Burst Reduction Properties of the Leaky Bucket Flow Control Scheme in ATM Networks

Venkat Anantharam and Takis Konstantopoulos

Abstract—**The leaky bucket is a simple flow control scheme for ATM networks. An arriving cell can be transmitted only if it finds a token in the token buffer, in which case it is transmitted instantaneously by consuming a token. If the token buffer is empty, the cell has to wait until the generation of a new token. For purposes of analysis we assume an infinite cell buffer. The control parameter is the token buffer size C. We examine the burstiness of the output flow as a function of C and show that the burstiness increases with C. In particular the output flow is always less bursty than the input flow. This monotonicity simplifies optimal choice of the token buffer size. Our result is true for fairly arbitrary input flows and deterministic token generation times.**

I. Introduction

ATM (asynchronous transfer mode) is a standard for broadband ISDN networks, which are proposed to handle different traffic such as video, audio, file transfers, etc., using a unified protocol suite. Handling bursty traffic that needs real time reception introduces problems that are not encountered in conventional packet switching networks. It is desirable to smooth the traffic by a flow control scheme that should be fast and easy to implement and should ideally operate essentially open loop since feedback may be unacceptably delayed in a high-speed network. One of the proposed standards is the so-called *leaky bucket* control scheme.

We consider a version of the leaky bucket which operates as follows (see Fig. 1). The arrival process consists of packets of fixed size (53 bytes, having a 48 byte information field), known as *cells*. It is controlled by *tokens* that are stored in a buffer of fixed size C. An arriving cell can be transmitted only if it finds a token in the token buffer, in which case it is transmitted instantaneously by consuming a token. If the token buffer is empty the cell has to wait for the generation of a new token. We model the cell buffer as having infinite storage capacity, so that packets are not lost. Tokens are generated at a fixed rate, typically with constant interarrival times. A stored cell is transmitted immediately upon the generation of a new token.

Paper approved by M. Sidi, the Editor for Communications Networks of the IEEE Communications Society. Manuscript received March 1, 1992; revised September 17, 1992 and March 4, 1993. The work of V. Anantharam was supported by NSF PYI award NCR 8857731, an IBM Faculty Development award, Bellcore Inc., and the AT&T Foundation. The work of T. Konstantopoulos was supported by NSF research initiation award NCR 9211343; part of the work was done while the author was visiting faculty at the University of California, Berkeley, CA. This paper was presented in part at the 29th Allerton Conference on Communication, Control, and Computing, October 1991.

V. Anantharam is with the School of Electrical Engineering, Cornell University, Ithaca, NY 14853 USA.

T. Konstantopoulos is with the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX 78712 USA.

IEEE Log Number 9406148.

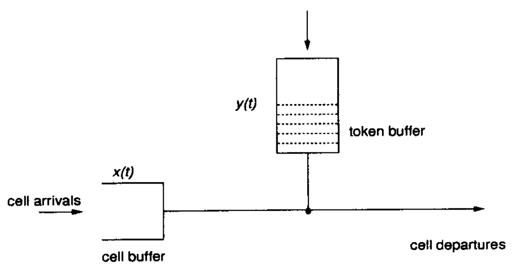

Fig. 1. The leaky bucket scheme.

Versions of the leaky bucket scheme have been investigated by several authors, see for example, Berger [3], Budka and Yao [4], Buttó *et al.* [5], Dittman *et al.* [6], Eckberg *et al.* [7], Kuang [8], Low and Varaiya [9], and Rathgeb [11] and their references. Our model is akin to that of [3] except that we assume an infinite cell buffer. In Section II we set up a stochastic model for the leaky bucket protocol. In Section III we describe a simultaneous pathwise construction of the cell departure process for all values of C from a single appropriately chosen state process. We define a burstiness ordering between the departure processes for various values of C in terms of the queueing caused by feeding it to another deterministic server queue. In Section IV we prove that the burstiness of the cell departure process decreases as C decreases. In particular, the leaky bucket scheme is *burst reducing* for any C. This monotonicity indicates that in design *one should choose the smallest token buffer size subject to acceptable overall delay*. What is important is that this design suggestion is valid for fairly arbitrary traffic processes. We investigate this design problem by simulation in Section V, using data for a video codec taken from [11].

