Feedback Control of Congestion in Packet Switching Networks: The Case of a Single Congested Node

Lotfi Benmohamed, *Member, IEEE*, and Semyon M. Meerkov, *Fellow, IEEE*

Abstract— **This paper addresses a rate-based feedback approach to congestion control in packet switching networks where sources adjust their transmission rate in response to feedback information from the network nodes. Specifically, a controller structure and system architecture are introduced and the analysis of the resulting closed loop system is presented. Conditions for asymptotic stability are derived. A design technique for the controller gains is developed and an illustrative example is considered. The results show that, under appropriately selected control gains, a stable (nonoscillatory) operation of store-and-forward packet switching networks with feedback congestion control is possible.**

I. Introduction

WIDE area networks are store-and-forward backbone networks consisting of switching nodes and communication links connecting them according to a certain topology. Each link is characterized by its packet transmission capacity (packets per unit time). Each node is characterized by its packet storing capacity (buffer length) and its packet processing capacity (packets per unit time). A node which reaches its maximum storing capacity due to the saturation of its processors or one or more of its outgoing transmission links is called *congested*. Some of the packets, arriving at a congested node, cannot be accepted and have to be retransmitted at a later time. This leads to a deterioration of the network's throughput (number of packets delivered per unit time) and delay performance. Therefore, congestion prevention is an important problem of packet switching networks management (see [1]–[9] for several classical and recent publications).

It follows from the above that congestion is a result of a mismatch between the network resources (buffer space, processing and transmission capacity) and the amount of traffic admitted for transmission. Consequently, congestion prevention can be interpreted as the problem of matching the admitted traffic to the network resources. This, in turn, could be viewed as a classical problem of feedback control (matching the output to the input of dynamical systems) and, indeed, has been recognized as such by many researchers [10]–[22].

Manuscript received August 11, 1992; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editot D. Mitra. This work was supported in part by the US Army Research Office under Grant DAALO3-90-D-0094 and by a Graduate Fellowship from The University of Michigan.

L. Benmohamed was with The University of Michigan, Ann Arbor, MI. He is now with the Advanced Systems Division, National Institute of Standards and Technology, Gaithersburg, MD (email: l.benmohamed@ieee.org).

S. Meerkov is with the Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109-2122 (email: smm@eecs.umich.edu).

IEEE Log Number 9215398.

Two window-based algorithms were proposed in [10] and [11]. A simulation study, comparing the Jacobson algorithm [10] and the DECbit algorithm [11], was reported in [12]. In [6], it was observed that the Jacobson algorithm, implemented in the Internet's Transmission Control Protocol, resulted in an oscillatory behavior. A version of the DECbit scheme, where the source rate is controlled with an additive increase/multiplicative decrease algorithm, was studied in [14], [15] and also it was shown that it exhibits oscillations. In [16], a feedback-based dynamic window adjustment algorithm was developed and analyzed using an asymptotic technique; its applications and simulations were reported in [17]–[18]. A rate-based equivalent of this window scheme was introduced and analyzed in [19], and again it was shown that the system oscillates around its equilibrium point. Other approaches to congestion control presented in [20]–[22] have not been analyzed from the stability point of view.

It follows from this overview that the previous work on feedback-based congestion control algorithms did not address the issue of oscillations suppression and did not provide formal methods for stabilizing congestion controllers design. Therefore, the main goal of this paper is to develop an analytical method for the design of congestion controllers that ensure good dynamic characteristics along with some fairness in resource allocations.

The approach used here is quite standard [23]: The level of network congestion is monitored through the occupancy x of the buffer with the control target being some threshold x^0. Based on the difference between x and x^0, the congestion controller associated with each link periodically calculates an *admission rate* and supplies all the sources of its traffic with the result of this calculation. In their turn, traffic sources reduce their transmission rate to the lowest level allowed by intermediate links.

Given this architecture, the main question is how the admission rates should be calculated so that the network performance is sufficiently good. We give here the answer to this question for a backbone network with a single congested node where the congestion is due to the saturation of one of its outgoing links. The extension to multiple congested nodes and links will be communicated elsewhere. The paper is structured as follows: Section II describes the network and the congestion controller. In Sections III and IV, analysis and design problems are addressed. A design example and numerical illustrations are presented in Section V and conclusions are given in Section VI. Due to space limitations, the proofs are omitted and can be found in [24].

II. MODELING

2.1. Assumptions

2.1.1. The Network:

i) The network employs a store-and-forward switching service where users are serviced without prior reservation.

ii) The network consists of S switching nodes and N communication links. Let $\mathcal{S} = \{1, 2, \ldots, S\}$ denote the set of nodes and $\mathcal{N} = \{1, 2, \ldots, N\}$ denote the set of links. For each node $a \in \mathcal{S}$, let $\mathcal{O}(a) \subset \mathcal{N}$ denote the set of its outgoing links.

iii) Each link has a transmission capacity $c_s = 1/\tau_s$ packets/s, where τ_s is the transmission time of a packet, and a propagation delay of τ_p s.

iv) Each node has a processing capacity of $1/\tau_{pr}$ packets/s, where τ_{pr} is the processing time of a packet. It is the time between the moment a packet is received by the node and the moment it is placed in the buffer of its outgoing link. The processing capacity of each node is assumed to be larger than the total transmission capacity of its incoming links.

v) The network load consists of traffic corresponding to all source-destination pairs (ab), a and $b \in \mathcal{S}$. The pair (ab) will be referred to as the (ab) connection, the (ab)-type traffic, or the (ab) flow. Let $\mathcal{C}$ denote the set of all such connections.

vi) For each (ab) connection, the source at node a sends packets to the destination at node b through a sequence of links referred to as the path of the connection and denoted by $p(ab)$. The routing policy which determines the path of each connection is assumed to be fixed in advance (static). Let $C(i)$ be the set of all connections (ab) which traverse link i.

vii) Each node has, for each of its outgoing links, a buffer for storing packets waiting to be transmitted. Let $x_i, i \in \mathcal{N}$, denote the number of packets buffered for transmission on link i and referred to as link i's buffer. Buffers are assumed to be of finite capacity $X < \infty$, where X is sufficiently large in comparison with the steady state buffer occupancy x^0 defined in assumption (viii) below.

