

# OPTIMAL FLOW CONTROL SCHEMES FOR ATM NETWORKS

Takis Konstantopoulos <sup>\*†</sup> and Venkat Anantharam <sup>‡</sup>

Takis Konstantopoulos, ECE Dept., University of Texas, Austin, TX 78712  
Venkat Anantharam, EECS Dept., University of California, Berkeley, CA 94720

## Abstract

The problem of designing burst reducing flow controllers for traffic in an ATM network is studied. By requiring that the output flow obey certain burstiness constraints, it is shown that an optimal design exists and can be easily implemented in real time. Two versions of the problem are considered. The first one places constraints on the buffer size and the second one on the maximum delay that a cell can experience. The problems are solved both in discrete and continuous time and for arbitrary traffic processes. As a by-product of our analysis and methods, the optimality of the popular leaky bucket flow control scheme is also established.

## 1 Introduction

In this paper we examine some fundamental problems in the design of flow control for ATM based B-ISDN. These networks are expected to handle new types of traffic with widely differing and bursty characteristics. There is a serious need to incorporate explicit burstiness modelling in the traffic models used for performance analysis in these networks. Several analyses of samples of traffic have pointed to the deficiency of traditional traffic models in the new applications context; see, for instance, Beran et. al. [4], and Jagerman and Melamed [6].

In this paper we treat the problem of the design of flow controllers that produce output traffic with prescribed constraints on burstiness. To represent burstiness, we employ a simple and versatile class of models which was introduced by Cruz [5], and has been further developed by Low and Varaiya, [9], [10]. We say a traffic process obeys  $(\sigma, \rho)$  constraints if, when fed to a server

with infinite buffer and service rate  $\rho$ , the backlog never exceeds  $\sigma$ . There are a number of reasons why this class of flows is useful to consider. First of all, the above is an intuitively appealing characterization of burstiness, measured by the backlog induced in a simple queueing system, and as such is amenable to analysis and so to intelligent design choices. Secondly, it is a very simple characterization, which mimics to some extent aspects of the proposed ATM standards. Users may have to negotiate the parameters of the flows that they offer the network: the parameters  $\sigma$  and  $\rho$  are a simple pair with which to work, representing burstiness and average offered load.

One of the most popular flow control schemes for ATM traffic is the "leaky bucket" flow control scheme; see, for example, Mitra et. al., [11], and Turner [12]. This scheme has been shown in practice to be quite effective in regulating burstiness and is also simple to implement. In [2], we analyzed a model of the leaky bucket, driven by a stationary and ergodic arrival point process, and showed that it is burst-reducing, in the sense that the delay induced by its output is stochastically reduced. A similar result was also obtained independently by Low and Varaiya, [9] and by Kuang, [7]. It is interesting to note that the output flow from bucket obeys  $(\sigma, \rho)$  constraints.

Previous work on flow control for high-speed networks has focused mainly on the performance evaluation and comparison between different proposed schemes. In this paper we take a prescriptive, design-oriented approach: We are given a target burstiness parameter pair  $(\sigma, \rho)$ . The flow controller is allowed to delay incoming cells by at most  $d$  time units,  $d \geq 0$ . At each point of time the flow controller can either reject or transmit part of the delayed cells or the incoming traffic. The decision is allowed to depend on the *entire past* of the input process up to time  $t$ . The only requirement is that the flow control scheme (policy) should create an output traffic stream that satisfies the  $(\sigma, \rho)$  constraints. The goal is to find the best such policy in the sense that the rejected traffic is as small as possible. It turns out that there is an optimal

<sup>\*</sup>Research supported in part by NSF grant NCR 92-11343

<sup>†</sup>Direct all correspondence to the first author; tel: 1-512-471-5977, fax: 1-512-471-5532, e-mail: [takis@alea.ece.utexas.edu](mailto:takis@alea.ece.utexas.edu)

<sup>‡</sup>Research supported in part by NSF grant NCR 88-57731

policy that is simple in nature and can be implemented in real time by *recursively* updating a quantity we call the *virtual backlog*. In particular, the entire relevant information about the past is summarized by a single number!

A second design problem places a bound  $K$  on the maximum number of cells that can be stored in the flow controller's memory, the other elements being the same. Once again, it turns out that there is a simple optimal policy based on recursively updating a virtual backlog. Further, this policy is essentially the leaky bucket scheme! Thus, as a by-product of our analysis, we obtain the optimality of the leaky bucket. This result is, to the best of our knowledge, the first one concerning the optimality of the leaky bucket.

Below we study both discrete and continuous time versions of the design problems.