Results of a similar nature have appeared in the recent literature, for instance [1], [4], [8], [9]. Our focus in this paper is more specific. The main tradeoff in implementing any flow control strategy is between the cell loss probability which is related to the probability of high cell buffer occupancy in the infinite buffer model, and the delay incurred by the accepted traffic in the network, which is related to queueing in the nodes of the network through Little's law. A natural parameter to effect such a tradeoff is the token buffer size, C; another parameter is the token generation rate. The focus of our paper is on analyzing the role of the token buffer size in effecting this tradeoff.

II. State Process and Stationary Regime

Let $x(t)(y(t))$, denote the number of cells (tokens), in the cell (token) buffer at time t. Thus, $0 \leq x(t) < \infty$ and

$0 \leq y(t) \leq C$. We may clearly assume that $x(t)y(t) = 0$ for all t. Thus, the system can be described by $z(t) = x(t) - y(t)$, an observation that is also made in [3]. If $z(t) > 0$ there are $z(t)$ cells and no tokens, if $-C \leq z(t) < 0$ there are $-z(t)$ tokens and no cells, and if $z(t) = 0$ there are neither cells nor tokens.

From now on all functions of time are assumed to be left-continuous. Let t_i, $i = 1, 2, \ldots$ (s_j, $j = 1, 2, \ldots$) be the cell (token) arrival times. We assume the cell and token arrival processes are stationary and ergodic. Normalize the token arrival rate to 1 and let λ denote the cell arrival rate. We make the convention that if a cell arrives at the time a token is generated, it is as if the cell arrived immediately after the token. Let $z_n = z(s_n-)$ denote the value of the process z immediately before time s_n (generation of nth token). If a_n denotes the number of cells arriving into the leaky bucket on the interval $[s_{n-1}, s_n)$, then z_n is given by the recursion

$$z_{n+1} = z_n + a_{n+1} - 1\{z_n - 1 \geq -C\}. \tag{1}$$

Defining the quantity $w(t) = z(t) + C = x(t) - y(t) + C$ and letting $w_n = w(s_n-)$ be the value of w just before s_n, we get the following recursion for w_n:

$$w_{n+1} = w_n + a_{n+1} - 1(w_n > 0) = (w_n - 1)^+ + a_{n+1} \tag{2}$$

where $x^+ = \max\{x, 0\}$. Note that w takes values in the set Z_+ of nonnegative integers. This is a familiar recursion from basic queueing theory [13]. The following result is due to Loynes [10]; see also Baccelli and Brémaud [2] and Walrand [13].

Theorem 1: If $\lambda < 1$ there exists a random process that satisfies the recursion (2) that is stationary and ergodic. Furthermore, this process is unique. If $\lambda > 1$ then, with probability 1, w_n converges to ∞ as $n \to \infty$.

While the condition $\lambda < 1$ is not necessary in practice for flow control with a finite cell buffer, the infinite buffer model loses its relevance when it does not hold. Thus we assume this condition. Note that the distribution of w does not depend on the choice of the token buffer size C. Nevertheless, the distributions of $z = w - C$, $x = \max\{w - C, 0\}$, and $y = \max\{C - w, 0\}$ do depend on C, as they should. In practice, the token generation times are deterministic and equally spaced in time. We will make this assumption in the subsequent sections.

III. The Leaky Bucket Departure Process

The departure process D^C from the leaky bucket with token buffer capacity C is the point process of (random) times at which cells are transmitted. We start by introducing an ordering between the departure processes corresponding to different values of C.

Definition 1: We say that D^{C_1} is less bursty than D^{C_2} if and only if the steady-state queue length in a deterministic single server queue with service time $\sigma < 1$ and arrival process D^{C_1} is stochastically smaller[1] than the steady state queue length in the same queue but with arrival process D^{C_2}.

[1] A random variable X is stochastically smaller than the random variable Y if $P(X \leq a) \geq P(Y \leq a)$, for all a; see, e.g., Ross, [12].

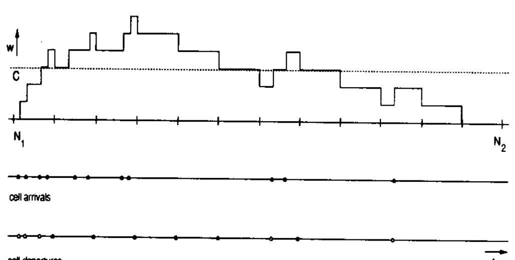

Fig. 2. Construction of the departure process. (The "o" denote departure times that coincide with arrival times; the "•" denote departure times of delayed cells).

