5, with the addition of blocks corresponding to the propagation delay added to the links. The exponents of the propagation delay terms are d_i, $i \in [1, 4]$, which each take on a value of 3 in the example explored in this chapter. The gains G_1 and G_2 are chosen to be 100. The above block diagram models yield the same form of loop gain equation as is given by Eq. (3.12), with corresponding Bode plots illustrated in Figs. 3.10 and 3.11.

As indicated in Chapter 3, the addition of delay such as queueing delay or propagation delay can only degrade the stability of the system. With this delay added to the base system, the system loses its robust stability, and the response of the system is expected to contain oscillations. The greater the queueing delay added to the base model, the greater the oscillations in the system step response.

As with the examples of Chapter 5, the complexity of the combinations of the equations in this chapter requires an algorithmic implementation of the equation-level simulation. The development of Section 5.4 holds for the example network of this chapter, with the addition of the effect of link delay. As with the queueing delay discussed in Section 5.4, the added link delay necessitates determining the past interval at which the cells currently entering the queue were produced. Based on the fixed value of the link delay and the calculated value of the queueing delay, it is possible to determine the rate at which cells of a source approach any switch in the system. To find the correct rate value, all that is necessary is to shift the rate index so that the proper interval is examined. This shift due to link delay is constant since the link delay is constant in this chapter. This is the algorithm that produces the equation-level simulation results in this chapter.

Figure 6.2 displays the switch 1 block diagram model for the example of this chapter. It takes virtually the same form as the switch 1 model shown in Fig. 5.3. The only difference is the addition of link delay in Fig. 6.2. Figure 6.3 shows the switch 2 block diagram model. Figure 6.3 is Fig. 5.4 with added link delay.

