Dynamic Traffic Control Using Feedback and Traffic Prediction in ATM Networks

Yu Gong
Dept. Computer Sc. & Eng.
Auburn University
Auburn, AL 36849
gong@eng.auburn.edu
Tel.: (205) 844-6316
Fax: (205) 844-6329

Ian F. Akyildiz
School of Electrical & Comp. Eng.
Georgia Institute of Technology
Atlanta, GA 30332
ian@armani.gatech.edu
Tel.: (404) 894-5141
Fax: (404) 853-9410

Abstract

It has been found recently that feedback in an ATM network is useful ***in the long run*** to alleviate congestion. This general conclusion, although theoretically important, does not address the practical concern as to how the network behaves when it is adjusting itself in short to medium range. The work reported in this paper is motivated by this practical concern. A new feedback based dynamic traffic control mechanism (called *balanced* mechanism) is proposed and compared with the pure PCC and an existing feedback based mechanism, known as Explicit Forward Congestion Notification (EFCN). It is shown that the balanced mechanism outperforms both PCC and the EFCN.

Key Words: ATM, Congestion Control, Closed-loop Control, EFCN, Feedback.

1 Introduction

To cope with the dilemma of large delay-bandwidth product in high-speed networks, the CCITT standard recommendation for asynchronous transfer mode (ATM) [1] selects a preventive congestion control (PCC) principle. This principle would be successful if two of its primary functions, connection admission control (CAC) and usage parameter control (UPC), were both effective. In reality, the contrary is often true. Previous study on CAC [2] and UPC [3, 4, 5, 6] show that they leave loopholes to possible uncontrolled congestion, reflected by failure to guarantee quality of service (QoS). As such, CACs are predominantly designed using conservative methods, leaving a large portion of network power untapped.

A feedback based mechanism known as explicit forward congestion notification (EFCN) is studied in [7] for assisting users to temporarily oversubscribe to network. It has been shown that using the EFCN, users can be facilitated with surplus bandwidth that would otherwise be wasted, especially during the off-peak hours. Oversubscription to network was first proposed in [8] by marking cells. Recently, this idea was extended to include source planning using "credit banking" [9] or feedback [7].

In [10] and [11], it is proven that feedback can be used to adjust source rates toward asymptotic (desirable) levels. As expected, the convergency requires a long transient delay. It is also verified that feedback does not help the protocol to converge to a stable operating point; rather, it brings the QoS into a neighborhood of the target operating point. Therefore, feedback must be continuously used throughout the lifetime of a connection, in order to keep the fluctuation small.

Although using feedback emerges as a viable approach to ATM traffic control, a number of practical issues have still not been solved, such as "how to deal with the large feedback delay?" "how do existing traffic control mechanisms compare in their transient performance?" "how to adjust the source rate?" In this paper, we focus on these important issues. We propose a new feedback based traffic control mechanism (called *balanced* mechanism) and compare it with pure PCC and the EFCN. In the comparisons, transient behaviors during the periods of load changes are observed. We show that the balanced mechanism outperforms both pure PCC and the EFCN mechanisms.

2 Feedback Delay Concept

Definition 1. The Logical Generation Time (LGT) of Feedback is the earliest time after which cells from a source cannot possibly affect the value of the feedback.

Suppose feedback A is received by a traffic source, which reflects the network condition caused by cells sent before LGT(A). In other words, feedback A does not convey the network condition since LGT(A).

Definition 2. The Physical Generation Time (PGT) of Feedback is the time when the feedback physically appears in the network, and begins its journey to the source.

By introducing Definition 2, we want to clarify the fact that even before the feedback is physically generated, the (logical) congestion state may have already been formed. Based on these two definitions, three important time points are identified.

1. At t_s, a tagged cell that will later carry an explicit congestion feedback is sent into the network.

2. At t_d, the tagged cell arrives at the destination and feedback is sent to the source.

3. At t_f, the feedback arrives at the source.

A straightforward observation is that at time t_s, when the tagged cell is being transmitted, all cells that *cause* the feedback have already been sent into the network and network condition has already been affected. Therefore, the feedback should be considered already *logically* formed at time t_s even though no physical feedback can be generated before t_d ($t_d > t_s$). Therefore, feedback delay, referred to as the *age* of feedback, is defined as follows.

Definition 3. The Age of Feedback is the difference between the current time and the LGT of the feedback.

Clearly, the age of feedback in EFCN is a constant, equal to a round-trip delay, i.e., $t_f - t_s$. The implication is that the feedback does not reflect the network status during this one round-trip interval $(t_s, t_f]$. Therefore, additional factors contributing to possible changes to the network status in this interval must be taken into consideration in rate adjustment. The three most important factors are: interfering cells from other connections, the source rate, and transmission rate (i.e., the rate for the cells to pass the UPC and enter the network). Unfortunately, the knowledge about interfering cells from other connections is theoretically impossible to obtain, although information regarding the other two factors can be collected locally. In practice, accurate heuristic estimates about the interfering traffic may be used.