Ideally, burstiness of the output of the flow control scheme should be measured through the delay encountered by the admitted flow in the network. Definition 1 attempts to capture this by the delay of the admitted flow in a deterministic server queue, which is somewhat simplistic, but at least amenable to exact analysis. The deterministic service times model the fact that cells are of fixed size. The condition $\sigma < 1$ corresponds to requiring that the network be able to handle *all* output flows from the flow control mechanism.

The departure process from the leaky bucket can be constructed from a sample path of the w-process as in Fig. 2. A cell departs either when w makes a transition from a level $i < C$ to $i + 1$ (in which case the cell is transmitted without delay) or when w makes a transition from a level $i > C$ to $i - 1$ (in which case the cell has a nonzero delay). Any other transition of w does not yield a cell departure. Note that the departure processes for a whole range of systems, corresponding to different values of C, can be read off directly from the sample path of w.

IV. Burst Reduction Properties of the Leaky Bucket

Suppose that tokens arrive at times $0, 1, 2, \ldots$. Our main result is the following.

Theorem 2: The leaky bucket departure process D^{C_1} is less bursty than D^{C_2} if $C_1 < C_2$.

Proof: Assume that the departure process is fed into a deterministic queue with service time $\sigma < 1$. Consider the systems with parameters $C_1 = C > 0$ and $C_2 = C + 1$. Consider a stationary left-continuous version of the w process, as in Section II. Let $N_1, N_2, \ldots$ be the times at which tokens are generated and lost, i.e., $N_1 = \inf\{i \in Z_+ : w(i) = 0\}$ and $N_{j+1} = \inf\{i \in Z_+ : i > N_j,\ w(i) = 0\}$, $j = 1, 2, \ldots$. Note that the N_j are integer-valued. Due to the construction of Section II these points can be taken to be the same in both systems. We refer to the piece of the process w between N_j and N_{j+1} as the jth cycle. Let $q^C(t)$ denote the queue length at the deterministic server queue at time t when the token buffer size is C.

Lemma 1: The stationary versions of the processes $\{q^C(t)\}$ and $\{w(t)\}$ are such that $q^C(N_i) = 0$, $i = 1, 2, \ldots$. Thus, the stationary queue length is zero at the beginnings of the cycles of w.

Proof of Lemma 1: Suppose that $N_1 = 0$, $N_2 = n$. We will prove that if $q^C(0) = 0$ then $q^C(n) = 0$. If $n = 1$

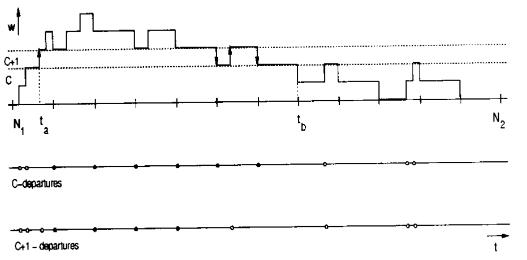

Fig. 3. Comparing the C- and $C+1$-departures. (The "o" denote departure times that coincide with arrival times; the "•" denote departure times of delayed cells.)

there is no arrival between 0 and 1, and thus the previous statement is trivially true. Thus, assume that $n > 1$. Let a_k denote the number of cells arriving during $[k-1, k)$. We clearly have $w_k = \sum_{i=1}^{k} a_i - (k-1)$ for $k = 1, \ldots, n-1$ and $w_n = 0$. Thus, $\sum_{i=1}^{k} a_i \geq k$ for $k = 1, \ldots, n-1$ and $\sum_{i=1}^{n} a_i = n-1$. Now, if we force the server of the deterministic queue to idle on certain intervals while keeping its arrival process (i.e., the departure process from the leaky bucket) fixed, then we can only increase the queue length pathwise. Suppose that $q^C(0) = 0$. Suppose the deterministic server is allowed to start working on exactly one customer at time k ($k = 1, 2, \ldots, n-1$) and is forced to idle between $k + \sigma$ and $k+1$. Let $\overline{q}^C(t)$ be the length of the queue under this modified policy with inserted idle times. Since there are exactly $n-1$ customers arriving on the interval $[0, n-1)$, we have $\overline{q}^C(n) = 0$. Since $q^C(t) \leq \overline{q}^C(t)$ for all t, we also have $q^C(n) = 0$. This concludes the proof of Lemma 1. $\square$