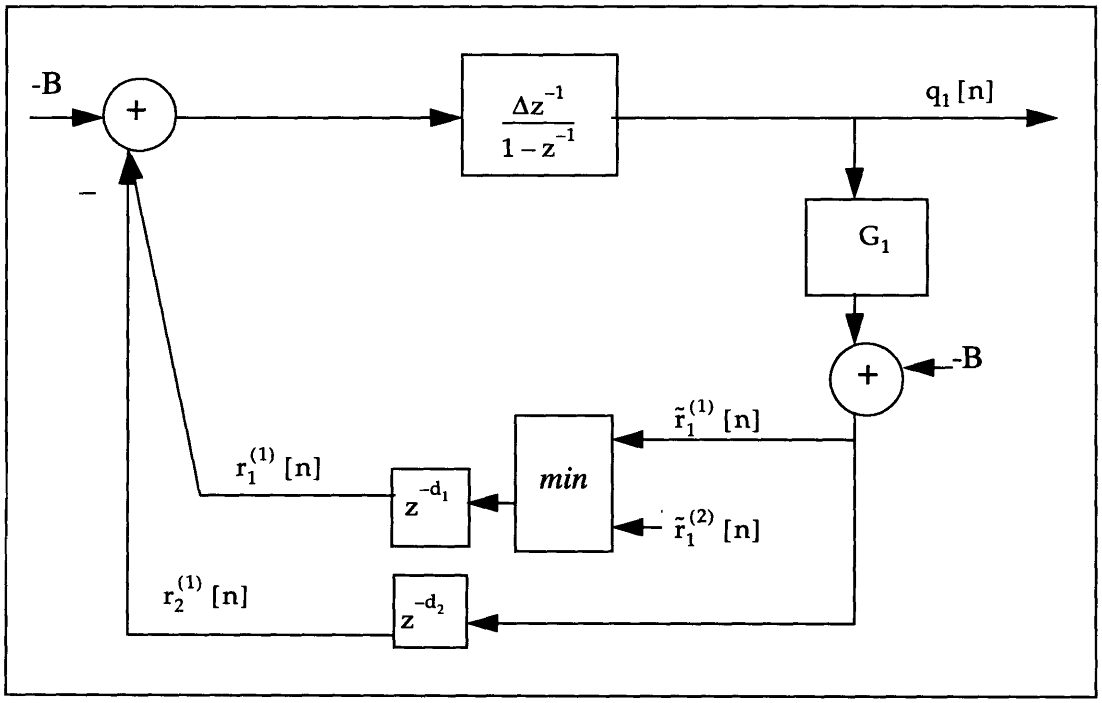

Figure 6.2: Switch 1 model for the example in Fig. 6.1

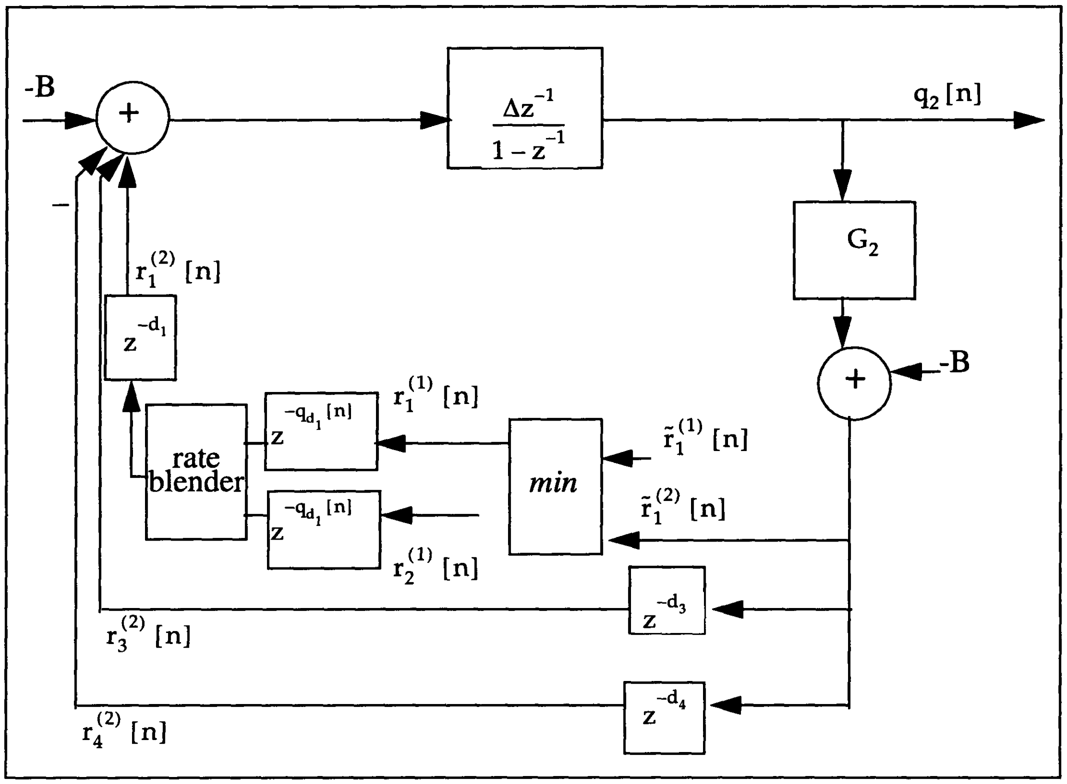

Figure 6.3: Switch 2 model for the example in Fig. 6.1

6.3 Experimental Results

Figures 6.4 - 6.7 display the equation-level and discrete-event simulation results for the example network of Fig. 6.1. The equation-level simulation results are calculated based on the block diagram of Fig. 6.2 which takes into account the *min* function, the rate blending, the queueing delay and the 3Δ round-trip propagation delay for each source. Figure 6.4 displays the equation-level and discrete-event step response results that correspond to cell rates produced by sources 1, 3, and 4. Figure 6.5 shows the cell rate produced by source 2. Figure 6.6 shows the equation-level and discrete-event results of the step response of queue 1. Finally, Fig. 6.7 shows the step response of queue 2.

In this example, the cell rates produced by source 1, source 3, and source 4 are all identical since all three sources receive the same feedback information. Source 1 contributes to the congestion in both queue 1 and queue 2, but the size of queue 2 is always larger than the size of queue 1, so switch 2 always provides the smaller feedback rate to the *min* function and the *min* function always selects the feedback from switch 2. Since switch 2 controls three sources, the steady-state cell rate is predicted by Eq. (2.11) to be 1.179×10^5 cells/second. This is equivalent to the experimental value shown in Fig. 6.4.

