A Simulation Study of Token Generation Policies for a Leaky Bucket Traffic Shaper

Narayanan Prithviraj
Carey L. Williamson
Telecommunications Research Laboratories (TR*Labs*)
108 – 15 Innovation Boulevard
Saskatoon, SK, CANADA S7N 2X8
Phone: (306) 668-8204
FAX: (306) 668-1944

February 11, 1994

Abstract

This paper presents a simulation study of different token generation policies for a leaky bucket traffic shaper. In particular, the paper addresses the following question: Can the performance of a leaky bucket traffic shaper be improved by matching the token generation policy to the characteristics of the input traffic? Three different token generation policies are considered: Deterministic, Bernoulli, and a Markov Modulated Bernoulli Process (MMBP). Each policy is simulated on three different input traffic models (Deterministic, Bernoulli, and MMBP), using a full factorial experimental design.

The simulation results show that, overall, the deterministic token generation policy provides the most consistent performance (lower cell loss and lower cell delay) across a wide range of input traffic types. Therefore the deterministic policy is the best token generation policy to use, even if the characteristics of the input traffic are known to be other than deterministic.

Technical Subject Area: Modeling and Simulation Techniques

1 Introduction

The *leaky bucket* is a recently proposed cell-level access control mechanism for high speed networks [1, 5, 8]. Combined with call-level admission controls, cell-level controls such as the leaky bucket form a key part of the congestion control framework [2, 3] in high speed integrated services networks, such as B-ISDN/ATM [7].

The leaky bucket traffic shaper serves two main purposes. First, it reduces burstiness in the input traffic, by limiting the maximum number of back-to-back cells that a traffic source can inject into the network. Second, it controls the average rate of cell transmission for each traffic source, ensuring that the long-term average traffic flow for each source is less than or equal to the rate reserved by the source at call admission.

The leaky bucket shapes network traffic by buffering input cells, as necessary, at the edges of the network, and then releasing the cells into the network at a controlled rate, determined by the *token generation rate*. The *input buffer size* of the shaper determines the maximum burst of input cells that can be accommodated at the shaper. Arriving cells that exceed this capacity are simply dropped. The *token pool size* at the shaper determines the maximum number of back-to-back cells that can enter the network.

Most of the existing research on the leaky bucket mechanism has concentrated on sizing the input buffer and token pool to reduce the burstiness of traffic entering the network [1, 5], and to balance the tradeoff between delay and loss at the shaper [6]. However, the token generation rate and the token generation policy are also important parameters.

The conventional leaky bucket mechanism uses a deterministic policy for token generation: if the token generation rate is R, then a single new token is generated every $1/R$ time units. The deterministic token generation policy is simple to implement, but is somewhat conservative in its use of the network [6]. That is, when the token pool is empty, a burst of arriving input cells is always released into the network at the token generation rate, with minimal burstiness, even if the network is otherwise idle.

Other token generation policies could be used to enforce the same mean token generation rate, while allowing a slightly greater degree of burstiness into the network. That is, rather than removing *all* randomness and burstiness from the input traffic, such a shaper would remove only *some* of the randomness and burstiness (or possibly add some). At first glance, such an approach seems foolish, since it replaces a deterministic process by a random one. However, such a scheme could potentially match the burst characteristics of the token generation policy to the burst characteristics of the input traffic to provide a "gentler" traffic shaper, reducing cell loss and cell delay at the shaper[1], and better exploiting the statistical multiplexing nature of ATM.

[1]The effects of such an approach on the *network level* cell loss and cell delay would need to be considered, of course.

The purpose of this paper, then, is to answer the following question:

> "Can the performance of a leaky bucket traffic shaper be improved by matching the token generation policy to the input traffic characteristics?"

More precisely, the objectives are:

1. To compare different token generation policies and to determine the most effective amongst them for different input traffic.

2. To use formal experimental design strategies and statistical techniques in evaluating performance.

The remainder of this paper is organized as follows. Section 2 presents the leaky bucket traffic shaper model. Section 3 describes the experimental design, including the experimental factors, their levels, and the performance metrics used. Section 4 presents the simulation results, and Section 5 concludes the paper.

2 The Leaky Bucket Traffic Shaper

2.1 Leaky Bucket Model

The leaky bucket traffic shaper regulates cell-level access to a network. The leaky bucket operates on a per-flow or per-VC basis, and operates in the discrete time domain (i.e., slotted time). All cells are the same size, and each cell requires one time unit for transmission on the network. There is at most one input cell arriving during each time slot, and there is at most one cell departing from the leaky bucket during each time slot.