2.1.2. The Control Architecture:

viii) Each node a has a congestion controller associated with each outgoing link $i \in \mathcal{O}(a)$. This controller periodically computes a traffic admission rate q_i based on a *control algorithm* (see below) that uses local information to node a: the difference between x_i and some threshold x^0, and also the control decision q_i at present and in the finite past.

ix) The control algorithm updates take place every T s, where T is the time to transmit c packets ($T = c\tau_s$), i.e., a new control update is generated after every c packets are transmitted. Thus, the controller time is slotted with the slot duration, $[n, n+1)$, $n = 0, 1, \ldots$, equal to T.

x) Each node sends the computed control information (q_i's) to the sources along a fixed feedback path, usually the reverse direction of the traffic (assuming bidirectional links which is often the case). This control information is serviced with high priority and is carried either in separate packets or along with data or acknowledgment packets. The sources respond to the control information received according to the *control protocol* defined below.

xi) Traffic at each link is serviced according to the First-Come-First-Serve (FCFS) priority discipline.

2.1.3. The Input Traffic:

xii) The input traffic is viewed as a deterministic fluid flow where packet boundaries are ignored.

xiii) During the settling time of the system, the input traffic, satisfying (xii), is constant but otherwise arbitrary and unknown.

xiv) For each connection (ab), let r^0_{ab} denote the rate (in packets per slot) of its offered traffic. Let $f^0_i = \sum_{(ab)\in C(i)} r^0_{ab}$ be the total offered rate of all connections flowing through link i. We assume that the input traffic is such that only one link is overloaded, i.e., there exists a link i_0 such that $f^0_{i_0} > c$ whereas $f^0_i < c$ for all $i \neq i_0$.

2.1.4. Remarks:

1) Assumption (i) means that the sources of the network traffic can be slowed down and do not need to reserve bandwidth. In integrated service networks, where some traffic types, such as voice or real-time video, cannot be slowed down and require a guaranteed transmission rate, we believe that our feedback congestion control scheme can coexist with a traffic admission algorithm. The latter makes decisions regarding bandwidth assignment to sources that require a guaranteed transmission rate whereas the former controls the sharing of some portion of the bandwidth among traffic sources that tolerate a certain level of time delay. The interaction between these two traffic management procedures needs to be investigated.

2) Assumption (iii) assumes that the network is homogeneous. A generalization to heterogeneous networks, where each link has a different transmission capacity and/or propagation delay, is straightforward.

3) Assumption (iv) means that links rather than processors are the bottlenecks. The case where processing is the bottleneck can be approached similarly. Note that τ_p/τ_s is the delay bandwidth product and is negligible for a slow speed network. Since τ_p is constant for a given transmission line but τ_s decreases as the transmission speed increases, the delay-bandwidth product is significant for high speed networks and represents the number of packets being propagated on the transmission line ("in-the-pipe" packets). The approach developed in this paper is applicable to both processing and transmission bottlenecks.

4) According to assumption (v), we distinguish between traffic types on the basis of their source and destination nodes, so that traffic from multiple users at some node a sent to one or more users at some node b constitutes a single connection between a and b. The congestion control algorithm proposed here can equally use other definitions of traffic types.

5) Although we have assumed in (vi) a static routing, our proposed congestion control architecture can operate under dynamic routing as well (see discussions in Section 4.5). The cardinality of $C(i)$ in (vi) is the maximum number of connections that might simultaneously compete for the transmission capacity of link i. The congestion controller has to evenly share this capacity among the competing connections.
6) Although buffers are assumed to be sufficiently large, the delay performance will degrade long before the buffer overflows and the threshold x^0 in assumption (viii) is introduced to insure a desired bound on the time delay in the steady state and to prevent congestion or underutilization of the transmission capacity during the transient periods of control.
7) As discussed in Section 4.4, a tradeoff is involved in the choice of the update period T defined in (ix). Shorter T's lead to better responsiveness to changes in the input traffic but require the processors to devote more time updating the feedback signals. The overhead resulting from this updating can be reduced if the control algorithm is implemented in such a way as (a) no updating is performed if it is known that the new value of the admission rate will be the same as the previous one (e.g., when the buffer level is not changing), and (b) when an updating takes place, the new value of the admission rate is transmitted to the traffic sources only when it differs from its previous value.
8) Assumption (xiii) could be understood in the sense that the input traffic is piecewise constant with the jumps occurring seldom enough so that the transients of the system have time to settle down between two consecutive jumps. This assumption is expected to hold for networks carrying aggregate traffic as it is implied by assumption (v). The reason is that, since many sessions contribute to a connection's traffic, each with a rate that is small relative to the total, session initiations are statistically counterbalanced by session terminations.

2.2. *Buffer Equations*

Although the offered traffic in backbone networks is typically random, assumption (xii) of the model (i)–(xiv) presumes a deterministic fluid description. This deterministic fluid model is the first-moment approximation of the underlying stochastic processes. One way of arriving at the fluid model is through averaging using asymptotic techniques [25], [26]. Indeed, let $\xi_i(n)$ denote the input process to link i (number of packets that arrive during time slot $[n, n+1)$) with distribution $P_{\xi_i}(n,k) \triangleq \text{Prob}\{\xi_i(n) = k\}$. Let $\tilde{x}_i(n)$ be the occupancy of the buffer of link i at time n. Then, the dynamics of buffer i is described by the following difference equation:

$$\tilde{x}_i(n+1) = \text{Sat}_X\{\tilde{x}_i(n) + \xi_i(n) - c\}, i \in \mathcal{N}, \quad (1)$$

where

$$\text{Sat}_a(z) = \begin{cases} 0 & \text{if } z < 0, \\ a & \text{if } z > a, \\ z & \text{otherwise.} \end{cases} \quad (2)$$

It has been shown in [25]–[26] that, if the buffer capacity X is much larger than the expected value of $\xi_i(n) - c$, $\forall n$, (3) can be approximated by a deterministic equation where $\xi_i(n)$ is substituted by its conditional expectation $f_i(n)$ to obtain the averaged equation:

$$x_i(n+1) = \text{Sat}_X\{x_i(n) + f_i(n) - c\}, i \in \mathcal{N}. \quad (3)$$

This approximation is rigorous in the following sense: If the averaged equation is asymptotically stable (which means that the solution $x_i(n)$ of (3) converges to a steady state value $\bar{x}_i$ as n approaches infinity), then, under some technical conditions, the trajectories $\tilde{x}_i(n)$ and $x_i(n)$ are close to each other in probability for all $n \in [0, \infty)$. Thus, the deterministic fluid-flow equation (3) can be used for the analysis of the network dynamics. To simplify the presentation and eliminate technical details, we begin directly from the fluid-flow description as stated in assumption (xii). In fact, this approximation is a typical assumption in the analysis of dynamic congestion control mechanisms [14], [15], [19], [20], [22].