The cell rate produced by source 2 is determined by feedback from switch 1 since queue 1 is the only queue in which source 2 cells contribute to congestion. Although switch 1 receives cells from 2 sources, source 1 is constrained by switch 2 to take on the value of 1.179×10^5 cells/second. Therefore, source 2 is expected to take on a steady-state value of 2.359×10^5 cells/second. This expected steady-state value of source 2 is matched experimentally in Fig. 6.5.

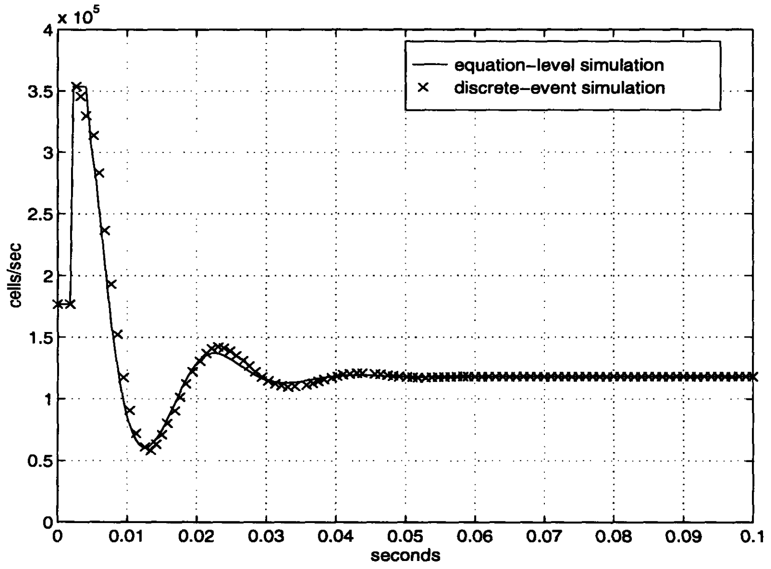

Figure 6.4: Source 1, 3 and 4 rates when all sources have link delay

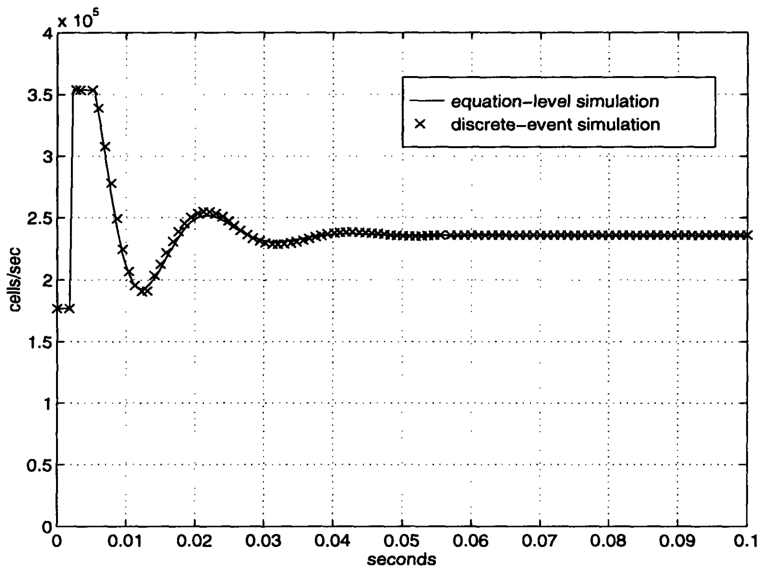

Figure 6.5: Source 2 rate when all sources have link delay

The $\mathrm{RMS}_{\%}$ values for Figs. 6.4 and 6.5 are 1.5% and 0.4%, respectively. These values quantitatively indicate that the experimental results which are derived from the block diagram model of the network are very good matches to the discrete-event simulation data. The appearance of the plots verifies this fact. The plots appear to have the same period of oscillation, the same settling time and have about the same overshoot and rise time.

The modeling of both queue 1 and queue 2 also appear to be adequate. The $\mathrm{RMS}_{\%}$ values for Figs. 6.6 and 6.7 are 3.0% and 1.7%, respectively. The actual steady-state values of 1179 cells for queue 1 and 2359 cells for queue 2 are as predicted by the development in Section 5.4.