3 A Feedback-Based Traffic Control Mechanism

In this section, we explain the two important aspects of a feedback-based mechanism, namely, the feedback and the rate adjustment function.

3.1 The Feedback

In contrast to the EFCN, which uses binary feedback, our traffic controller uses *continuous feedback*. A special dedicated cell, called *feedback cell (FC)* is issued periodically by the controller to probe the maximum switch utilization along a virtual connection. This FC corresponds to the tagged cell in Section 2. The LGT for the feedback in an FC is the time when the FC is being transmitted. This conclusion is useful to the rate adjustment function because the age of feedback can be tracked quite easily. In fact, the age of feedback is equal to one round-trip time plus queueing delay experienced by an FC in intermediate switches. In all intermediate switches, FCs are given the highest priority to transmit. Therefore, the total queueing delay is extremely small when compared with the round-trip time, making the age of feedback close to a constant.

We assume that only one FC is used and the possibility of FC loss is ignored. The controller works in a cyclic manner as follows.

```
1. Issue an FC.
2. Wait for the feedback.
3. Receive feedback and adjust rate.
4. Go to 1.
```

3.2 The Rate Adjustment

The source controller consists of two Leaky Bucket UPCs arranged in tandem, the first is for policing the average rate (with token pool size K_a and token generation rate r_a) and the second for the peak rate (with token pool size K_p and token generation rate r_p). A cell must pass both policers to enter the network.

Four parameters (i.e., r_a, r_p, K_a, and K_p) can thus be used to control the traffic submitted to the network. The effect of all these parameters is not the same. It is shown in [4] that changing the token pool size has lesser impact on the cell loss probability than modifying the token rate. Therefore, in our work we assume that both pool sizes are constant. Also, as concluded in earlier research [3, 4], the peak rate parameter can be enforced quite accurately using the LB policer. Therefore, the peak rate parameter r_p is not allowed to vary. According to [3, 4], the parameter r_p should be set to the target peak rate. As a result, only one parameter (the token rate r_a in the average rate policer) remains, which can be modified based on the feedback. We assume that the range for r_a is $[r_m, r_M]$, i.e., the maximum and the minimum token rates are r_M and r_m, respectively. We describe a method for tuning r_a within this interval in the remainder of this subsection.

Consider the time axis as shown in Figure 1. Time τ_i ($i = 1, 2, \ldots$) in Figure 1 is the moment when the ith FC enters the network. The interval $[\tau_i, \tau_{i+1}]$ is referred to as the ith *cycle*. Note that the duration of a cycle is exactly equal to the age of the feedback available at the end of the cycle. As explained earlier, the cycle length is almost a constant.

The following notations are defined for the ith cycle.

r_i: the token rate in the average rate policer during the ith cycle. Note that, for notational simplicity, here we drop the subscript "a" which we used earlier.

η_i: the feedback available at the end of the ith cycle.

s_i: the average source rate during the ith cycle.

t_i : the average transmission rate during the ith cycle.

In the beginning of the ith cycle (time τ_i in Figure 1), five parameters circled in Figure 1 regarding what happened in the previous two cycles are used for estimating the token rate r_i for cycle i.

The estimate for r_i is denoted by $\hat{r}_i$. The first restriction on $\hat{r}_i$ is

$$r_m \leq \hat{r}_i \leq r_M \qquad (1)$$

The estimation procedure proceeds by first predicting, for cycle i, the cell rates $\hat{s}_i$ from the source and $\hat{t}_i$ into the network. The smaller of the two is used as $\hat{r}_i$, i.e.,

$$\hat{r}_i = \min\{\hat{s}_i, \hat{t}_i\} \qquad (2)$$

Note that the actual rates s_i and t_i always satisfy $s_i \geq t_i$, but this relation does not necessarily hold for their