Since $f_i(n)$, the total rate at which traffic is flowing through link i during the $[n, n+1)$ slot, is less than f_i^0 (admitted traffic $\leq$ offered traffic) and $f_i^0 < c$ for $i \neq i_0$ (assumption (xiv)), then

$$x_i(n) = 0, n = 0, 1, \ldots, \forall i \neq i_0, \quad (4)$$

assuming that buffers are initially empty. This implies that traffic will incur no queueing delay at buffers of nonoverloaded links.

Let $r_{ab}(n)$ denote the rate at which connection (ab) has been admitted to the network during $[n, n+1)$. Then, the total amount of traffic arriving at buffer i_0 during $[n, n+1)$ is

$$f_{i_0}(n) = \sum_{(ab) \in C(i_0)} r_{ab}\left(n - \hat{\tau}_{ai_0}^{ab}\right), \quad (5)$$

where $C(i_0)$ is defined in assumption (vi) and $\hat{\tau}_{ai_0}^{ab} = \tau_{ai_0}^{ab}/T$ is the delay, measured in time slots T, that the (ab) traffic incurs from its point of entry to the network at node a until it arrives at buffer i_0, while $\tau_{ai_0}^{ab}$ is the sum, along the path of connection (ab) from a to i_0, of all transmission, propagation and processing delays (the subscript ai_0 means that $\tau_{ai_0}^{ab}$ is the delay from a to i_0 whereas the superscript ab refers to the path over which this delay has to be evaluated). Throughout this paper, $\hat{\tau}$ denotes the normalized (i.e., measured in time slots T) value of the delay τ.

Since the network time is slotted, when $\tau_{ai_0}^{ab}$ is not an integer multiple of T, (5) can be rewritten as follows (see Fig. 1):

$$f_{i_0}(n) = \sum_{(ab) \in C(i_0)} \left\{\left(1 - \theta_{ai_0}^{ab}\right) r_{ab}\left(n - d_{ai_0}^{ab}\right) + \theta_{ai_0}^{ab}\, r_{ab}\left(n - d_{ai_0}^{ab} + 1\right)\right\}, \quad (6)$$

where

$$d_{ai_0}^{ab} = \lceil \hat{\tau}_{ai_0}^{ab} \rceil : \text{the smallest integer} \geq \hat{\tau}_{ai_0}^{ab},$$
$$\theta_{ai_0}^{ab} = \lceil \hat{\tau}_{ai_0}^{ab} \rceil - \hat{\tau}_{ai_0}^{ab}, \theta_{ai_0}^{ab} \in [0, 1). \quad (7)$$

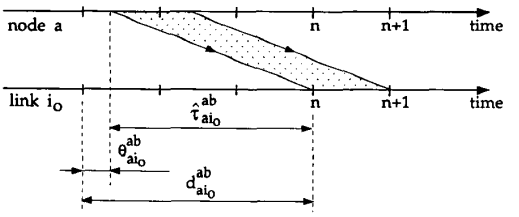


Fig. 1. Network delay: an illustration of the fractions $d^{ab}_{ai_0}$ and $\theta^{ab}_{ai_0}$ that comprise the delay $\hat{\tau}^{ab}_{ai_0}$.

Note that $d^{ab}_{ai_0}$ and $\theta^{ab}_{ai_0}$ are physical parameters of the network. Along one hop (i.e., through one link) we have

$$\hat{\tau} = \frac{\tau_s + \tau_p + \tau_{pr}}{T} = \frac{\tau_s + \tau_p + \tau_{pr}}{c\tau_s} = \frac{1}{c} + \frac{\tau_p + \tau_{pr}}{c\tau_s}. \quad (8)$$

Therefore, as the transmission capacity increases (τ_s decreases), the delay $d \triangleq \lceil \hat{\tau} \rceil$ increases and results in an increase in the dimension of the closed loop equations (see Section III).

The feedback delay of connection ab with respect to link i_0, $\tau^{ab}_{i_0 a}$, is defined in a way similar to the forward delay $\tau^{ab}_{ai_0}$. In fact, due to assumption (x), they are equal since the feedback path is the reverse of the forward path. Therefore,

$$\tau^{ab}_{ai_0} = \tau^{ab}_{i_0 a} = \frac{\tau^{ab}}{2}, \quad (9)$$

where $\tau^{ab} \triangleq \tau^{ab}_{ai_0} + \tau^{ab}_{i_0 a}$ is the round trip delay of connection ab with respect to link i_0.

Substituting (6) in (3), we obtain the buffer equation for the overloaded link:

$$x_{i_0}(n+1) = \mathrm{Sat}_X \Bigg\{ x_{i_0}(n) + \sum_{(ab)\in C(i_0)} \Big[(1 - \theta^{ab}_{ai_0}) \cdot r_{ab}(n - d^{ab}_{ai_0}) + \theta^{ab}_{ai_0}\, r_{ab}(n - d^{ab}_{ai_0} + 1) \Big] - c \Bigg\}. \quad (10)$$

2.3. Controller

2.3.1. Idea of the Control Law: The idea of the congestion controller employed in this work can be illustrated as follows.

Consider a single-node network with a traffic source of rate r and an outgoing link with capacity c_s packets/s, where $r > c_s$. A congestion controller is supposed to throttle the offered traffic so that the steady state buffer occupancy at the node is a given number x^0. The admission rate $q(t) < r$, defines the part of the offered traffic that will be admitted to the network. Assuming, for simplicity, that all variables are continuous, the dynamics of the buffer occupancy, $x(t)$, can be described as follows:

$$\dot{x} = q - c_s.$$

If a proportional control law is used, the admission rate obeys the equation

$$\dot{q} = -\alpha_0 (x - x^0).$$

This results in

$$\ddot{x} + \alpha_0 x = \alpha_0 x^0,$$

and both the buffer occupancy, $x(t)$, and the admitted traffic, $q(t)$, exhibit nondecaying oscillations (if $\alpha_0 > 0$; otherwise the system is unstable).

To eliminate this problem, a proportional-plus-derivative (PD) control law can be utilized:

$$\dot{q} = -\alpha_0 (x - x^0) - \alpha_1 \dot{x}. \quad (11)$$

Then,

$$\ddot{x} + \alpha_1 \dot{x} + \alpha_0 x = \alpha_0 x^0, \quad (12)$$

and $x(t)$ and $q(t)$ converge to x^0 and c_s, respectively (if α_0 and $\alpha_1 > 0$). Moreover, the speed of the convergence can be assigned arbitrarily by an appropriate choice of α_0 and α_1.