Let $\{v^C(t)\}$ be the workload process in the deterministic queue. Note that $q^C(t) = \lceil v^C(t)/\sigma \rceil$ where $\lceil x \rceil$ denotes the smallest integer k such that $k \geq x$. Thus, it suffices to show that the steady-state workload, call it v^C, is stochastically increasing in C. Let α be the expectation of $N_2 - N_1$ (conditional that N_1 is 0). For any level $l \geq 0$, let $T^C(l) = \int_{N_1}^{N_2} 1\{v^C(t) \leq l\}\, dt$. Then one has $P(v^C \leq l) = ET^C(l)/\alpha$. We must prove that $P(v^C \leq l) \geq P(v^{C+1} \leq l)$, for all l. Since α does not depend on C, it suffices to show that $T^C(l) \geq T^{C+1}(l)$, for all l.

Consider again the cycle of the process w between N_1 and N_2. In Fig. 3 below we have drawn the levels C and $C+1$ as well as the points corresponding to C-departures (departures when token buffer is of size C) and $C+1$-departures. If w does not exceed C throughout the cycle duration then the C- and $C+1$-departures coincide and the desired conclusion is immediate. So suppose that this is not the case.

Suppose that $N_1 = 0$ and $N_2 = n$. Recursively define

$$t_a^{(i+1)} = \inf\{t_b^{(i)} < t \leq n : w(t-) = C,\ w(t+) = C+1\}$$

$$t_b^{(i+1)} = \inf\{t_a^{(i+1)} < t \leq n : w(t-) = C,\ w(t+) = C-1\}, \quad \text{for } i = 0, 1, 2, \ldots,$$

with the convention that $t_b^{(0)} = 0$. Thus, $w(t) \geq C$ for $t \in S = \bigcup_{i \geq 1} [t_a^{(i)}, t_b^{(i)})$, while $w(t) \leq C$, otherwise. We claim that Lemma 2 below holds.

Lemma 2: $v^C(t) = v^{C+1}(t)$, for all $t \in [0, n) - S$.

Proof of Lemma 2: For convenience, we write $t_a(t_b)$ for $t_a^{(1)}(t_b^{(1)})$. We first observe that the workload is the same in both systems just before t_a: $v^C(t_a-) = v^{C+1}(t_a-)$. It thus suffices to prove that $v^C(t_b-) = v^{C+1}(t_b-)$. Note that 1) There is at least one C-departure between t_a and t_b; if there are more than one departures then they are equally spaced with all interdeparture times equal to 1 (these times correspond to downward transitions of the process w from levels higher than C). 2) The $C+1$-departures between t_a and t_b *occur at earlier times* than the C-departures and are not necessarily equally spaced; in fact, the discrepancies between the C- and $C+1$-departures correspond to transitions of w either from C to $C+1$ or from $C+1$ to C. 3) The number of C-departures between t_a and t_b is exactly equal to $\lfloor t_b - t_a \rfloor$, for the latter quantity is the number of transitions of w from a level $i > C$ to $i-1$ (note that $\lfloor x \rfloor$ denotes the largest integer k such that $x \geq k$); the number of $C+1$-departures is also $\lfloor t_b - t_a \rfloor$ (indeed, for every $C \to C+1$ transition between t_a and t_b there must be a corresponding $C+1 \to C$ transition). Now observe that the work brought by these $\lfloor t_b - t_a \rfloor$ departures on the interval between t_a and t_b is $\lfloor t_b - t_a \rfloor \sigma \leq t_b - t_a$. From the observations above we see that, if the deterministic server gives priority to this work that was generated between t_a and t_b, then this work will finish before t_b. Hence, we have $v^C(t_b-) = v^{C+1}(t_b-)$. Thus, $v^C(t) = v^{C+1}(t)$ for $t \in [t_b^{(1)}, t_a^{(2)})$. Hence, $v^C(t_a^{(2)}-) = v^{C+1}(t_a^{(2)}-)$. The argument can then be repeated. This finishes the proof of Lemma 2. $\square$