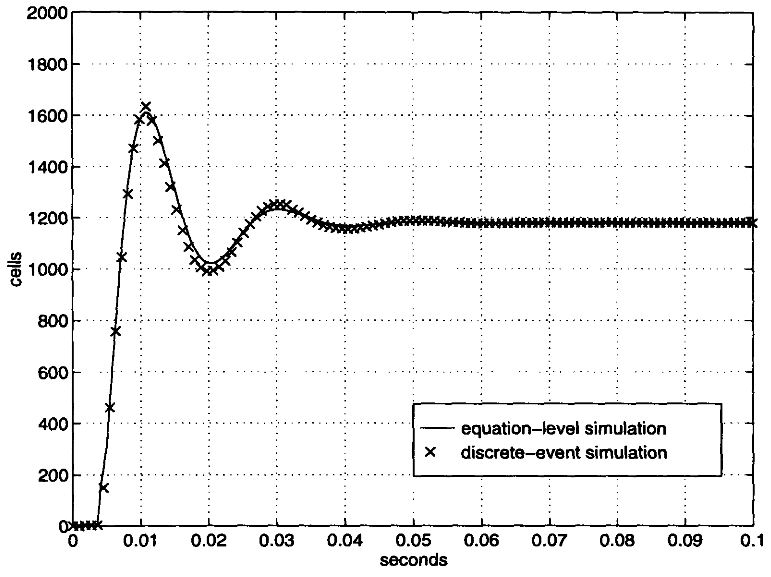

Figure 6.6: Queue 1 when all sources have link delay

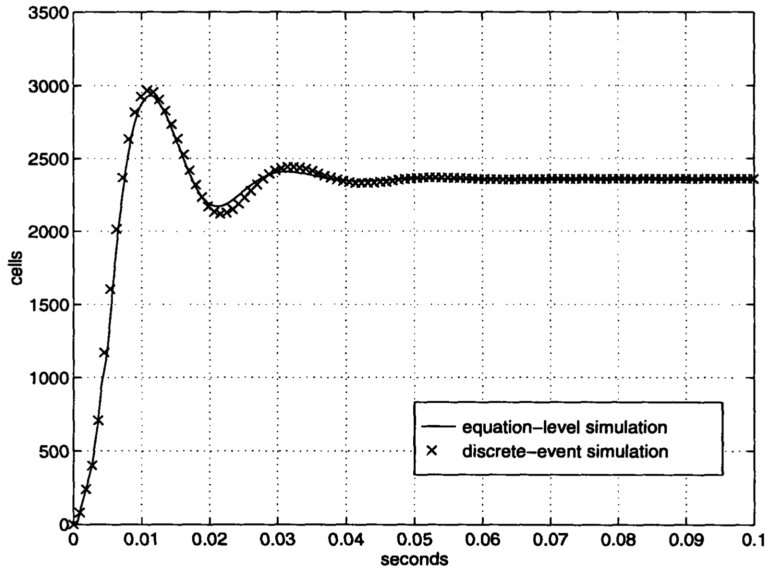

Figure 6.7: Queue 2 when all sources have link delay

In this example, the role of cell blending is especially significant. As indicated by the analytical predictions of the previous section, the system is operating near the boundary of instability due to the presence of several Δ units of both queueing delay and propagation delay. As indicated in Chapter 4, the blending of cells from source 1 and source 2 improves the stability of the system. Cell blending reduces the rate at which source 1 cells enter switch 2. When source 1 and source 2 are in sync, the highest rate at which source 1 cells may enter switch 2 is $B/2$. Therefore, switch 2 is never flooded with the maximum possible number of cells per second which is B. The number of cells entering queue 2 has been limited by the presence of switch 1. Since the total number of cells entering queue 2 is less than it would be if source 1 cells did not have to pass through switch 1, queue 2 achieves a smaller peak than if blending where not present. A smaller queue 2 overshoot means fewer oscillations in the step response.

6.4 Conclusions

A complete block diagram model of the network of Fig. 6.1 allows prediction of the behavior of the multi-switch system even with added propagation delay. The block diagram can be converted to Bode plots and an algorithm relating current values of the rates and the queue sizes to previous values of rates and queue sizes. System step responses can be predicted from this information. Since the equation-level simulation and the discrete-event simulation results match so well, the block diagram models of Figs. 6.2 and 6.3 represent the system well. All of the most significant facets of the behavior of a multi-switch system have been discovered and modeled.