In networks with multiple switching nodes, however, PD controllers will not work either. The reason is that, due to delays in information transmission between the nodes, the buffer occupancy at each node is described by a time-delay equation of the form

$$\dot{x} = q(t - \tau) - c_s,$$

where $\tau > 0$. In this case, the closed loop system may be oscillatory even if the PD controller (11) is employed. The main idea of this work is to combat this difficulty by constructing a controller of the form

$$\dot{q}(t - \tau) = -\alpha_0 (x(t) - x^0) - \alpha_1 \dot{x}(t).$$

In this case, the closed loop behavior is still described by (12), and any desired dynamics can be achieved. The question is whether such a controller can be constructed for various types of input traffic patterns that result in different τ's. It turns out, however, that it is, indeed, possible, and the equations for such a controller are given in the next section. It will be shown that, as the input traffic pattern changes, the network dynamics itself changes. The congestion controller has to account for the changes in the dynamics by either being robust or adaptive to these changes. Both of these approaches (robust and adaptive designs) are discussed in this paper.

2.3.2. Controller Equations: The congestion controller includes two parts: the control algorithm and the control protocol. The control algorithm determines the admission rates and the control protocol specifies how the traffic sources react to these admission rates.

The following control protocol is used in this work: Assume that $q_i(n+1), i \in \mathcal{N}$, is the admission rate calculated at time n by the node for which link i is an outgoing link. Then, during the time slot $[n, n+1)$, the (ab)-type traffic is admitted with the rate

$$r_{ab}(n) = \min\{ q_i(n + 1 - \hat{\tau}^{ab}_{ia}) \text{ for } i \in p(ab),\ r^0_{ab} \}, (ab) \in \mathcal{C}, \quad (13)$$

where $\hat{\tau}_{ia}^{ab}$ is equal to τ_{ia}^{ab}/T and τ_{ia}^{ab}, defined in (9), is the delay incurred by the feedback signal q_i to reach node a. In compliance with this expression, the source transmission rate is defined by the smallest among all admission rates corresponding to the links along the path $p(ab)$ and the desired transmission rate. Note that this expression does not imply that the controller requires the knowledge of the offered traffic rate r_{ab}^0. Indeed, (13) means only that the rate of the admitted traffic will be either the minimum of q_i along the path of the connection or the demand rate r_{ab}^0, whichever is smaller. Since q_i is generated every T, the most recent feedback signal available at node a at time n is $q_i(n+1-d_{ia}^{ab})$ where $d_{ia}^{ab} = \lceil \hat{\tau}_{ia}^{ab} \rceil$. Therefore, (13) becomes

$$r_{ab}(n) = \min\{q_i(n+1-d_{ia}^{ab}) \text{ for } i \in p(ab),\ r_{ab}^0\},\ (ab) \in \mathcal{C}. \tag{14}$$

When the minimum in (14) corresponds to the admission rate of link $j \in p(ab)$, connection (ab) is said to be *throttled* by link j.

The control algorithm is defined by the following equation:

$$q_i(n+1) = \mathrm{Sat}_{q^0}\left\{ q_i(n) - \sum_{j=0}^{J} \alpha_j \left(x_i(n-j) - x^0\right) - \sum_{k=0}^{K} \beta_k\, q_i(n-k) \right\},\ i \in \mathcal{N}, \tag{15}$$

where $q^0 \geq c$, and J and K are nonnegative integers. The saturation function, defined in (2), is introduced to impose bounds on q_i's. The lower bound zero keeps q_i positive whereas the upper bound q^0 limits the sending rate of connections with an underloaded path.

When $J = 1$ and $K = 0$, (15) results in the PD controller

$$q_i(n+1) = \mathrm{Sat}_{q^0}\{q_i(n) - a\,(x_i(n) - x^0) - b\,(x_i(n) - x_i(n-1))\},$$

where $a = \alpha_0 + \alpha_1$ and $b = -\alpha_1$.

It is shown in Section III that the control gains, α_j and β_k in (15), must satisfy, among others, the following requirements:

$$\sum_{j=0}^{J} \alpha_j > 0\,,\ \sum_{k=0}^{K} \beta_k = 0. \tag{16}$$

For $i \neq i_0$, since, according to (4), $x_i(n) = 0, n \geq 0$, solving (15) under conditions (16) we obtain $q_i(n) = q^0,\ n \geq 0$ (assuming $q_i(n) = q^0,\ n \leq 0,\ i \neq i_0$). Therefore, underloaded links have their buffer at zero and their control signal (admission rate) equal to q^0. This implies that, for all connections $(ab) \in C(i_0)$, (14) becomes

$$r_{ab}(n) = \min\{q_{i_0}(n+1-d_{i_0a}^{ab}), r_{ab}^0\},\ (ab) \in C(i_0). \tag{17}$$

It follows from (15) that the control algorithm is defined by $J + K + 2$ control gains α's and β's. In the following two sections, the analysis of the closed loop behavior of (10), (15), and (17) for given gains α's and β's is described and a method for choosing these gains to ensure good dynamic characteristics is given.

III. Analysis

3.1. Closed Loop Equations

It follows from the previous section that the closed loop behavior of the network with i_0 as the single overloaded link is described by the following equations:

$$x_{i_0}(n+1) = \mathrm{Sat}_X\left\{ x_{i_0}(n) + \sum_{(ab)\in C(i_0)} \left[(1-\theta_{ai_0}^{ab}) \cdot r_{ab}(n - d_{ai_0}^{ab}) + \theta_{ai_0}^{ab} r_{ab}(n - d_{ai_0}^{ab} + 1)\right] - c \right\}, \tag{18}$$

$$r_{ab}(n) = \min\left\{q_{i_0}(n+1-d_{i_0a}^{ab}), r_{ab}^0\right\},\ (ab) \in C(i_0), \tag{19}$$

$$q_{i_0}(n+1) = \mathrm{Sat}_{q^0}\left\{ q_{i_0}(n) - \sum_{j=0}^{J} \alpha_j \left(x_{i_0}(n-j) - x^0\right) - \sum_{k=0}^{K} \beta_k\, q_{i_0}(n-k) \right\}. \tag{20}$$

Equation (19) can be rewritten as follows:

$$r_{ab}(n) = \delta_{ab}(n)\, q_{i_0}(n+1-d_{i_0a}^{ab}) + (1-\delta_{ab}(n))\, r_{ab}^0\,, \tag{21}$$

where

$$\delta_{ab}(n) \triangleq \begin{cases} 1 & \text{if } r_{ab}^0 > q_{i_0}(n+1-d_{i_0a}^{ab}), \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