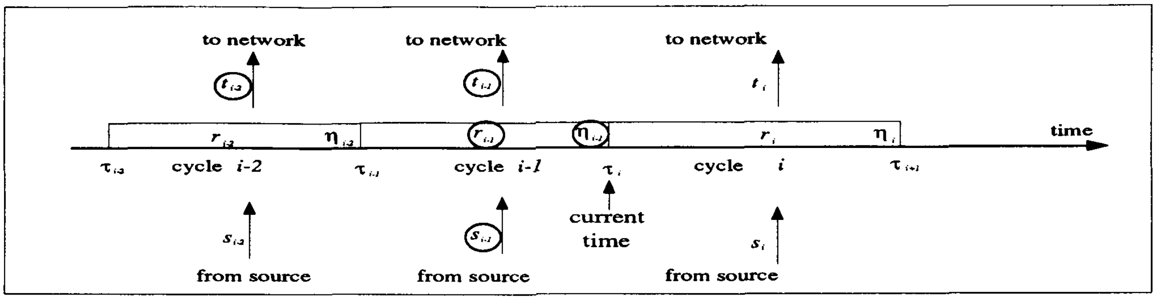

Figure 1: Operating Cycles and Important Parameters

estimates $\hat{s}_i$ and $\hat{t}_i$ since $\hat{s}_i$ and $\hat{t}_i$ are predicted with conflicting QoS goals: $\hat{s}_i$ is estimated for the purpose of minimizing the source cell loss probability, while $\hat{t}_i$ is computed to satisfy QoS requirement within the network.

To achieve the smallest cell loss probability, $\hat{s}_i$ must be as close to the actual average rate as possible. Since the duration of all cycles are necessarily long, the following estimation seems to be reasonable.

$$\hat{s}_i = s_{i-1} \tag{3}$$

where s_{i-1} is the measured source rate during the previous cycle.

The target network QoS is difficult to determine because it is related to all connections in a complicated manner. Therefore, we propose an addition to the ATM admission control mechanism as follows. When a user requests a virtual connection, a ***target VC utilization*** (denoted by η^*) is also negotiated between the user and the network. During the communication phase, the network must keep the utilization along the virtual connection lower than this target value. Congestion is, therefore, defined as the violation of this restriction. There are a number of advantages of this QoS control. First , the traffic controller does not have to observe the detailed QoS measures (e.g., cell loss, delay, delay jitter, etc.) within the network, which can be difficult to obtain. Second, users can be better protected from affecting each other. Third, the constant availability of switch utilizations at the network boundary can also assist routing and connection setup.

We now describe how to estimate $\hat{t}_i$ by distinguishing several cases based on the feedback value η_{i-1} at time τ_i.

Case 1 ($\eta_{i-1} > \eta^*$): In this case, the network is considered being congested in cycle $i-1$. Therefore, in the next cycle (cycle i), the token rate (r_i) must be reduced. The extent of this rate deduction, however, is dependent upon whether the source has already submitted less traffic during the age of the feedback.

Case 1.a ($t_{i-1} < t_{i-2}$): In this case, the source has already submitted lighter traffic to the network during the age of the feedback. Therefore, $\hat{t}_i$ can be estimated using an optimistic function.

$$\hat{t}_i = \min\{t_{i-1}, \mathcal{F}_o[\eta_{i-1}, \phi_1(\eta_{i-1})]\} \tag{4}$$

where

$$\mathcal{F}_o(\eta, y) = (r_m - r_M)\frac{\log(y + 1 - y\eta)}{\log(y + 1)} + r_m \tag{5}$$

and

$$\phi_1(\eta) = (C_M - C_m)\frac{1 - \eta}{1 - \eta^*} + C_m \tag{6}$$

In equation (6), C_m and C_M (where $C_m \leq C_M$) are two parameters explained shortly. The key to this estimation is the heuristic function (5) which is plotted in Figure 2 with various values of y. This is simply a monotonically decreasing logarithmic function of the network utilization η. In addition, parameter y controls the decreasing speed as η increases — the higher the y, the slower the decreasing rate. Because of the nature of the logarithmic function, the estimated

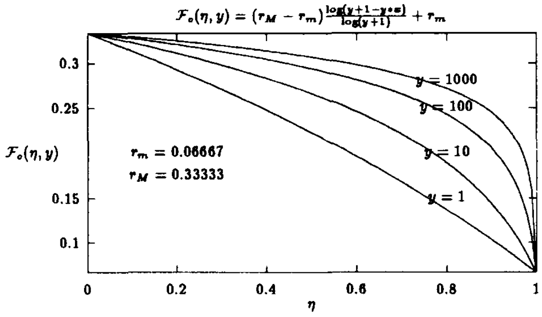

Figure 2: The Optimistic Rate Adjustment Function

transmission rate does not decrease drastically when η increases. So, we call this function an *optimistic function*. Furthermore, y is called the *degree of confidence* (for the optimism).

Equation (6) requires the degree of confidence to linearly decrease with the utilization η in an interval $[C_m, C_M]$.