## 2 The discrete time flow controller with delay constraints

The system operates in slotted time. We denote by  $a_n$  the amount of traffic arriving at time  $n = 0, 1, \dots$ . This incoming traffic is arbitrary - in particular, it may be highly bursty. The role of a flow controller is to produce an output traffic stream, denoted by  $b_n$  satisfying prescribed burstiness constraints. We say that  $\{b_n\}$  is  $(\sigma_0, \sigma, \rho)$  constrained iff

$$\sum_{\ell=0}^{n-1} b_\ell \leq \sigma_0 + \rho n, \quad n \geq 1, \quad (1)$$

$$\sum_{\ell=m}^{n-1} b_\ell \leq \sigma + \rho(n-m), \quad n > m \geq 1. \quad (2)$$

The constant  $\sigma_0$  is taken to be smaller than or equal to  $\sigma$  and sets up the initial conditions. Recursively define the *virtual backlog* by

$$\sigma_{n+1} = \min(\sigma, \sigma_n + \rho - b_n).$$

It is easily seen that (1) and (2) are equivalent to

$$\sum_{\ell=0}^{n-1} b_\ell \leq \sigma_n + \rho n, \quad n \geq 1.$$

We now assume that there is a constraint  $d$  on the delay that incoming traffic can experience. Denote by  $m_n^i$ ,  $i = 1, \dots, d$ , traffic that is in the system at time  $n$  and which arrived at time  $n - i$ . The evolution of  $m_n^i$ ,  $i = 1, \dots, d$ , depends on the particular flow control policy. The policy can be arbitrary, as long as

- (i) at each time  $n$  it transmits an amount of traffic equal to  $b_n$  such that

$$b_n \leq a_n + m_n^1 + \dots + m_n^d,$$

- (ii) the transmitted traffic stream is  $(\sigma_0, \sigma, \rho)$  constrained, for given  $\sigma, \rho$ , and
- (iii) the decision taken at time  $n$  depends only on information up to time  $n$ .

We are interested in designing an optimal flow control policy, in the sense that the throughput is as large as possible. A feasible policy is any policy that satisfies (i)–(iii) above. This problem is solved in the theorem that follows. There is a policy that maximizes the amount of traffic transmitted over a time interval  $[0, n]$ , simultaneously for all  $n$ . This is just the greedy myopic policy that releases as much traffic as possible subject to the constraint imposed by the virtual backlog of the released traffic, releasing older traffic first.

**Theorem 1** Given  $d, \sigma_0, \sigma \geq \sigma_0, \rho, m_0^1, \dots, m_0^d$ , and an arrival sequence  $\{a_n, n = 0, 1, \dots\}$ , let  $\sigma_0^* = \sigma_0, m_0^{*i} = m_0^i, 1 \leq i \leq d$ . Consider a policy which, at time  $n$ , releases an amount of traffic  $b_n^*$ , recursively defined through

$$b_n^* = \min\{\sigma_n^* + \rho, a_n + m_n^{*1} + \dots + m_n^{*d}\} \quad (3)$$

$$\sigma_{n+1}^* = \min\{\sigma, \sigma_n^* + \rho - b_n^*\} \quad (4)$$

$$m_{n+1}^{*1} + \dots + m_{n+1}^{*i} = a_n + \min\{m_n^{*1} + \dots + m_n^{*i-1}, m_n^{*1} + \dots + m_n^{*d} - b_n^*\},$$

$$i = 1, \dots, d, \quad (5)$$

for  $n = 0, 1, \dots$ . Then  $\{b_n^*\}$  is  $(\sigma_0, \sigma, \rho)$  constrained, and for any feasible policy  $\{b_n\}$ , we have

$$\sum_{\ell=0}^n b_\ell^* \geq \sum_{\ell=0}^n b_\ell, \quad n \geq 0. \quad (6)$$

The theorem can be proved by forward induction arguments. For a complete proof see Konstantopoulos and Anantharam [8].

## 3 Flow control in continuous time

In this section we consider the design problems in continuous time. Now induction type arguments cannot be used in the construction of optimal flow controllers. We need a more global approach to the matter. It turns out that the

structure of the optimal flow controller relies on the construction and properties of the so-called *reflection mappings*. Their use not only achieves easier proof schemes but also allows for a wider interpretation of the burstiness constraints.