Figure 1 depicts the operation of a buffered leaky bucket traffic shaper. Arriving cells enter an input queue Q. If the queue is full, the arriving cell is dropped. For a cell at the head of the queue to enter the network, the cell must first obtain a token from a token pool. Tokens are generated at a fixed deterministic rate and enter this pool. If the token pool is full, newly generated tokens are discarded.

There are several parameters that govern the operation of the leaky bucket. The size of the token pool M limits the maximum number of back-to-back cells that can enter the network. The token rate R determines the mean rate at which cells enter the network. Typically, this rate is set to the rate allocated to the connection at the time of call admission. The size of the input queue Q presents a tradeoff between cell delay and cell loss. Increasing the queue size decreases the probability of cells being dropped at the shaper, but increases the waiting time that cells can experience at the shaper. In our model, the input buffer is fixed at $Q = 10$, the token pool size is $M = 5$, and the token generation policy is allowed to vary.

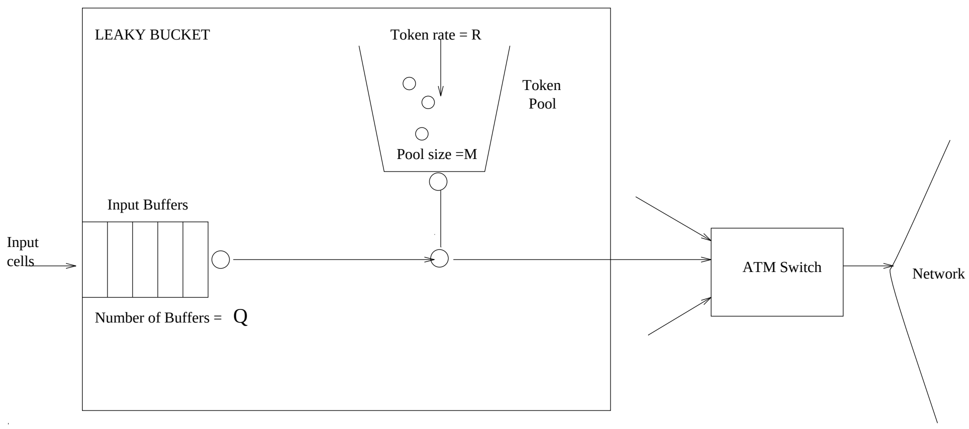

Figure 1: A Buffered Leaky Bucket Traffic Shaper

2.2 Terminology

This paper uses the following terminology:

- The *cell arrival policy* refers to the arrival process of input cells at the leaky bucket. In this paper, the following cell arrival policies are considered:
  - *Deterministic* : Deterministic cell arrivals represent the simplest cell arrival policy. If the mean cell arrival rate is λ cells/slot, then a new cell arrives every $\frac{1}{\lambda}$ slots. The cell interarrival times are constant.
  - *Bernoulli*: The Bernoulli cell arrival policy models *memoryless* arrivals. If the mean cell arrival rate is λ, then the probability of a cell arrival in any given time slot is λ. Each time slot is independent, and the cell interarrival times are geometrically distributed with mean $\frac{1}{\lambda}$.
  - *Markov Modulated Bernoulli Process (MMBP)*: The MMBP arrival policy is a commonly used traffic model for ATM that can model a range of traffic characteristics, including burstiness and correlation. In this paper, we use a two-state MMBP model. Cells arrive in each of these states according to a Bernoulli process with means of λ_1 and λ_2 respectively. The mean holding times in state 1 and state 2 are geometrically distributed with means of $\frac{1}{\beta}$ and $\frac{1}{\alpha}$ respectively. The mean cell rate is given by $\lambda = \frac{\alpha\lambda_1+\beta\lambda_2}{\alpha+\beta}$. The peak cell rate λ_p is given by $\lambda_p = \lambda_1$ (assuming $\lambda_1 > \lambda_2$). The burstiness level b is given by $b = \frac{\lambda_p}{\lambda}$.

  The Deterministic arrival model represents predictable traffic with no burstiness. Bernoulli arrivals can model randomness in traffic but with minimal burstiness, while the MMBP cell arrival policy can model both randomness and burstiness in the input traffic.