We now proceed to show that $T^C(l) \geq T^{C+1}(l)$ for all l. In view of Lemma 2, It is enough to show that $\int_{t_a^{(i)}}^{t_b^{(i)}} 1\{v^C(t) \leq l\}\, dt \geq \int_{t_a^{(i)}}^{t_b^{(i)}} 1\{v^{C+1}(t) \leq l\}\, dt$, for all $i \geq 1$. We do that for $i = 1$. The number of C-departures and the number of $C+1$-departures between t_a and t_b both equal $k = \lfloor t_b - t_a \rfloor$. Define $t_0 = \inf\{t_a < t < t_b : v^C(t) = 0\}$ with the convention that $t_0 = t_b$ if the set is empty. It is clear that $v^C(t) \leq v^{C+1}(t)$ for all $t \in [t_a, t_0)$. Thus, $\int_{t_a}^{t_0} 1\{v^C(t) \leq l\}\, dt \geq \int_{t_a}^{t_0} 1\{v^{C+1}(t) \leq l\}\, dt$. So if $t_0 = t_b$ the statement has been proved. Next, suppose that $t_0 < t_b$. Let $k_1 = D^C[t_a, t_0)$ be the number of C-customers arriving into the deterministic queue on the interval $[t_a, t_0)$. Since $v^C(t) > 0$ for $t_a < t < t_0$ we have $t_0 = v^C(t_a) + k_1\sigma + t_a$. Let $k_2 = D^C[t_0, t_b)$, $k_1' = D^{C+1}[t_a, t_0)$, $k_2' = D^{C+1}[t_0, t_b)$. Then we have $k_1 + k_2 = k_1' + k_2' = k$ and $v^{C+1}(t_0) = v^{C+1}(t_a) + k_1'\sigma - t_0 + t_a = v^C(t_a) + k_1'\sigma - t_0 + t_a = (k_1' - k_1)\sigma$. We may interpret the remaining workload $v^{C+1}(t_0)$ as being due to $k_1' - k_1$ customers entering the queue at time t_0. So, on the interval $[t_0, t_b)$, the number of $C+1$-customers entering the queue is $(k_1' - k_1) + k_2' = k - k_1 = k_2$ which is the same as the number of C-customers.

Since $v^C(t_0) = 0$ these k_2 customers will result in a workload process v^C that consists of k_2 nonoverlapping busy cycles of 1 customer each. We claim that any other arrangement of k_2 customers will yield a workload process that spends more time above any level l. To see this, given a workload process $\{v(t), t \in [t_0, t_b)\}$ we define $\{\overline{v}(t), t \in [t_0, t_b)\}$ by "squeezing out" the idle time intervals. Formally, we consider the time change $\phi(t) = t_0 + \int_{t_0}^{t} 1\{v(r) > 0\}\, dr$ and its inverse

$\phi^{-1}(s) = \sup\{t > t_0 : \phi(t) \le s\}$, and let $\overline{v}(s) = v(\phi^{-1}(s))$. Observe that $\int_{t_a}^{t_b} 1\{v(t) > l\}\,dt = \int_{t_a}^{t_b} 1\{\overline{v}(t) > l\}\,dt$ for all l. With $\overline{v}^C$ and $\overline{v}^{C+1}$ defined in terms of v^C, v^{C+1} respectively, we claim that $\overline{v}^{C+1} \ge \overline{v}^C$. Indeed $\overline{v}^C(s) = m\sigma + t_0 - s$ if $s \in (t_0+(m-1)\sigma,\ t_0+m\sigma)$, $m = 1, \ldots, k_2$, and we can also write, in a unique way, $\overline{v}^{C+1}(s) = g(s)+t_0-s$ where $g(s)$ is a piecewise constant nondecreasing process. Since $\overline{v}^{C+1}(s) \ge 0$, we have $g(s) \ge s-t_0$. Say that $s \in (t_0+(m-1)\sigma,\ t_0+m\sigma)$. Since $g(s)$ takes values $l\sigma$, $l = 1, 2, \ldots$, the inequality $g(s) \ge s - t_0$ actually implies $g(s) \ge m\sigma$. This concludes the proof of Theorem 2. □