Case 1.b ($t_{i-1} \geq t_{i-2}$): In this case, more traffic was submitted into the network during the age of the feedback. Most likely, the congestion condition has not been changed at the current time τ_i. Therefore, $\hat{t}_i$ is estimated using a conservative heuristic function.

$$\hat{t}_i = \min\{t_{i-1}, \mathcal{F}_c[\eta_{i-1}, \theta_1(\eta_{i-1})]\} \qquad (7)$$

where

$$\begin{aligned}\mathcal{F}_c(\eta, y) \quad = \quad & \frac{1}{1-e^{-y}}[(r_M - r_m)e^{-y\eta} + \\ & r_m - r_M e^{-y}] \qquad (8)\end{aligned}$$

and

$$\theta_1(\eta) = (C_M - C_m)\frac{\eta - \eta^*}{1 - \eta^*} + C_m \qquad (9)$$

The function in (8) is a negative exponential function with the property that a small increase of η can incur a large rate deduction. Thus, we call this function *conservative function*. Similarly, equation (9) requires the *degree of confidence* for the conservativeness to linearly increase with the utilization.

Case 2 ($\eta_{i-1} \leq \eta^*$): This feedback indicates no congestion in the network. But this is only true for the network condition before the LGT of the feedback. As such, the traffic source still needs to proceed prudently.

Case 2.a ($t_{i-1} < t_{i-2}$): In this case, the source has submitted lighter traffic to the network during the age of the feedback; therefore, $\hat{t}_i$ is computed using an optimistic function

$$\hat{t}_i = \mathcal{F}_o[\eta_{i-1}, \phi_2(\eta_{i-1})] \qquad (10)$$

with a linearly decreasing function for the degree of confidence.

$$\phi_2(\eta) = (C_M - C_m)\frac{\eta^* - \eta}{\eta^*} + C_m \qquad (11)$$

Different from equation (4) in Case 1.a, the estimated rate in (10) is not subject to the upper bound t_{i-1}. Thus, the token rate $\hat{t}_i$ may increase for the next cycle, which is not allowed in Case 1.a.

Case 2.b ($t_{i-1} \geq t_{i-2}$): In this case, the transmission rate has been increased during the age of the feedback. However, the effect of this rate increase is not known yet at the current time τ_i. Therefore, a conservative estimation is used for this case.

$$\hat{t}_i = \min\{t_{i-1}, \mathcal{F}_c[\eta_{i-1}, \theta_2(\eta_{i-1})]\} \qquad (12)$$

where

$$\theta_2(\eta) = (C_M - C_m)\frac{\eta}{\eta^*} + C_m \qquad (13)$$

3.3 Related Work in the Literature

In concluding this section, we discuss work related to ours presented in this paper. The closest work to ours is reported in [12] and [13]. In [12], a feedback scheme is proposed which consists of sending test cells into the network. A test cell brings back to the source the one-way delay from source to destination. As in all distributed systems, the clocks in source and destination nodes may be out of synchronization, giving rise to errors in the measured one-way delay. However, the feedback errors in two consecutive cells can be cancelled if we subtract one feedback value from the other. The result is the exact *change* on the one way delay in the past one round-trip time. This change of one-way delay is used to modify the intercell distance, which roughly corresponds to the peak rate. The adjustment function is linear in both congestion and noncongestion cases.

In [13], feedback is used to adjust the token rate in a Leaky Bucket mechanism. The rate adjustment is both overly optimistic and pessimistic in that the rate is immediately dropped to the minimum level when congestion is detected, or is abruptly raised to the maximum when there is no congestion. What is important in [13] is that the authors aim at achieving the aggregated QoS, i.e., the total QoS within the network and at its entrance. Most previous work either focused exclusively on the network performance, and overlooked the QoS at the network boundary, or vice versa.

In our work, as in [12], we take the change on delay into consideration in rate adjustment. However, the feedback is the maximum switch utilization along the virtual connection (as in [13]), rather than the change on one-way delay. There are two major differences between our work and others:

1. In all existing feedback-based mechanisms, feedback delay is not considered. In our work, feedback delay plays a central role.

2. We do not solely rely on a conservative rate adjustment principle, as in all other existing mechanisms. We use feedback and heuristic predictions to balance optimism and conservativeness.

4 Simulation Model

In Table 1 we list all parameters used in the simulator. The parameters are categorized based on the three types of important components: source, controller, and switch. The parameters related to the controller have been explained in the previous section. In this section we explain the parameters related to other components.

Table 1: Parameters for the Simulator

COMPONENT	PARAMETERS
Source	μ_0, μ_1, λ_0, λ_1
Controller	K_a, K_p, $r_a \in [r_m, r_M]$, r_p, $[C_m, C_M]$
Switch i $(1 \le i \le n)$	L_i, d_i, p_i, $x_i(j)$

4.1 The End-to-End Connection

The approach taken in design of the simulation (for reducing simulation complexity) is to explicitly represent only one connection and aggregate all other connections into "cross traffic" having general interarrival distributions. The distribution is set quite arbitrarily since it is not known what the actual distribution should be. A structural simulation model is reported in [14] to capture the fact that the cross traffic in a switch "deep" in the network would be heavier than what may be seen in a switch close to the network boundary. However, the "network boundary" may be difficult to identify.