Chapter 7

Conclusions

7.1 Results

It is possible to make use of linear control theory to develop a rate-based congestion control algorithm. Early work such as [3] and [10] made use of linear control theory to model the behavior of a one switch system. This thesis extends that work by adding previously unmodeled dynamics that arise when more complex networks are considered.

The previously unmodeled dynamics of queueing delay and rate blending indicate that queues in a multi-switch system influence the behavior of the sources and switches throughout the network. Queueing delay results when a source cell must pass through a switch with a non-zero queue in route to its destination. Delay destabilizes the system, and, therefore requires consideration. The rate blending is a result of the fact that the total rate at which cells depart a queue may not exceed the service rate of the queue, and cells from all of the sources which use that queue must share its resources. These two dynamics indicate that the cell rate entering a queue is often different from the rate exiting the queue.

Another dynamic also results when the system contains more than one active switch. A switch calculates a source cell rate that it wishes to feedback to all sources that have caused congestion in its queue. The switch replaces the feedback cell rate calculated by another switch which may already be in the RM cell if the rate that it has calculated is lower. Otherwise, it leaves the data in the RM cell unchanged. If a source has cells which cause congestion in more than one switch, then more than one switch calculates a feedback rate to suggest. The lowest feedback rate must be the rate that the source adopts. Thus a *min* function operates whenever the cells of a source encounter two or more non-zero queues in route to their destination.

With these dynamics modeled, an advanced model of the network which included link delay, queueing delay, rate blending and the *min* function was formulated. Since all of these modeling elements have been represented, an arbitrarily complex network which contains all of these elements may be represented in block diagram form. Then the state of the system at any time can be determined based on the information from the block diagram.

7.2 Future Directions

This thesis developed representations for several previously unmodeled dynamics. Unfortunately, those representations took the form of complex equations which were best suited to algorithmic implementation. For this reason, little insight could be directly drawn from the equation-level representations of the system. It is desirable to simplify the equations of this thesis so that they take on a form more amenable to direct analysis and thus to the formation of intuition and insight. This requires making suitable approximations to the forms of the equations or algorithms as they now stand.

It is also desirable to make greater use of control theory to analyze the networks of this thesis. That is, it may be possible to use control techniques to predict the behavior of the system to a greater degree. This again would provide more insight into the response of a system prior to formulating an equation-level simulation of the network under analysis.

References

[1] "ATM Forum Traffic Management Specification Version 4.0," ATM Forum/95-0013R10, February 1996.

[2] Barnhart, A. W., "Use of the Extended PRCA with Various Switch Mechanisms," ATM Forum/94-0898, September 1994.

[3] Berry, R., "A Linear Control Approach to ATM Congestion Control," thesis, MIT Department of Electrical Engineering and Computer Science, 1996.

[4] Charny, A., D. D. Clark, R. Jain, "Congestion Control with Explicit Rate Indication," Proc. ICC'95, June 1995.

[5] "Closed-Loop Rate-Based Traffic Management," ATM Forum/95-0013R3, editing version, January 1995.

[6] "Closed-Loop Rate-Based Traffic Management," ATM Forum/94-0438R1, July 1994.

[7] "Closed-Loop Rate-Based Traffic Management," ATM Forum/94-0438R2, September 1994.

[8] Jain, R. "Congestion Control and Traffic Management in ATM Networks: Recent Advances and A Survey," to be published in *Computer Networks and ISDN Systems.*

[9] Oppenheim, A. V., and R. W. Schafer, *Discrete-Time Signal Processing,* Prentice Hall, 1989.

[10] Rohrs, C. E., R. A. Berry, and S. J. O'Halek, "A Control Engineer's Look at ATM Congestion Avoidance," to be published in *Computer Communications Review*, 1996.

[11] Rohrs, C. E., J. L. Melsa, and D. G. Schultz, *Linear Control Systems*, McGraw-Hill, 1993.

[12] Siu, K. and H. Tzeng, "Intelligent Congestion Control for ABR Service in ATM Networks," *Computer Communications Review*, Vol. 24, No. 5 pp. 81-106, October 1995.

[13] Yin, N. and M. G. Hluchyj, "On Closed-Loop Rate Control for ATM Cell Relay Networks," *Proceedings, IEEE Infocom '94*, pp. 99-108, June 1994.

Appendix A

Network Figures

The models of the networks shown in this thesis consist of switches, sources and destinations. Networks may be composed of various numbers of these elements. Each element is described in detail below so that the connectivity of the elements and the role that each plays in the network may be understood.