A *flow controller* is a map  $\varphi : A \rightarrow B$ , that takes an incoming traffic process  $A$  and produces an outgoing process  $B$ , such that  $B_t \leq A_t$  for all  $t$  (here  $A_t$  denotes the amount of traffic over the interval  $(0, t]$ ). The latter is a causality requirement. It is required that  $B$  obey prescribed burstiness constraints: given a *shape function*  $R_t$  and  $\sigma \geq \sigma_0 \geq 0$ , we need

$$B_t \leq \sigma_0 + R_t, \text{ for all } t > 0,$$

$$B_t - B_s \leq \sigma + R_t - R_s, \text{ for all } t \geq s > 0. \quad (7)$$

A shape function  $R_t$  is a non-decreasing function and is also a parameter of the design. For instance,  $R_t = \rho t$  reduces to the familiar linear constraint.

Two types of burst-reducing flow controllers are considered. First, suppose that a buffer of size  $K$  is available for storing incoming packets. Any packet finding the buffer full is automatically rejected. A burst-reducing flow controller is said to be *buffer-constrained* if, besides (7) it also satisfies

$$B_t - B_s \leq K + A_t - A_s, \quad t \geq s > 0. \quad (8)$$

Note that (8) precisely expresses the fact that a buffer of size  $K$  is used.

Secondly, suppose that an incoming packet can be delayed by at most  $d$  time units, by which time it has to be either transmitted or rejected. We can express this by the constraint

$$B_t - B_s \leq A_t - A_{s-d}, \quad t \geq s > 0. \quad (9)$$

This defines a *delay-constrained* flow controller.

We pose the following design problems.

**Problem 1.** Given  $\sigma_0, \sigma, R_t, K$ , design a *buffer-constrained flow controller*  $\varphi^* : A \rightarrow B^*$  such that

$$B_t^* \geq B_t, \quad t \geq 0,$$

for any other flow controller with the same parameters, producing an output stream  $B$ .

**Problem 2.** Given  $\sigma_0, \sigma, R_t, d$ , design a *delay-constrained flow controller*  $\varphi^* : A \rightarrow B^*$  such that

$$B_t^* \geq B_t, \quad t \geq 0,$$

for any other flow controller with the same parameters, producing an output stream  $B$ .

A practical and elegant solution to both these problems can be given. It relies on the so-called reflection mapping which is briefly described in the sequel. For more details see [8].

**Reflection mapping.** Let  $x_t$  be a function of time thought of as the unrestricted motion of a particle. Let  $\alpha_t, \beta_t$  be two functions such that  $\alpha_t \leq \beta_t$  and  $\alpha_0 \leq x_0 \leq \beta_0$ . We want to find a way to modify the motion  $x_t$  so that it stays between  $\alpha_t$  and  $\beta_t$  for all times  $t$ . This can be intuitively done by instantaneously reflecting the particle upwards (resp. downwards) whenever it hits the lower (resp. upper) boundary. The resulting motion, call it  $q_t$ , is uniquely defined. The mapping from  $x$  into  $q$  is the mathematical object that is known as reflection mapping; it enjoys properties such as *causality* and *minimality*; c.f. [8]. Furthermore,  $q$  has a representation as

$$q_t = x_t + \ell_t - u_t, \quad (10)$$

where  $\ell_t, u_t$  are non-decreasing functions; in fact  $\ell_t$  can increase only when  $q_t = \alpha_t$  and  $u_t$  can increase only when  $q_t = \beta_t$ .

Here is how reflection mappings are used in the design.

**Design of optimal flow controller with buffer constraints.** Given the incoming traffic process  $A$ , we shall describe how  $B^*$  is obtained. Consider the free process  $x_t = \sigma - \sigma_0 + A_t - R_t$  and reflect it at  $\alpha = 0$  and  $\beta = \sigma + K$ . Let  $q_t^*$  be the reflected process. The formula that corresponds to (10) is

$$q_t^* = [\sigma - \sigma_0 + A_t - R_t] + \ell_t^* - u_t^*. \quad (11)$$

Define  $\varphi^* : A \mapsto B^*$  by

$$B_t^* := \begin{cases} A_t - u_t^* & \text{if } q_t^* < \sigma \\ \sigma_0 + R_t - \ell_t^* & \text{otherwise.} \end{cases} \quad (12)$$

The above equations form the definition of the flow controller. For the proof of its **feasibility** and **optimality** see [8]. The interpretation of the optimal flow controller is especially interesting in this case, since, it turns out that it is the *leaky bucket* flow control scheme. (see Section 5 below).