Combining (18) and (21), we obtain

$$x_{i_0}(n+1) = \mathrm{Sat}_X\left\{ x_{i_0}(n) + \sum_{(ab)\in C(i_0)} \left[\delta_{ab}(n - d_{ai_0}^{ab}) \cdot (1-\theta_{ai_0}^{ab})\, q_{i_0}(n+1-d^{ab}) + \delta_{ab}(n - d_{ai_0}^{ab} + 1) \cdot \theta_{ai_0}^{ab} q_{i_0}(n+1-(d^{ab}-1))\right] + r_{i_0}^0(n) - c \right\}, \tag{23}$$

where

$$r_{i_0}^0(n) \triangleq \sum_{(ab)\in C(i_0)} \left[(1-\delta_{ab}(n-d_{ai_0}^{ab}))\,(1-\theta_{ai_0}^{ab})\, r_{ab}^0 + (1-\delta_{ab}(n-d_{ai_0}^{ab}+1))\,\theta_{ai_0}^{ab}\, r_{ab}^0\right],$$

$$d^{ab} \triangleq d_{ai_0}^{ab} + d_{i_0a}^{ab}\,. \tag{24}$$

When $\delta_{ab}(n) = 1$, the (ab) traffic is being throttled during $[n, n+1)$, otherwise (when $\delta_{ab}(n) = 0$) it is admitted with its desired transmission rate and $r_{i_0}^0$ is the cumulative rate of all nonthrottled flows. We will refer to d^{ab} as the (normalized) round trip delay (with respect to link i_0) of connection (ab). From (9), it is given by

$$d^{ab} = 2\left\lceil \frac{\hat{\tau}^{ab}}{2} \right\rceil = 2\left\lceil \frac{\tau^{ab}}{2T} \right\rceil. \tag{25}$$

Equation (23) can be rewritten as

$$x_{i_0}(n+1) = \mathrm{Sat}_X\left\{x_{i_0}(n) + \sum_{i=0}^{D} l_i(n)\, q_{i_0}(n+1-i) + r^0_{i_0}(n) - c\right\}, \quad (26)$$

where

$$l_i(n) \triangleq \sum_{(ab)\in C^i} \delta_{ab}(n - d^{ab}_{ai_0})(1 - \theta^{ab}_{ai_0}) + \sum_{(ab)\in C^{i+1}} \delta_{ab}(n - d^{ab}_{ai_0} + 1)\theta^{ab}_{ai_0}, \quad (27)$$

$$D \triangleq \max_{(ab)\in C(i_0)} d^{ab}, \quad (28)$$

and

$$C^d \triangleq \{(ab) \in C(i_0) : d^{ab} = d\}. \quad (29)$$

As defined above, C^d is the set of all connections flowing through link i_0 with round trip delay equal to d, and D is the largest round trip delay and is given by

$$D = 2\left\lceil \frac{\tau}{2T} \right\rceil, \quad (30)$$

where τ is the largest round trip delay of all connections in $C(i_0)$. The l_i's can be interpreted as the number of throttled flows with round trip delay equal to i time slots.

Omitting the index i_0, the closed loop equations become

$$x(n+1) = \mathrm{Sat}_X\left\{x(n) + \sum_{i=0}^{D} l_i(n) q(n+1-i) + r^0(n) - c\right\}, \quad (31)$$

$$q(n+1) = \mathrm{Sat}_{q^0}\left\{q(n) - \sum_{j=0}^{J} \alpha_j\big(x(n-j) - x^0\big) - \sum_{k=0}^{K} \beta_k q(n-k)\right\}. \quad (32)$$

Note that since (31) involves delayed versions of $q(n)$ up to $q(n-(D-1))$, it is natural to assume that $K \geq D-1$. Equations (31) and (32) describe a $(J+K+2)$-dimensional system with the state

$$Y(n) = \big(x(n) - x^0, x(n-1) - x^0, \ldots, x(n-J) - x^0, q(n), q(n-1), \ldots, q(n-K)\big)^T.$$

The steady states and dynamic properties of this system are described below.

3.2. Steady States

Let x_s and q_s be the steady state values corresponding to (31) and (32) under the assumption that the input traffic is constant. Then,

$$x_s = \mathrm{Sat}_X\left\{x_s + \left(\sum_{i=0}^{D} l_i\right) q_s + r^0 - c\right\}, \quad (33)$$

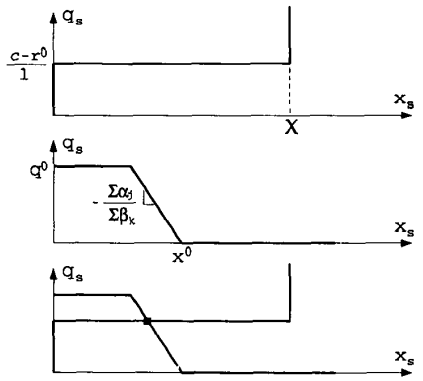


Fig. 2. Steady state: (a) and (b) are illustrations of the solutions of (33) and (34), respectively; (c) is the unique solution of (33) and (34) simultaneously.

$$q_s = \mathrm{Sat}_{q^0}\left\{\left(1 - \sum_{k=0}^{K} \beta_k\right) q_s - \left(\sum_{j=0}^{J} \alpha_j\right)(x_s - x^0)\right\} \quad (34)$$

where l_i and r^0 are the steady state values of $l_i(n)$ and $r^0(n)$, respectively. Note that $l \triangleq \sum_{i=0}^{D} l_i = \sum_{(ab)\in C(i_0)} \delta_{ab}$ is the total number of connections being throttled by link i_0 and is a positive integer since link i_0 is overloaded and therefore at least one connection is throttled.

Fig. 2(a) and (b) shows the solutions to (33) and (34), respectively. The combined solution is shown in Fig. 2(c) and corresponds to

$$q_s = \frac{c - r^0}{l}, \quad (35)$$
$$x_s = x^0 - \frac{\sum_{k=0}^{K} \beta_k}{\sum_{j=0}^{J} \alpha_j} q_s,$$

assuming that $q_s < q^0$ (the case $q_s = q^0$ happens only when $q^0 = c$ and one single connection is active). In order to ensure that $x_s = x^0$, we choose β_k's so that

$$\sum_{k=0}^{K} \beta_k = 0. \quad (36)$$

Thus, the steady state

$$Y_s = (0, \ldots, 0, q_s, \ldots, q_s)^T \quad (37)$$

exists if and only if the vector of control parameters

$$G = (\alpha_0, \alpha_1, \ldots, \alpha_J, \beta_0, \beta_1, \ldots, \beta_K)^T$$

satisfies constraint (36). Note that q_s in (35) can be rewritten as follows:

$$q_s = \frac{c - r^0}{l} = \frac{c}{M} + \frac{(M-l)\frac{c}{M} - r^0}{l}.$$

This expression presents the following fairness property: If M connections share a transmission link (i.e., M is the cardinality of $C(i_0)$), then each gets $1/M$ of the bandwidth and if $(M-l)$ connections use less than their share (i.e., only l connections are throttled), then the unused portion $((M-l)\frac{c}{M} - r^0)$ is equally distributed among the rest.