A.1 The Switch

The standard model of a switch to be used throughout this thesis is shown next as Fig. A.1. Each switch is numbered for reference. A switch has several input ports on which it can receive cells and several output ports from which it can transmit cells that it has received. Connections are made to indicate the input and output ports that a given source will use. The switch below has five input ports and five output ports. In this figure, cells enter input port 2, and exit via output port 3. It is possible for several sources to share an output port. Not shown in the figure is the queue that exists as part of the output port. Cells must either be serviced immediately or enter a queue in the switch before they may leave the switch via the output port. Traffic may flow in either direction as indicated by the bi-directional links, therefore a port may act as either an input port or an output port.

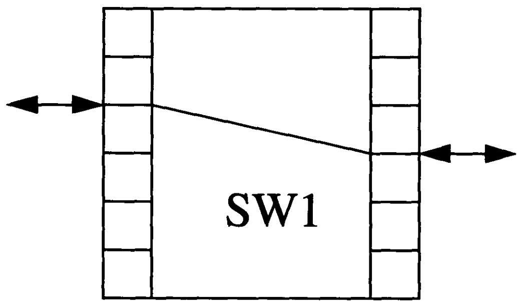

Figure A.1: Model of a switch

A.2 The Source

The next figure shows the form that a source will take in the network models in this thesis. Each source will be numbered for reference purposes. A source is capable of generating cells and transmitting them via its output port. It initializes to a given cell rate, and future cell rates are determined by feedback information that it receives via its input port. A two-way connection is shown on one side of the source below. This emphasizes the point that it transmits and receives information. The arrow on the right points toward an the input port of an adjacent switch.

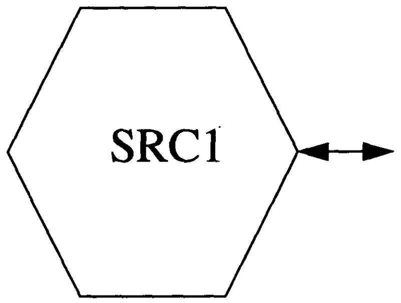

Figure A.1: Model of a source

A.3 The Destination

Destinations are the simplest elements to model for the purposes of this thesis. Each destination receives data cells from one or more sources. It selects RM cells from the input data stream and returns the RM cells via the same path to the sources from which they came. The following figure shows the model of the destination to be used throughout this thesis. It has a two-way connection to indicate that it receives and transmits information.

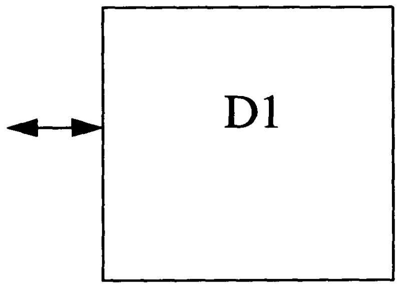

Figure A.1: The Destination Model

Appendix B

Analysis Tools

B.1 Matlab

The mathematical software package Matlab is used for analysis in this thesis. Matlab 4.2, manufactured by The Mathworks, Inc. is a matrix-based mathematical tool which includes many predefined functions. Among these are several functions for use in control system applications. Matlab may also be used like a programming language. Functions or programs can be written to suit needs of the user. In these chapters, Matlab is used for applications such as pole-zero analysis, for generating Bode plots, and for performing simulations based on approximate block diagram models of example networks. Matlab simulation results are referred to as equation-level simulation results in this thesis.

B.2 OPNET

OPNET 2.5.A is an event based network simulation tool manufactured by MIL3, Inc. In OPNET, a network is constructed based on a hierarchical structure. On the lowest level, processors written in C manipulate the network cells and test for conditions based on the code entered by the user. The algorithms by which the sources and switches of a network operate are coded on the processor level. OPNET also includes several standard processors. On the node level, the connectivity of the processors is determined. Each node is comprised of one or more processors and may include other predefined OPNET modules such as transmitters and receivers. OPNET also provides some standard nodes. On the highest level, the network level, the connections between nodes are determined. OPNET provides a structure and many tools while allowing freedom to implement the algorithms of choice. OPNET simulations closely model actual network conditions. Therefore, the outputs of OPNET simulations shall serve as the discrete-event simulation results.