**Design of optimal flow controller with delay constraints.** Here, given an incoming traffic process  $A$ , we define the boundaries

$$\alpha_t = 0, \quad \beta_t = \sigma + \psi_t,$$

where  $\psi_t = A_t - A_{t-d}$ . We perform the reflection of the same free process as above, namely  $\sigma - \sigma_0 + A_t - R_t$  at the new boundaries. One gets (11) for the reflected process and (12) for the output process  $B^*$ . The form of the equations is the same. What has changed is that we

use different boundary processes. In particular, the upper boundary is now time-varying (c.f. [8]).

A particular case that is common to both problems is that of  $d = 0$  or  $K = 0$ . We refer to this flow controller as *instantaneous flow controller*.

The above designs are very general. One can say that the problems have been posed and solved in a model-free context. Since the description of the flow control designs is rather abstract (due to the unavoidable use of reflection mapping) one is interested in explicit algorithms that can be used for their implementation. Such algorithms are indeed available in discrete time and are described in the sequel.

## 4 Implementation issues and algorithms

We assume throughout that all traffic processes  $A, B, R$ , etc., are supported on the positive integers, and we let

$$A_t = \sum_{k=1}^{\infty} 1\{k \leq t\} a_{k-1}, \quad B_t = \sum_{k=1}^{\infty} 1\{k \leq t\} b_{k-1},$$

$$R_t = \sum_{k=1}^{\infty} 1\{k \leq t\} r_{k-1}, \text{ etc.}$$

**Algorithm for the optimal flow controller with buffer constraints.** We show (for complete derivation see [8]) that the flow controller  $\varphi^*$  defined by (11) and (12) can be easily implemented by the following algorithm :

$$b_n^* = (a_n + \lambda_n^*) \wedge (r_n + \sigma_n^*)$$

$$\lambda_{n+1}^* = (\lambda_n^* + a_n - b_n^*) \wedge K$$

$$\sigma_{n+1}^* = \sigma \wedge (\sigma_n^* + r_n - b_n^*)$$

Note that the quantity  $\sigma_n^*$  is the virtual backlog, encountered in Section 2. The physical meaning of this algorithm is clear: send as much flow forward as possible subject to the burstiness constraint given by the virtual backlog. Retain as much of the remaining flow as possible in the buffer (up to  $K$  units of flow). Older traffic is sent forward first.

One observation that is probably worth mentioning at this point is that the algorithm remains valid even if the traffic processes do not necessarily have jumps on the integers  $k = 1, 2, \dots$ , but on an increasing sequence of times  $T_1, T_2, \dots$ .

**Algorithm for the optimal flow controller with delay constraints.** The algorithm for the flow controller

with delay constraints takes the following form:

$$b_n^* = (a_n + \lambda_n^*) \wedge (r_n + \sigma_n^*)$$

$$\lambda_{n+1}^* = (\lambda_n^* + a_n - b_n^*) \wedge (a_{n-d+1} + \dots + a_n)$$

$$\sigma_{n+1}^* = \sigma \wedge (\sigma_n^* + r_n - b_n^*)$$

It can be seen that this is a simplified form of the algorithm obtained in Theorem 1.

## 5 Optimality of the leaky bucket flow controller

The leaky bucket is a flow controller that operates as follows : Cells arrive according to some (bursty) traffic process  $A$ . Their transmission is controlled by objects called tokens. Tokens are generated according to a process  $R$  (in practice  $R$  is a periodic process with rate  $\rho$ , i.e., one token is generated every  $\rho^{-1}$  time units) and are stored in the token buffer that can hold up to  $\sigma$  tokens. An arriving cell that finds a token in the token buffer is transmitted instantaneously; otherwise, it waits till the generation of the next token. For more details see [2].

Now consider the flow controller  $\varphi^*$  which is the solution of Problem 1. Its evolution equations in discrete time are given in Section 4. Let  $q_t^*$  be the reflected process (11). Interpret  $(\sigma - q_t^*)^+$  as the number of tokens and  $(q_t^* - \sigma)^+$  as the number of cells in the system at time  $t$ . It is seen that  $B_t^*$ , given by (12), behaves exactly as the output process of a leaky bucket. We thus have the following result:

**Fact 1** *The optimal flow controller that produces a  $(\sigma_0, \sigma, R)$  constrained output traffic process and uses a buffer of size  $K$  is a leaky bucket with token buffer size  $\sigma$ , cell buffer size  $K$ , token arrival process  $R$  and initial number of tokens  $\sigma_0$ .*

A leaky bucket with  $K = \infty$  is referred to as a  $(\sigma, \rho)$  regulator (c.f. [5]) In [2] we showed that this leaky bucket is burst-reducing in the following sense: Let  $A$  be a stationary and ergodic point process with rate  $\lambda$  and let  $R$  be deterministic with rate  $\rho > \lambda$ . Consider the output process as an input to a queue with deterministic service rate  $\mu > \rho$ . Then the steady state queue length in the latter queue increases stochastically as  $\sigma$  increases.