The parameters used in the simulation are listed below.

- n: the total number of switches.
- L_i: the capacity of the queue in switch i along the virtual connection. The server of each queue has a constant service rate of one cell per unit time, and it follows the FCFS discipline.
- d_i: link propagation delay between switches i and $i+1$, $i = 1, 2, \ldots, n$. (Switch $n+1$ is the destination node.)
- $x_i(j)$: the probability of cross traffic cells' interarriving distance to be j in switch i, i.e., the cross traffic intercell distance is assumed to follow a discrete general distribution.
- p_i: cell deflecting probability after switch i, i.e., after having been served, a cell in switch i will not stay on the virtual connection with probability p_i. In the simulation, these deflecting cells are discarded.

It is also of interest to see how the traffic over the explicitly represented connection affects all other connections. The simulation results in Section 5 show that by adjusting the rate of the explicitly represented connection, all cross traffic streams' cell loss and delay decrease. Therefore, one can infer that even though switches do not employ any protection mechanisms (such as priority based scheduling mechanisms like Fair Queueing [15]), the QoS of all connections is guaranteed.

4.2 Traffic Source

The traffic source is modeled by a two state MMPP whose state transition rates are μ_0 and μ_1. In state 0 (1), cell arrivals follow the Poisson distribution with rate λ_0 (λ_1).

The peak rate of the MMPP model is simply the larger of λ_0 and λ_1, and the average rate is given by

$$s_a = \frac{\mu_1}{\mu_0 + \mu_1}\lambda_0 + \frac{\mu_0}{\mu_0 + \mu_1}\lambda_1 \qquad (14)$$

5 Performance Results

Three traffic control methods, the pure PCC, the EFCN with the well-known ***multiplicative decrease and additive increase (MDAI)*** rate adjustment, and the balanced mechanism proposed in this paper, are compared in four scenarios. In the first two scenarios, the virtual connection is persistently congested, and not congested at all, respectively. The condition of the connection does not change in these two scenarios. In the last two scenarios, the network condition changes at the end of the fifth round-trip time from no congestion condition to heavy congestion (in scenario 3) or from heavy congestion to no congestion (in scenario 4).

Scenario 1: A Heavily Congested Virtual Connection

Some of the parameters selected for this scenario are shown in Table 2. Four switches are assumed in this case; all have identical distributions of interarriving time for cross traffic cells (not shown in Table 2). The expected cross traffic intercell distance is 0.192. Therefore, the load of cross traffic cells on each switch is $1/(1+0.192) = 0.84$, a relatively heavy load.

The discrete general distribution also allows many cross traffic cells to arrive at the same time, i.e., a batch arrivals model. The batch size follows a geometric distribution with mean 4.76. Note that in Table 2, the buffer size L_i in each switch is 5. As such, buffers in all switches can easily become saturated. In both EFCN and the balanced mechanisms, the threshold switch utilization η^* is targeted at 0.5.

Table 2: Parameters for Scenario 1

COMPONENT	PARAMETERS
Source	$\mu_0 = 0.4, \mu_1 = 0.4, \lambda_0 = 0.333, \lambda_1 = 0.067$
Controller	Peak Rate UPC: $K_p = 10, r_p = 0.333$
	Average Rate UPC: $K_a = 100, r_a \in [0.05, 0.25]$
Switches $(n = 4)$	Switch 1: $L_1 = 5, d_1 = 500, p_1 = 0.5$
	Switch 2: $L_2 = 5, d_2 = 500, p_2 = 0.0$
	Switch 3: $L_3 = 5, d_3 = 500, p_3 = 0.5$
	Switch 4: $L_4 = 5, d_4 = 500, p_4 = 0.5$

Because the cross traffic cells' deflecting probability in switch 2 is $p_2 = 0$, the utilization of switch 3 becomes persistently higher than 0.5, as shown in Figure 3 where the utilization of switch 3 for the first 15 round trip time is plotted. Note that the utilization of switch 3 is also the feedback in the balanced mechanism, since all other switches have much lower utilizations.

Figure 3 shows that the two mechanisms EFCN(MDAI) and the balanced mechanism do not differ significantly in terms of the maximum switch utilization. Observing the maximum switch utilizations in Figure 3, one may conclude that congestion persists no matter which mechanism is being used, and the balanced mechanism appears not much better than the EFCN(MDAI). We will return to this point shortly after we have examined the QoS performance.