- The *mean cell rate* represents the average rate at which input cells arrive at the leaky bucket. The mean cell rate is denoted by λ.

- The *token generation policy* refers to the arrival process of tokens to the token pool. The token generation policies considered in this paper are Deterministic, Bernoulli, and MMBP.

- The *mean token rate* is the average rate at which tokens enter the token pool. This rate is fixed for the duration of the call. The mean token rate is denoted by R.

- The *load factor* γ is a parameter used to represent the relative load at which input cells arrive at the leaky bucket. The load factor is expressed as a percentage and is given by $\gamma = \frac{mean\ cell\ rate}{mean\ token\ rate} * 100\%$.

  The purpose of this parameter is to allow consideration of a traffic source that transmits at a rate λ that is less than, equal to, or greater than the token rate R. Since this parameter represents *relative* load, the simulation results are all independent of the link speeds in the network or the absolute rate R reserved by the source.

3 Experimental Methodology

3.1 Experimental Design

The simulation experiment for the leaky bucket traffic shaper is conducted using formal experimental design methodology. As discussed in Section 2, there are several parameters that determine the operation of the leaky bucket. The parameters can be divided into two categories. The *system* parameters are those that are an integral part of the leaky bucket scheme, while the *workload* parameters are those that characterize the arrival process of input cells that enter the leaky bucket. Table 1 lists these parameters.

The *factors* of the experiment are the parameters that are varied during the experiment. Table 1 shows that there are a total of six factors that could be varied in a full factorial experiment. To keep the experiment simple, however, only three factors are selected to be

Table 1: Parameters Affecting the Operation of the Leaky Bucket

SYSTEM PARAMETERS	WORKLOAD PARAMETERS
Input buffer pool size	Mean cell rate
Token pool size	
Mean token rate	Cell arrival policy
Token generation policy	

varied: the mean cell rate, the cell arrival policy, and the token generation policy. The other factors are assigned fixed values[2], as shown in Table 2.

Table 2: Fixed Factors

PARAMETER	VALUE
Input buffer pool size	10
Token pool size	5
Mean token rate	0.10 cells/slot

The *levels* within the experiment are the range of values that the variable factors can take. Table 3 lists the levels chosen for each of the variable factors in our experiment.

The three token generation policies chosen are the Deterministic, Bernoulli, and MMBP distributions. These policies represent the central factor in this experiment. The mean cell rate is varied from 0.09 cells/slot to 0.11 cells/slot in steps of 0.002 cells/slot so that the load factor γ ranges from 90% to 110%. This range of values represents a traffic source operating at an average rate that is within 10% (either above or below) of its reserved token rate. Finally, three different input cell arrival policies (Deterministic, Bernoulli, and MMBP) are used to evaluate the effectiveness of the token generation policies. A full factorial experiment for the variable factors and levels was performed yielding a total of 99 sessions.

Table 3: Variable Factors

FACTOR	LEVELS	NO. OF LEVELS
Token generation policy	Deterministic, Bernoulli, MMBP	3
Mean cell rate	0.09 . . . 0.11 cells/slot	11
Cell arrival policy	Deterministic, Bernoulli, MMBP	3

3.2 Performance Metrics

To evaluate the effectiveness of the various token generation policies, the following performance metrics were selected:

- The *cell loss probability* measures the fraction of cells dropped at the leaky bucket due to input buffer overflow. A low cell loss probability is desirable.

[2] A separate experiment (not described here) was used to establish the input buffer size and the token pool size. Input buffer sizes that are too small (say, less than 5) create too much cell loss, and input buffers that are too large (say, greater than 20) create too much cell delay. Similarly, token pool sizes that are too small leave no room for the arrival of a burst of tokens, and token pool sizes that are too large make the token arrival policy moot. Thus the values chosen represent reasonable tradeoffs. In general, our conclusions are not dependent on the specific parameter settings shown here.

- The *mean cell delay* measures the time from the cell's arrival at the leaky bucket to the time at which the cell enters the network. Note that the cell delay metric applies only to those cells that successfully enter the network. Like the cell loss probability, a low mean cell delay is desirable.

Cell delay jitter and cell inter-departure time were also included as performance metrics in the simulation experiment. For brevity, however, these results are omitted from the paper.