3.3. Stability Analysis

In general, the stability properties of the equilibrium point (37) could be investigated using either a global or a local approach. Unfortunately, due to the complex nature of the dynamics (18)–(20), we are unable to solve the global stability problem; we only investigated it by simulations described in Section V. Attempts at deriving global stability using various sufficient conditions for absolute stability were not successful. On the other hand, the (local) asymptotic stability problem can be addressed analytically and formal stability conditions could be derived. This section presents this development.

To study the asymptotic stability properties, we simplify the dynamic equations for $Y(n)$ in the neighborhood of Y_s by

a) removing the saturation nonlinearities in (31) and (32) since they are not activated for small deviations around Y_s ((x_s, q_s) is in the interior of $[0, X] \times [0, q^0]$);
b) treating $l_i(n)$ and $r^0(n)$ as constants since, based on (22), there exists a neighborhood of Y_s within which δ_{ab} is constant (assuming that $r^0_{ab} \neq q_s, (ab) \in C(i_0)$).

The resulting equations are

$$x(n+1) = x(n) + \sum_{i=0}^{D} l_i\, q(n+1-i) + r^0 - c, \quad (38)$$

$$q(n+1) = q(n) - \sum_{j=0}^{J} \alpha_j (x(n-j) - x^0) - \sum_{k=0}^{K} \beta_k\, q(n-k). \quad (39)$$

Theorem 3.1: For any given $l_0, l_1, \ldots, l_D$, the poles of the closed loop system (38) and (39) can be placed at will by an appropriate choice of the gains $\alpha_0, \ldots, \alpha_J, \beta_0, \ldots, \beta_K$, with $\sum_{k=0}^{K} \beta_k = 0$, if $J = 1$ and $K = D$. □

Proof: Due to space limitations, the proof of this and other theorems are omitted and can be found in [24].

In other words, a PD controller with respect to x and a controller of order D with respect to q are sufficient to ensure any desired dynamics of the closed loop system (38) and (39).

It is shown in the proof of the theorem that the characteristic polynomial of the closed loop system can be represented as

$$P(\lambda) = (-1)^{D+1} \lambda P_\Gamma(\lambda),$$

where

$$P_\Gamma(\lambda) = \lambda^{D+2} + \gamma_1 \lambda^{D+1} + \gamma_2 \lambda^D + \ldots + \gamma_{D+2}, \quad (40)$$

and the vector of coefficients $\Gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{D+2})^T$ is uniquely defined by the control gains

$$G = (\alpha_0, \alpha_1, \beta_0, \beta_1, \ldots, \beta_D)^T.$$

Therefore one of the closed loop poles is at 0 and the remaining $D+2$ poles can be chosen at will.

It is shown in the proof of Theorem 3.1 that for any desired set of coefficients γ_i, $i = 1, 2, \ldots, D+2$, the corresponding control gain G is

$$G = [M(L)]^{-1} \tilde{\Gamma}, \quad (41)$$

where

$$\tilde{\Gamma} = \begin{pmatrix} \gamma_1 + 2 \\ \gamma_2 - 1 \\ \gamma_3 \\ \gamma_4 \\ \vdots \\ \gamma_{D+1} \\ \gamma_{D+2} \\ 0 \end{pmatrix}, \quad L = \begin{pmatrix} l_0 \\ l_1 \\ \vdots \\ l_D \end{pmatrix},$$

$$M(L) = \begin{pmatrix} l_0 & 0 & 1 & & & & \\ l_1 & l_0 & -1 & 1 & & & \\ l_2 & l_1 & & -1 & \ddots & & \\ \vdots & \vdots & & & \ddots & 1 & \\ l_D & l_{D-1} & & & & -1 & 1 \\ 0 & l_D & & & & & -1 \\ 0 & 0 & 1 & 1 & \ldots & & 1 \end{pmatrix}.$$

It is also shown that expression (41) can be rewritten in the following alternative form:

$$\Gamma = \Gamma_\beta + W_\alpha L, \quad (42)$$

where

$$\Gamma_\beta = \begin{pmatrix} \beta_0 - 2 \\ \beta_1 - \beta_0 + 1 \\ \beta_2 - \beta_1 \\ \beta_3 - \beta_2 \\ \vdots \\ \beta_D - \beta_{D-1} \\ -\beta_D \end{pmatrix}, W_\alpha = \begin{pmatrix} \alpha_0 & & & \\ \alpha_1 & \alpha_0 & & \\ & \alpha_1 & \ddots & \\ & & \ddots & \alpha_0 \\ & & & \alpha_1 \end{pmatrix}.$$

Both of these expressions, (41) and (42), will be used in the following section for design purposes.

Note that the stability results described above are derived under the assumption of a constant input traffic rate. When the input changes, the steady state of the network changes as well. When changes are slow enough, as in assumption (xiii), the network has the time to reach a steady state before a new change take place. When changes are fast, the network state tracks the changes in the input.

Concluding this section, we note that the sum of the coefficients of $P_\Gamma(\lambda)$, $1 + \sum_{i=1}^{D+2} \gamma_i$, is equal to $(\alpha_0 + \alpha_1) \sum_{i=0}^{D} l_i$. Since it is necessary for asymptotic stability that this sum be positive, $\alpha_0 + \alpha_1$ has to be positive as stated in (16).

IV. Design

4.1. Adaptive versus Robust Design

The design of the congestion controller (15) involves the computation of the parameters of the control law, i.e., the vector of control gains $G = (\alpha_0, \alpha_1, \beta_0, \beta_1, \ldots, \beta_D)^T$. As has been shown in Section III, the closed loop dynamics depend on the number of throttled flows $L = (l_0, l_1, \ldots, l_D)^T$, where l_i is the number of throttled flows with round trip delay equal to i. Therefore, if each node can take into account the variations of L when computing the control law, the resulting design is

referred to as *adaptive*; here G is updated whenever L changes. If information about L is not available, a *robust* design has to be considered; here the control law is implemented with a fixed G so that stability and performance requirements are satisfied for all admissible values of L.

Note that an adaptive design has the advantage of achieving good performance but at the expense of higher computational requirements (the updating algorithm for G) whereas a robust design requires less computations but the achieved performance may be inferior. Each of these design approaches are investigated below.