QoS measures show that the balanced mechanism is able to improve the performance on cell loss rate and cell delay for the source, as well as for all cross traffic. This conclusion is summarized from the results shown in Figures 4–7, where the virtual connection (VC) cell loss probability, cross traffic (CT) cell loss probability, VC cell delay, and CT cell delay, respectively, are plotted against switch index.

In the simulation, the initial average rate policer's token rate is selected as 0.167, lower than the average source rate $s_a = 0.2$ (computed using (14) in Section 4). In light of the existing congestion, this initial token rate may not be appropriate, i.e., it may need to be further lowered. In the case of pure PCC, the token rate, once selected, cannot be changed during the entire communication phase, while in the other two mechanisms, changes can be made based on the feedback.

The balanced mechanism clearly shows its advantages (Figures 4–7) in adjusting both cell loss and delay performance for both VC and CT cells. This performance should be expected since the balanced mechanism more closely follows the dynamic (and most recent) changes of the network condition, as argued in Section 3. Cells injected into the network using this mechanism are less likely to be dropped than in the EFCN(MDAI) case.

We now revisit Figure 3 which seems to suggest similarity between the balanced and the EFCN(MDAI) mechanisms. Given the fact that the QoS is substantially improved by the balanced mechanism, we conclude that compared with the EFCN(MDAI), the balanced mechanism achieves the QoS improvement without actually giving up the network utilization. This conclusion, which further demonstrates the advantage of the balanced mechanism, is not intuitively immediate; but it has been verified by all experiments we have conducted.

Scenario 2: No Congestion along the Virtual Connection

Parameters for Scenario 1 are changed to make the connection lightly loaded: (1) We double the buffer capacity in each switch to change it from 5 to 10. (2) We change the target switch utilization η^* from 0.5 to 0.8. (3) The distance between adjacent switches is increased from 500 time units to 1000. Therefore, links can function better as cell buffers. (4) The deflection probability in switch 2 is now changed from 0 to 0.8, thus only 20% cross traffic cells served by switch 2 can actually arrive at switch 3.

In a congestion-free connection, the optimal token rate in the average rate policer should be 0.2, i.e., the average rate of the source. Therefore, the initial token rate $r_a = 0.167$ needs be dynamically increased using the feedback mechanisms.

In Figure 8, cell delay results for cross traffic in all intermediate switches are plotted. (Other performance measures convey the same conclusion to be discussed below.) Figure 8 shows that the performance of the pure PCC mechanism is the best (using the fixed token rate of 0.167). The performance of the balanced mechanism is only slightly worse than the pure PCC mechanism, while the one for the EFCN(MDAI) is much worse.

We note that the pure PCC outperforms the feedback based mechanisms only under the condition that the parameter settings in the controller is optimally selected. In reality, however, to accurately configure the policers is often extremely difficult. As such, a feedback mechanism that performs close to the optimal pure PCC would be attractive. The balanced mechanism appears to be such a choice.

Scenario 3: From No Congestion to Heavy Congestion

To test the responsiveness of the feedback-based mechanisms, we allow switch 2's deflection probability to change from 0.8 to 0 at the end of the fifth round-trip time. The initial token rate in the average rate policer is 0.1.

In the pure PCC mechanism, the pessimistic token rate ($r_a = 0.1$) leads to unnecessary cell loss at the policer during the first five round-trips when no congestion exists along the VC; but it protects the switches during the period of congestion after the fifth round-trip. As the result, the pure PCC mechanism still performs the best in terms of protecting QoS in switches. This protection, however, is at the expense of the QoS over the *entire* virtual connection, as shown in Table 3. Table 3 shows that the pure PCC offers the best protection of switches, and the EFCN(MDAI) offers lenient control at the controllers but poor performance in switches. The balanced mechanism balances the QoS in these two types of components.

Table 3: Cell Loss over the Entire VC (Scenario 4)

MECHANISM	CELL LOSS PROBABILITY			
	P.R. Policer	A.R. Policer	Switches	Total
PCC	0	0.492	0.033	0.525
EFCN(MDAI)	0	0.345	0.086	0.431
Balanced	0	0.465	0.046	0.511

Scenario 4: From Heavy Congestion to No Congestion

In this last scenario, we use the same parameter settings as in Scenario 3, except that the deflection probability of switch 2 changes from 0 to 0.8 at the end of the fifth round-trip time. Therefore, the virtual connection is heavily congested in the first five round-trip time, and then the congestion disappears.