V. Simulation Results and Experimental Observations

In choosing the token buffer size C of the leaky bucket flow control scheme, a small value of C is on one hand desirable, because it reduces the delay inside the network, but, on the other hand, it may increase the delay at the access point of the network, roughly translating into high cell-loss probability with a finite cell buffer. Here we report on some experimental observations for this design problem that were obtained via a discrete-event simulation for the leaky bucket followed by a deterministic queue. We modeled the source by a two-state Markov-modulated Poisson process (MMPP). The data for the source, which is a video codec, were taken from [11]. Recall that to define a Markov-modulated Poisson process with two states, say 0 and 1, we need a Markov chain X_t with values 0 or 1, with transition rates $q_0 = q(0, 1)$ and $q_1 = q(1, 0)$, and two Poisson processes, N_t^0 and N_t^1 with rates Λ_0 and Λ_1, respectively, with Λ_1 representing the output peak cell rate of the codec. The number of points N_t of the two state MMPP up to time t is then given by $N_t = \int_0^t 1\{X_s = 0\}\,dN_s^0 + \int_0^t 1\{X_s = 1\}\,dN_s^1$, while the mean intensity is given by $\lambda = (q_0\Lambda_1 + q_1\Lambda_0)/(q_0 + q_1)$. State 0 is supposed to be the low bit rate state and state 1 the high bit rate (bursty) state. Specifically, the following numbers are used: $\Lambda_0 = 0$, $\Lambda_1 = 10.5$ Mbps, $q_0 = 47.52$ s^{-1}, $q_1 = 81.36$s^{-1}. This gives a mean intensity $\lambda = 3.9$ Mbps. Each ATM cell has length 53 bytes or $53 \times 8 = 424$ b. So we have $\Lambda_1 = 24{,}941$ c/s (cells per second) and $\lambda = 9198$ c/s. We assume that the service rate (available bandwidth for our source) is $\mu = 16$ Mbps. Typically, the available bandwidth may be higher if the network is used only by a single user. The 16 Mbps value is chosen to study a situation where there are 5 to 6 other users using the network at full rate. Translated in cells per second, we have $\mu = 37{,}736$ c/s.

The code for the simulations was written in the C language.[2] In the graphs below C denotes the size of the token buffer, D_1 is the mean cell delay at the leaky bucket, D_2 is the mean cell delay in the deterministic server queue, and r is the rate at which tokens are generated. We vary r between the mean arrival rate (3.6 Mbps) and the service rate (16 Mbps). Specifically, we present plots for the following values of r: $r_1 = 8$ Mbps $= 18868$ c/s, $r_2 = 10$ Mbps $= 23585$ c/s, and $r_3 = 12$ Mbps $= 28302$ c/s. About 10^6 events were simulated in order to produce each point of the graphs (corresponding to specific values of C and r).

[2]We would like to thank Dr. J. Roychowdhury for his assistance with the code for the simulations.

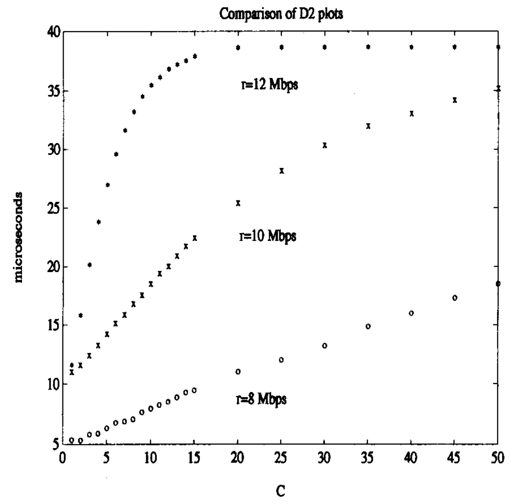

Fig. 4. Reduction of the network delay (D_2) due to leaky bucket flow control.

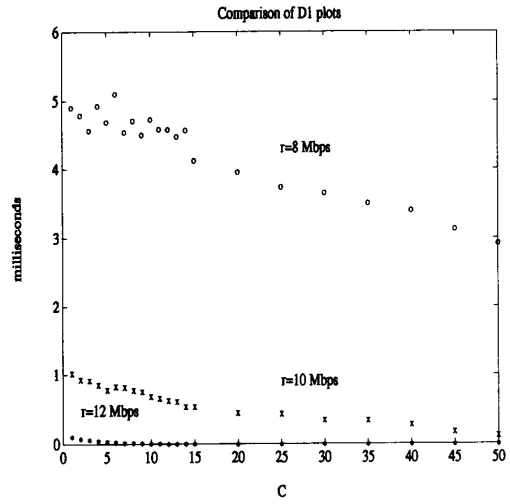

Fig. 5. Mean cell delay (D_1) at the leaky bucket.