3.3 Simulation Methodology

The run length of the simulation is chosen using formal methods so that the results can be interpreted with a strong degree of confidence. The method of *batch means* [4] was used so as to be 95% confident that the error in the mean cell delay results was less than 1%. Batch sizes of 10000 time slots were used. The number of batches required varied depending on the cell arrival policy being simulated (typically 30-50). For all simulation runs, the first five batches were ignored to reduce the effect of initial bias on results.

4 Simulation Results

This section presents the results of the simulation for each cell arrival policy and token generation policy combination. Two graphs are presented for each input traffic model: one for cell loss performance, and one for cell delay performance. In all graphs, the mean cell rate is varied from 0.09 cells/slot to 0.11 cells/slot, so as to vary the load factor γ from 90% to 110%. The mean token rate is always fixed at 0.10 cells/slot.

Each graph has three lines corresponding to the Deterministic, Bernoulli, and MMBP token generation policies. All the graphs show some common characteristics. These are:

- As the load factor increases, the cell loss probability increases rapidly.
- Queueing delay also increases with an increase in the load factor.

These results are straightforward, and will not be discussed further. The remaining discussion focuses on the details of the graphs for each cell arrival policy.

4.1 Deterministic Input Traffic

Figures 2 and 3 present the simulation results for each token generation policy when the input traffic is Deterministic. Figure 2 presents the cell loss results, and Figure 3 presents the results for cell delay.

There are two regions of interest in each graph. In the underload region (load factor γ less than 100%), the traffic source is generating input cells at a rate λ that is less than the

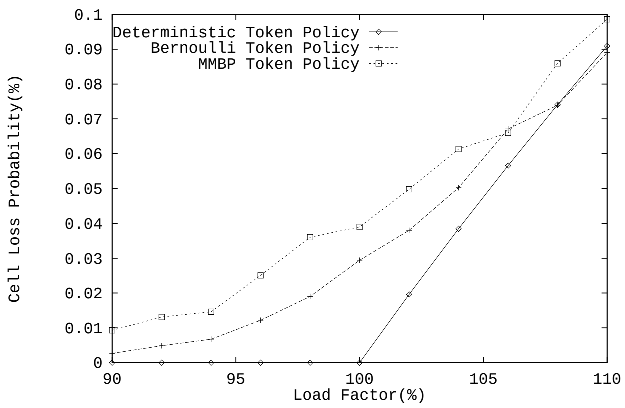

Figure 2: Cell Loss Probability for Deterministic Input Traffic

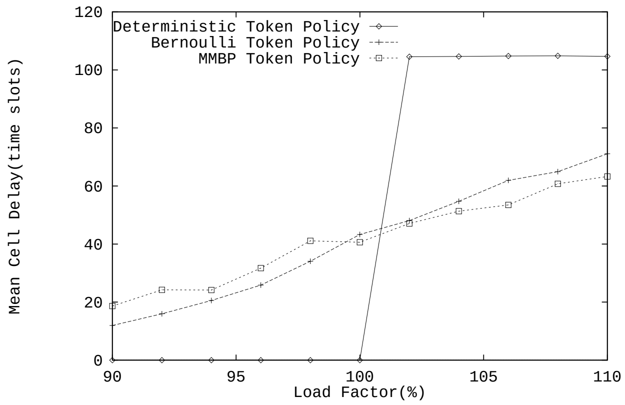

Figure 3: Mean Cell Delay for Deterministic Input Traffic

token rate R. In the overload region ($\gamma > 100\%$), the traffic source is exceeding its reserved rate.

The Deterministic token generation policy provides the best cell loss performance in both regions (see Figure 2). In the underload region, there are always tokens waiting in the token pool, and the cell loss is zero. In the overload region, the token pool is nearly always empty, and the excess traffic $\frac{\lambda - R}{\lambda}$ is lost at the shaper. Thus there is a linear increase in the cell loss probability, as shown in the figure. In both the underload and overload regions, the Deterministic token generation policy provides a lower bound for the average cell loss probability.

The cell loss probability for the Bernoulli and MMBP token generation policies are consistently higher than for the deterministic token generation policy. Because of the randomness in these token generation policies, input cells may have to wait arbitrarily long for the next token, causing the input buffer to fill and more cells to be lost (even in the underload region). The MMBP policy produces even higher cell loss than the Bernoulli policy.

Figure 3 presents the mean cell delay results for Deterministic input traffic. In the underload region, the Deterministic token policy shows zero cell delay, since the input buffer is always empty and the token pool is always full. In the overload region, however, the Deterministic token generation policy has very high delay, since the token pool is always empty, and the input buffer is always full. The delay characteristics change dramatically at the point $\gamma = 100\%$.