The QoS performance follows a similar pattern as observed in Scenario 3. Thus, we avoid repeating the same conclusion here. In Figure 9, the token rate r_a in the average rate policer versus round-trip index is shown. When the network changes from heavy congestion to no congestion, the token rate in the average rate policer is able to increase to the vicinity of the *optimal* 0.2. However, the EFCN(MDAI) increases its token rate indefinitely until the permissible upper limit r_M is reached. This figure clearly shows that the balanced mechanism controls the traffic more prudently and accurately than the EFCN(MDAI) mechanism.

6 Conclusion

In this paper we introduced a balanced feedback-based congestion control mechanism. Two other mechanisms, the pure PCC and the EFCN(MDAI) are compared with the balanced mechanism through simulation. The major findings of this work are summarized below.

First, The balanced mechanism is superior to the EFCN(MDAI) in satisfying the QoS expectations for both a virtual connection over which the balanced mechanism is used and for all cross traffic. Second, the balanced mechanism improves the QoS performance without sacrificing network utilization. Third, as the congestion condition changes within the network, the balanced mechanism is able to adjust the token rate in the LB policer to appropriate level both promptly and accurately. Finally, in all scenarios studied in our work, we have found that there are optimal parameter settings which make the pure PCC outperform all other mechanisms. In reality, an optimal parameter setting is often not known a priori, and must be promptly and accurately tracked using feedback-based mechanisms. We showed in this paper that the balanced mechanism satisfies this requirement.

References

[1] CCITT Working Group XVIII. *Meeting Document.* Unapproved Draft, June. 1990.

[2] J. Kurose. "Open Issues and Challenges in Providing Quality of Service Guarantees in High-Speed Networks". *Computer Communication Review*, Vol.23(No.1):6–15, Jan. 1993.

[3] M. Butto, E. Cavallero, and A. Tonietti. "Effectiveness of the 'Leaky Bucket' Policing Mechanism in ATM Network". *IEEE Journal on Selected Areas in Communications*, Vol.SAC-9:335–342, April 1991.

[4] E. P. Rathgeb. "Modeling and Performance Comparison of Policing Mechanisms for ATM Networks". *IEEE Journal on Selected Areas in Communications*, Vol.SAC-9:325–334, April 1991.

[5] G. Rigolio and L. Fratta. "Input Rate Regulation and Bandwidth Assignment in ATM Networks: an Integrated Approach". In *Proc. 13th International Teletraffic Congress*, pages 141–146, Copenhagen, Denmark, June 1991.

[6] F. Borgonovo and L. Fratta. "Policing Procedures: Implementations, Definitions, and Proposals". In *Proc. 13th International Teletraffic Congress*, pages 859–866, Copenhagen, Denmark, June 1991.

[7] B. A. Makrucki. "Forward Congestion Notification in ATM Networks". In *Proc. the 3rd Annual Workshop on Very High Speed Networks*, pages 136–148, Greenbelt, MD, March 1992.

[8] A. E. Eckberg, Jr., D. T. Luan, and D. M. Lucantoni. "Meeting the Challenge: Congestion and Flow Control Strategies for Broadband Information Transport". In *Proc. IEEE GLOBECOM '89*, pages 49.3.1–5, Dallas, TX, Nov. 1989.

[9] I. F. Akyildiz and Y. Gong. "Average Rate Enforcement for Real-Time Traffic in ATM Networks". Technical Report GIT-CC-92/08, College of Computing, Georgia Institute of Technology, Feb. 1992.

[10] A. Mukherjee and J. Strikwerda. "Analysis of Dynamic Congestion Control Protocols – A Fokker Plank Approximation". In *Proc. ACM SIGCOMM Conference*, pages 159–169, Zurich, Switzerland, Sept. 1991.

[11] K. W. Fendick, M. A. Rodrigues, and A. Weiss. "Analysis of Rate-Based Control Strategy with Delayed Feedback". In *Proc. ACM SIGCOMM Conference*, pages 136–147, Baltimore, MA, Aug. 1992.

[12] Z. Hass. "Adaptive Admission Congestion Control". *Computer Communication Review*, Vol.21(No.5):58–76, Oct. 1991.

[13] W. Matragi and K. Sohraby. "Combined Open-Loop/Close-Loop Approach for Congestion Control in ATM Networks". In *Proc. International Conference on Communications (ICC)*, pages 1336–1342, May 1993.

[14] Y. David, Kurose J., Towsley D., and M. G. Hluchyj. "On Per-Session End-to-End Delay Distributions and the Call Admission". In *Proc. ACM SIGCOMM Conference*, pages 2–12, San Francisco, CA, Sept. 1993.

[15] A. Demers, S. Keshav, and S. Shenker. "Analysis and Simulation of a Fair Queueing Algorithm". *Journal of Internetworking Research and Experience*, pages 3–26, Sept. 1990.