4.2. Adaptive Design

Suppose that the desired performance of the congestion control system is specified in terms of closed loop poles that correspond to the vector of characteristic polynomial coefficients $\Gamma = (\gamma_1, \gamma_2, \ldots, \gamma_{D+2})^T$. Then, as follows from (41), the adaptation of the control gain $G = (\alpha_0, \alpha_1, \beta_0, \ldots, \beta_D)^T$ to changes in L takes the following form:

$$G(n) = [M(L(n-1))]^{-1}\tilde{\Gamma}, \tag{43}$$

where $L(n) = (l_0(n), l_1(n), \ldots, l_D(n))^T$ and $\tilde{\Gamma} = (\gamma_1 + 2, \gamma_2 - 1, \gamma_3, \ldots, \gamma_{D+2}, 0)^T$. Note that $G(n)$ depends on L at time $n-1$ since at time n only $L(n-1)$ is available.

In order to be able to implement an adaptive design, the congestion control protocol should be implemented in such a way that L can be known to the switching nodes. One way of doing so is to stamp each packet with an identifier of the throttling link, if any. In this way, every node can maintain a table of throttled flows. The performance of the network utilizing the adaptive control law (43) is illustrated in Section 5.3.

4.3. Robust Design

The goal of this section is to find a fixed gain $G = (\alpha_0, \alpha_1, \beta_0, \beta_1, \ldots, \beta_D)^T$ that ensures stability of the closed loop system (38) and (39) for all values of L belonging to an admissible set $\mathcal{L}$ defined as

$$\mathcal{L} = (\mathcal{L}_0 \times \mathcal{L}_1 \times \ldots \times \mathcal{L}_D) - \{\mathbf{0}\},\ \mathcal{L}_i = [0, \bar{l}_i],\ i = 0, 1, \ldots, D, \tag{44}$$

where $\bar{l}_i$ is the maximum number of throttled flows with round trip delay equal to i. From (27),

$$\bar{l}_i = \sum_{(ab)\in C^i} (1 - \theta^{ab}_{ai_0}) + \sum_{(ab)\in C^{i+1}} \theta^{ab}_{ai_0}, \tag{45}$$

where C^d is defined in (29).

The solution of this problem is given by the following two theorems:

Theorem 4.1: There exists a positive number $k^*(\bar{L})$ where $\bar{L} = (\bar{l}_0, \bar{l}_1, \ldots, \bar{l}_D)^T$ such that the closed loop system (38) and (39) is asymptotically stable for any $L \in \mathcal{L}$ if the control gains are chosen as

$$G = G_k \triangleq (k\alpha_0, k\alpha_1, \beta_0, \beta_1, \ldots, \beta_D)^T,\ k \in (0, k^*),$$

where

$$\alpha_0 = \frac{D+4}{2(D+1)},$$
$$\alpha_1 = -\frac{D+2}{2(D+1)},$$
$$\beta_0 = \frac{3D}{2(D+1)},$$
$$\beta_i = \frac{D-2(i+1)}{2(D+1)},\ i = 1, 2, \ldots, D.$$

□

Thus, the robust control gain, G_k, is the $(D+3)$-dimensional vector with components defined by the above theorem.

Let S_θ be the surface in $\mathbb{R}^{D+2}$ defined by the equations

$$S_\theta : \begin{cases} \sum_{i=1}^{D+2} \gamma_i \cos(i\theta) = -1, \\ \sum_{i=1}^{D+2} \gamma_i \sin(i\theta) = 0,\ \theta \in (0, \pi), \end{cases}$$

and let the inequality $L \le \bar{L}$ imply the componentwise inequalities $l_i \le \bar{l}_i$, $i = 0, 1, \ldots, D$.

Theorem 4.2: The positive number k^*, referred to in Theorem 4.1, is defined as

$$k^* = \min\{k_1, k_2\}.$$

Here,

$$k_1 = \frac{3D + 5 + (-1)^D(D+3)}{2(D+3)\sum_{i=0}^{\lfloor D/2 \rfloor} \bar{l}_{2i}},$$

and k_2 is the solution of the following optimization problem:

$$\begin{aligned} &\textit{minimize } k \textit{ subject to the constraints} \\ &\quad k \ge 0, \\ &\quad \Gamma_\beta + W_\alpha L \in S_\theta, \\ &\quad \mathbf{0} \le L \le k\bar{L}, \end{aligned} \tag{46}$$

where Γ_β and W_α are defined in (42). If the solution of this optimization problem does not exist, $k_2 = +\infty$. □

Thus, the design of the robust congestion controller can be accomplished in two stages. First, the optimization problem of Theorem 4.2 is solved and k^* is determined. At the second stage, a specific gain G_k is chosen from the family defined in Theorem 4.1 so that the performance requirements are satisfied as much as possible. This design procedure is illustrated in Section 5.2.

For each θ, the optimization problem of Theorem 4.2 is a linear programming (LP) problem. Indeed, we minimize a linear function, k, subject to the following linear equality and inequality constraints obtained by substituting in (46) the expressions of Γ_β, W_α and S_θ:

$$\begin{aligned} &k \ge 0, \\ &0 \le l_i \le k\bar{l}_i,\ i = 0, 1, \ldots, D, \\ &\textstyle\sum_{i=0}^{D} a_i(\theta) l_i = c_1(\theta), \\ &\textstyle\sum_{i=0}^{D} b_i(\theta) l_i = c_2(\theta), \end{aligned} \tag{47}$$

TABLE I
VALUES OF k^* FOR A RING NETWORK

D	1	2	3	4	5	6	7	8	9
k_1	.533	.286	.178	.121	.088	.067	.052	.042	.035
k_2	.640	.340	.196	.137	.089	.069	.050	.049	.029
k^*	.533	.286	.178	.121	.088	.067	.050	.042	.029

where

$$a_i(\theta) = \alpha_0 \cos((i+1)\theta) + \alpha_1 \cos((i+2)\theta),$$
$$b_i(\theta) = \alpha_0 \sin((i+1)\theta) + \alpha_1 \sin((i+2)\theta),$$
$$c_1(\theta) = -1 - 2\cos(\theta) + \cos(2\theta) + \sum_{i=0}^{D} \beta_i(\cos((i+1)\theta) - \cos((i+2)\theta)),$$
$$c_2(\theta) = -2\sin(\theta) + \sin(2\theta) + \sum_{i=0}^{D} \beta_i(\sin((i+1)\theta) - \sin((i+2)\theta)).$$

Therefore, if $k_2(\theta)$ denotes the result of the above minimization, the solution of the optimization problem (46) can be found as

$$k_2 = \inf_{\theta \in (0,\pi)} k_2(\theta).$$

Practically, an approximate solution can be found by discretizing the interval $(0, \pi)$ as $\theta_i = i\frac{\pi}{N+1}$, $i = 1, 2, \ldots, N$, solving the N resulting LP problems, and choosing the smallest of the N numbers. This procedure is illustrated below.