The change in delay for the other two token generation policies is much less dramatic. Delay increases steadily with load. In the underload region, cell delay is higher than with the Deterministic token policy, due to the randomness in token generation. In the overload region, cell delay is lower, for two reasons. First, since more cells are lost in each of these two policies, there are fewer cells waiting in the buffer. Second, bursts of token arrivals could remove more than one cell from the input buffer, rather than just one, making the mean queue size shorter. The MMBP token generation policy has slightly higher cell delay than the Bernoulli policy in the underload region, and slightly lower cell delay in the overload region (due to its higher burstiness).

4.2 Bernoulli Input Traffic

Figure 4 presents the cell loss results for the Bernoulli cell arrival policy. Because of the randomness in the input traffic now, all three token generation policies, including Deterministic, experience cell loss, even in the underload region. All three policies show a similar increase in cell loss as the load increases.

As before, the Deterministic token generation policy provides the best cell loss performance across the entire load region shown. It provides a much lower cell loss probability than the other two policies. The Bernoulli token policy also appears better than the MMBP token policy over the entire load range.

Figure 5 shows the mean cell delay results for the Bernoulli input traffic. At load factors less than 100%, the Deterministic token generation policy has lower cell delay than the other policies. However, in the overload region, this policy has higher cell delay. Both the Bernoulli and MMBP token policies exhibit similar cell delay.

4.3 MMBP Input Traffic

Figure 6 presents the cell loss results for the MMBP cell arrival policy, which has both randomness and burstiness in the input traffic. As in Section 4.2, all token generation policies experience cell loss throughout the load region shown, and all token generation policies show a similar increase in cell loss with increasing load. In this case, however, the Deterministic and the MMBP token generation policies provide almost identical performance, and the Bernoulli token policy is significantly worse.

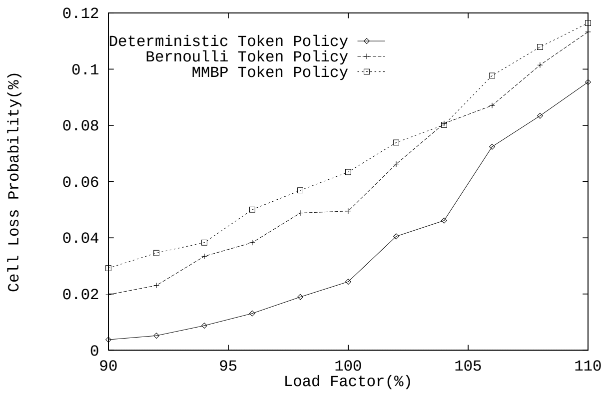

Figure 4: Cell Loss Probability for Bernoulli Input Traffic

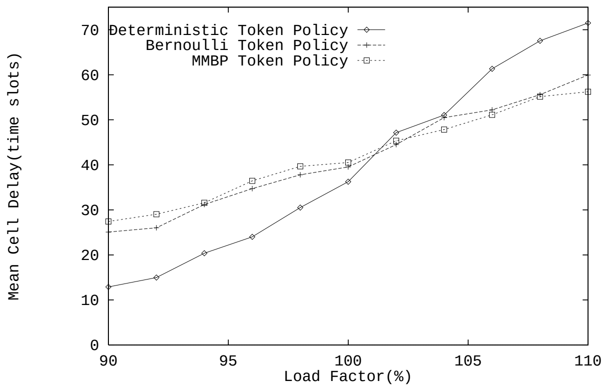

Figure 5: Mean Cell Delay for Bernoulli Input Traffic

Figure 7 presents the cell delay results for the MMBP input traffic. Here, there is very little difference between the three policies. As in Figure 6, the Deterministic and MMBP token policies provide very similar performance. The Bernoulli policy is slightly worse in the underload region, and slightly better in the overload region.

4.4 Summary

The observations presented in the preceding subsections suggest that there is no merit to matching the token generation policy to the input traffic characteristics. In almost all cases, the Deterministic token generation policy is either significantly better than, or no worse than, other token generation policies. In the underload region (hopefully the most likely region of operation in a real network), the Deterministic policy provides consistently lower cell loss and cell delay for all input traffic types.