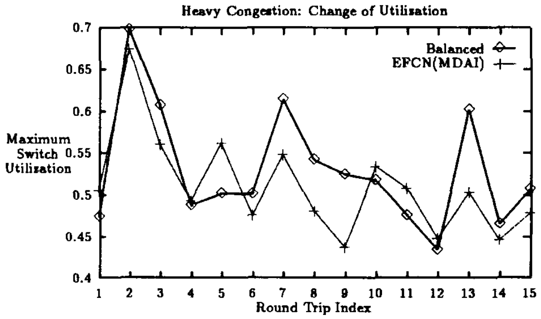

Figure 3: Maximum Switch Utilization along VC (Scenario 1)

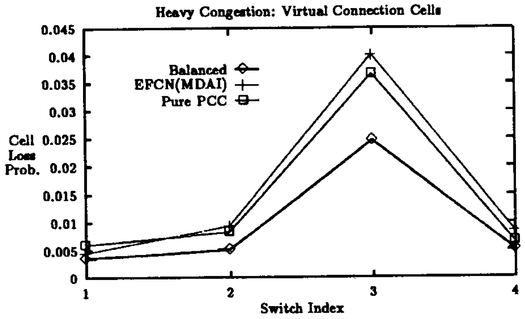

Figure 4: VC Cell Loss Probability along the Connection (Scenario 1)

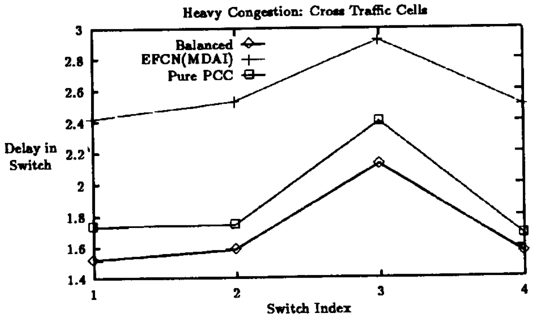

Figure 7: CT Cell Delay along the Connection (Scenario 1)

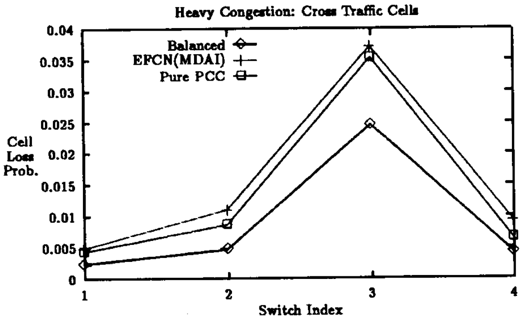

Figure 5: CT Cell Loss Probability along the Connection (Scenario 1)

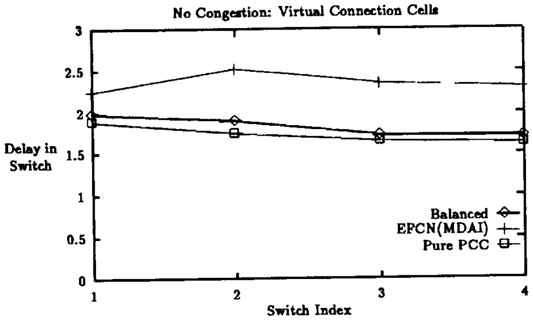

Figure 8: VC Cell Delay along the Connection (Scenario 2)

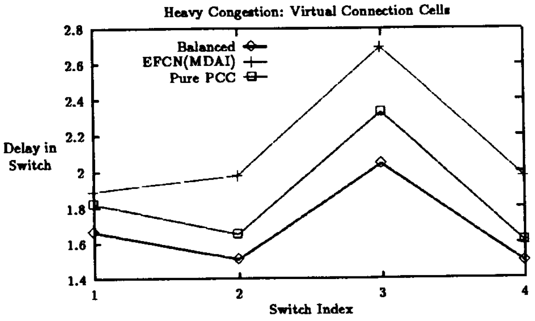

Figure 6: VC Cell Delay along the Connection (Scenario 1)

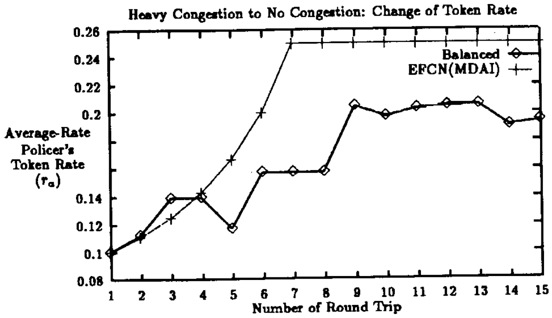

Figure 9: Change of the Token Rate (Scenario 4)