## 6 Concluding remarks

The problem of designing optimal flow controllers that regulate the burstiness of traffic, while guaranteeing a bound on the number of buffered cells or on the delay, was solved in a very general framework. In a sense, our result is “model free”: no specific assumptions on the input model have been used. There are two important methodological points that should be emphasized. First, the burstiness constraints of a traffic stream can be summarized by the virtual backlog, which is a recursively updatable real number summarizing the entire past information of a burstiness constrained flow for many purposes. This simple realization in fact allows one to pose and study a new class of control and optimization problems of substantial relevance to the design of networks handling burstiness constrained flows, [1]. Second, in this work, we realized that constraints on the buffer size and constraints on the delay can be thought of as “generalized  $(\sigma, \rho)$  constraints” that can be expressed via a reflection mapping on a suitable “free process” (Section 3). The analysis then hinges on some structural properties of this reflection mapping. It is likely that this simple realization will be useful in other optimization problems as well, for instance in a multi-user environment.

Since the problems we considered were formulated in a model free context, one could very well pose them in a stochastic context as well, by requiring, for instance, that the arrival process be an arbitrary stochastic process. The greedy flow controller  $\varphi^*$  is then seen to minimize the average loss rate when such averages exist. Indeed, our results hold pathwise.

The leaky bucket flow control scheme was shown to be optimal among the class of causal flow controllers. This is a new result, to the best of our knowledge, that should be comforting to network engineers, who plan to use it in any case. In fact, it should be stressed that nowhere in the analysis did we make explicit use of the causality of the decision rule of a feasible flow controller (except, of course, that we cannot borrow flow from the future). Thus, the leaky bucket (a causal controller by itself) is optimal even when one allows the flow control decision to be made noncausally.

## References

- [1] Anantharam, V. (1992) An approach to the control of ATM Networks. Talk presented at IEEE International Symposium on Information Theory, Bahia, Brazil, July 1992. See also *Proc. of the Intern. Teletraffic Congress*, Indian Inst. of Service, Bangalore, Nov. 1993, 163-167; and *Proc. 32nd IEEE CDC*.
- [2] Anantharam, V. and Konstantopoulos, T. (1991). Burst reduction properties of the leaky bucket flow control scheme in ATM networks. *Proc. 29th Allerton Conf.*, 302-309. Subm. to *IEEE Tr. Comm.*
- [3] Anantharam, V. and Konstantopoulos, T. (1993). An optimal flow control scheme that regulates the burstiness of traffic subject to delay constraints. *Proc. 32nd IEEE CDC*, 3606-3610.
- [4] Beran, J., Sherman, R., Taquq, M., and Willinger, W. (1992). Variable-bit-rate Video traffic and Long-range dependence. *Preprint*, Bell Communications Research, Morristown, NJ.
- [5] Cruz, R.L. (1991). A calculus for network delay, part I: network elements in isolation. *IEEE Tr. Inf. Th.*, **37**, 114-131.
- [6] Jagerman, D., and Melamed, B. (1992). The Transition and Autocorrelation Structure of TES Processes. Part I: General Theory; Part II: Special Cases. *Communications in Statistics and Stochastic Models*, Vol. 8, No. 1, pp. 194 -219; Vol. 8, No. 3, pp. 499 -527
- [7] Kuang, L. (1992) On the Variance Reduction Property of Buffered Leaky Bucket. *Preprint*.
- [8] Konstantopoulos, T. and Anantharam, V. (1993). Optimal Flow Control Schemes that Regulate the Burstiness of Traffic. U.T. Austin Tech. Report SCC-93-15.
- [9] Low, S. and Varaiya, P. (1991). A simple theory of traffic and resource allocation in ATM. *Proc. Globecom*, 1633-1637.
- [10] Low, S. (1992). *Traffic management of ATM networks: Service provisioning, Routing, and Traffic shaping.*, Ph.D. Dissertation, University of California, Berkeley.
- [11] Mitra, D., Mitrani, I., Ramakrishnan, K.G., Seery, J., and Weiss, A. (1991). A unified set of proposals for control and design of high-speed data networks. *Queueing Systems: Theory and Applications*, Vol. 9, Nos. 1-2, 215-234.
- [12] Turner, J. (1986). New Directions in Communications (or Which Way in the Information Age). *IEEE Communications Magazine*, Vol 24, 8 -15.