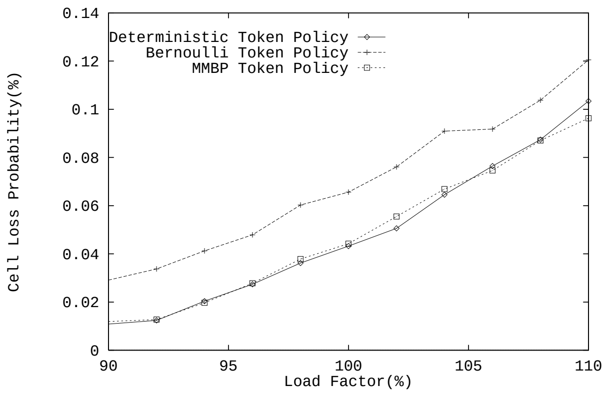

Figure 6: Cell Loss Probability for MMBP Input Traffic

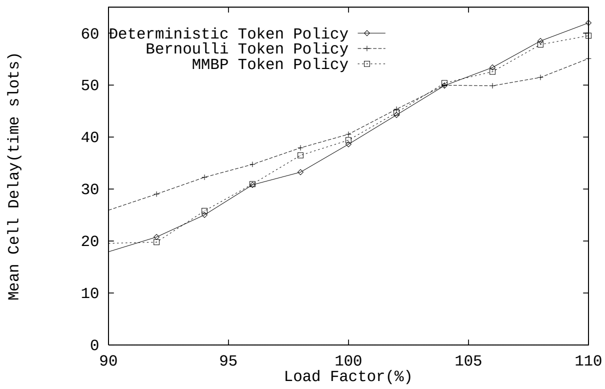

Figure 7: Mean Cell Delay for MMBP Input Traffic

Token generation policies other than deterministic are a bad idea for several reasons. First, they produce higher cell loss and higher cell delay at the shaper. Second, they are more likely to cause higher cell loss and higher cell delay within the network itself[3]. Finally, they are more complicated to implement.

[3]A separate analysis (not shown here) shows that the mean cell inter-departure time increases slightly (since fewer cells are forwarded), and the variance in the interdeparture time increases significantly, which means more queueing within the network.

5 Conclusions

This paper has presented a simulation evaluation of different token generation policies for a leaky bucket traffic shaper using formal experimental methodology. The effectiveness of a policy is measured in terms of the mean cell loss and mean cell delay at the shaper for three different types of input traffic.

The simulation results show that the Deterministic token generation policy is the best one to use. Providing that traffic sources generate cells at an average rate that is less than or equal to their reserved rate, the Deterministic policy provides consistently lower cell loss and cell delay for all input traffic types. Therefore the deterministic policy is the best token generation policy to use, particularly if the characteristics of the input traffic are unknown. The deterministic policy also works well even if the characteristics of the input traffic are known to be other than deterministic.

References

[1] K. Bala, I. Cidon, and K. Sohraby, "Congestion Control for High Speed Packet Switched Networks", *Proceedings of INFOCOMM 1990*, San Francisco, pp. 520-525, June 1990.

[2] H. Gilbert, O. Aboul-Magd, and V. Phung, "Developing a Cohesive Traffic Management Strategy for ATM Networks", *IEEE Communications Magazine*, Vol. 30, No. 10, pp. 36-45, October 1991.

[3] I. Habib and T. Saadawi, "Controlling Flow and Avoiding Congestion in Broadband Networks", *IEEE Communications Magazine*, Vol. 30, No. 10, pp. 46-53, October 1991.

[4] R. Jain, *The Art of Computer System Performance Analysis*, Wiley & Sons, 1991.

[5] W. Liu, I. Cidon, I. Gopal, and M. Sidi, "Congestion Control Through Input Rate Regulation", *Proceedings of IEEE GLOBECOM '89*, pp. 1764-1768, November 1989.

[6] H. Liu, "Traffic Shaping for Congestion Control in High Speed ATM Networks", Research Report 92-6 (M.Sc. Thesis), Department of Computational Science, University of Saskatchewan, August 1992.

[7] S. Minzer, "Broadband ISDN and Asynchronous Transfer Mode (ATM)", *IEEE Communications Magazine*, Vol. 27, No. 9, pp. 17-24, September, 1989.

[8] J. Turner, "New Directions in Communications (or Which Way to the Information Age?)", *IEEE Communications Magazine*, Vol. 24, No. 10, pp. 8-15, October 1986.