When $C = \infty$ (leaky bucket is absent) the mean delays are $D_1 = 0$, $D_2 = 38.8$ μs. In all cases, we see that D_1 is strictly decreasing and D_2 is strictly increasing as C increases, thus verifying (partially) the results obtained theoretically. However, when r increases, the delay D_1 decreases and the delay D_2 increases. Thus, the biggest savings for the network delay are obtained by setting r to the lowest of the three values used (18868 c/s) (see Fig. 4). This, however, leads to a delay at the leaky bucket access point of the order of 5 ms, which is probably unacceptable (see Fig. 5). In Fig. 6, we see plots of D_2 and D_1 for $r = 28302$ c/s (the largest of the three values). We see that with a token buffer of size below 3 we achieve a 50% reduction in the network delay. From 38.8 μs, the delay drops to about 20 μs. At the same time we manage to

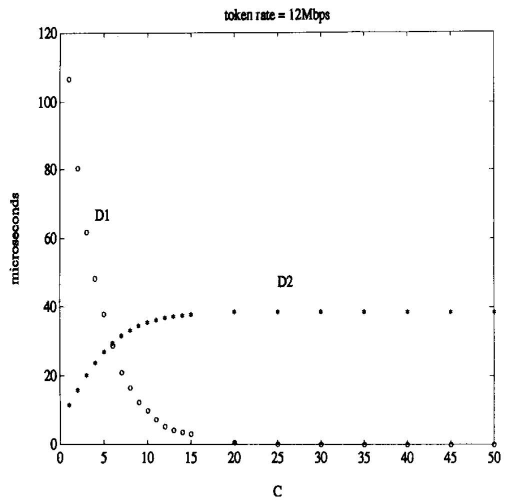

Fig. 6. For suitable token rate we can achieve a reduction in the network delay without introducing excessive delay at the network access point.

maintain the delay at the leaky bucket within the same order of magnitude.

VI. Conclusions

We proved that the output from the leaky bucket flow control scheme is less bursty than the input in that it incurs stochastically less delay in a deterministic queue. This is true for arbitrary stationary point process models for the source traffic. In fact the output process becomes less bursty as the size of the token buffer decreases. We verified our results by simulating a video codec source, modelled as a Markov-modulated Poisson process. Our simulations indicated that the increase of the cell delay at the leaky bucket can be tolerable provided that the token buffer size and token generation rate are chosen suitably.

References

[1] V. Anantharam and T. Konstantopoulos, "Burst reduction properties of leaky bucket strategies strategies in ATM networks," in *Proc. 29th Allerton Conf.,* 1991.
[2] F. Baccelli and P. Brémaud, "Palm probabilities and queues," *Springer Lect. Notes in Stat.,* 1986.
[3] A. W. Berger, "Performance analysis of a rate-control throttle where tokens and jobs queue," *IEEE J. Select. Areas Commun.,* vol. 9, pp. 165–170, 1991.
[4] K. C. Budka and D. D. Yao, "Monotonicity and convexity properties of rate control throttles," in *Proc. 29th Conf. Decision Contr.,* IEEE Press, pp. 883–884, 1990.
[5] M. Buttó, E. Cavallero, and A. Tonietti, "Effectiveness of the "leaky bucket" policing mechanism in ATM networks," *IEEE J. Select. Areas Commun.,* vol. 9, pp. 335–342, 1991.
[6] L. Dittman, S. B. Jacobsen, and K. Moth, "Flow enforcement algorithms for ATM networks," *IEEE J. Select. Areas Commun.,* vol. 9, pp. 343–350, 1991.
[7] A. E. Eckberg and D. M. Lucantoni, "A traffic/performance analysis of the bandwidth management throughput-burstiness filter," in *Proc. 29th IEEE Conf. Decision Contr.,* 1988, pp. 2118–2123.
[8] L. Kuang, "On the variance reduction property of buffered leaky bucket," preprint, 1992.
[9] S. Low and P. Varaiya, "A simple theory of traffic and resource allocation in ATM," in *Proc. GLOBECOM,* 1991.
[10] R. M. Loynes, "The stability of queues with non independent inter-arrival and service times," in *Proc. Cambridge Philosoph. Soc.,* vol. 58, 1962, pp. 497–520.
[11] E. P. Rathgeb, "Modeling and performance comparison of policing mechanisms for ATM networks," *IEEE J. Select. Areas Commun.,* vol. 9, pp. 325–334, 1991.
[12] S. Ross, *Stochastic Process.* New York: Wiley, 1983.
[13] J. Walrand, *An Introduction to Queueing Networks.* Englewood Cliffs, NJ: 1988.