Consider a network with $2D+3$ nodes connected according to a bidirectional ring topology with the minimum-hop routing strategy. If the controller update period T is equal to the delay along one hop (transmission + propagation + processing), then $\bar{L}$ is given by

$$\bar{l}_{2i} = D + 1 - i, \ \bar{l}_{2i+1} = 0, \ \ i = 0, 1, \ldots, D.$$

Table I shows k_1, k_2 and k^* for different values of D. Note that k^* is a decreasing function of D since, as D increases, the network size increases and therefore the number of flows that might be active simultaneously increases and this requires a smaller gain in order to insure stability for any admissible vector L of throttled flows.

4.4. The Update Period

Both the robust and the adaptive design approaches require the selection of the update period T for the controller (see assumption (ix) in Section II). The value of T affects both the transient response and the control overhead due to the computation and transmission of the feedback information. In this section, we provide some insight into the major trade-offs involved in choosing the update period T.

Consider the transient due to the initiation of a connection through an initially overloaded link. The settling time and the buffer overshoot are the main characteristics of the transient response. The settling time, t_s, is the time interval between the start of transmission and the time when the transmission rate reaches and remains in the 5% neighborhood of its steady state value. Obviously, if τ is the round trip delay of the connection, we have

$$t_s \geq \tau + \kappa_1 T,$$

where $\kappa_1 > 0$ is a constant defined by the control law. If N is the average number of packets to be transmitted per connection and M is the average number of connections sharing the link, the ratio of settling time to transmission time of a connection, t_c, can be characterized as

$$\frac{t_s}{t_c} \geq \frac{\tau + \kappa_1 T}{NM\tau_s}. \tag{48}$$

The buffer overshoot Δx can be evaluated as follows:

$$\Delta x = x - x^0 \geq (\tau + \kappa_2 T)\, k\, c_s, \tag{49}$$

where x^0 is the desired buffer occupancy, $k\, c_s = k/\tau_s$ is the excess rate beyond the link capacity c_s, and $\kappa_2 > 0$ again depends on the control law . The inequalities in (48) and (49) are due to the fact that the control action cannot start before the round trip delay has elapsed and would require a number (κ_1 or κ_2) of update periods to reach either the steady state or the maximum buffer occupancy. From these inequalities we see that faster updates (i.e., smaller T) lead to shorter settling time and smaller buffer overshoot which are some of the desirable features for the network.

On the other hand, the decrease in T leads to an increase in the control overhead. Indeed, if τ_0 is the time necessary for the node to compute and transmit the control signals (feedback information) every update period T, then τ_0/T is the control overhead. The time τ_0 is shorter when the network speed is higher and therefore, it can be assumed proportional to τ_s ($\tau_0 = \kappa_3 \tau_s$).

Coefficient κ_3 can be viewed as a constant if (20) is implemented in hardware, or as a function of the update period T, if (20) is implemented in software. Indeed, the number of terms in (20) increases as T decreases since K is equal to the largest round trip delay D (Theorem 3.1). Therefore, as T decreases D increases and results in more terms being involved in (20). For a hardware implementation, the calculation of the last term in (20) does not require significantly different time for all reasonable values of K. Therefore, κ_3 can be viewed as a constant. On the other hand, for a software implementation, the increase in D has two implications: First, the time to compute a new control update becomes longer and, second, storage requirements increase. However, the increase in storage does not pose a problem since it is rather easy to equip the nodes with the required amount of memory.

Given the three performance criteria discussed above and assuming a hardware implementation of (20), the choice of the update period T can be approached as a multicriteria optimization problem and an appropriate Pareto set can be calculated. However, if we assume that there exist penalties ν_1, ν_2, and ν_3 for settling delay, queueing delay, and control overhead, respectively, then the following single function of T should be minimized in order to determine the compromise value of the controller update period:

$$J(T) = \nu_1 \frac{\tau + \kappa_1 T}{NM\tau_s} + \nu_2(\tau + \kappa_2 T)\frac{k}{\tau_s} + \nu_3 \frac{\kappa_3 \tau_s}{T}.$$

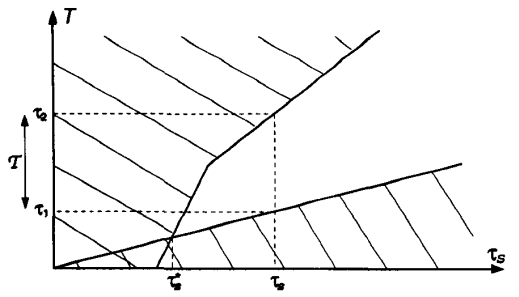


Fig. 3. Admissible set of update periods: the shaded region is excluded due to requirements (a)–(c).

The solution, obviously, is

$$T_0 = \tau_s \sqrt{\frac{\nu_3 \kappa_3}{\nu_1 \frac{\kappa_1}{NM} + \nu_2 \kappa_2 k}}.$$

This solution assumes that no constraints on the selection of T are imposed. Often it is the case, and the minimization of $J(T)$ should take place over some admissible set $\mathcal{T}$ resulting from the following requirements:

a) Ratio of settling time to transmission time of a connection less than some constant $\delta_1 < 1$:

$$\frac{\tau + \kappa_1 T}{NM\tau_s} \leq \delta_1. \tag{50}$$

b) Buffer overshoot less than δ_2:

$$(\tau + \kappa_2 T)\frac{k}{\tau_s} \leq \delta_2. \tag{51}$$

c) Control overhead less than $\delta_3 < 1$:

$$\frac{\kappa_3 \tau_s}{T} \leq \delta_3. \tag{52}$$

From constraints (50)–(52), we obtain the admissible set $\mathcal{T}$ as a function of τ_s as shown in Fig. 3. This figure is obtained under the assumption

$$\frac{\kappa_3}{\delta_3} < \min\left\{\frac{NM\delta_1}{\kappa_1}, \frac{\delta_2}{k\kappa_2}\right\}. \tag{53}$$

If (53) is not satisfied, the set $\mathcal{T}$ is empty, otherwise, $\mathcal{T}$ is the interval $[\tau_1, \tau_2]$ where

$$\tau_1 = \frac{\kappa_3}{\delta_3}\tau_s,$$

$$\tau_2 = \min\left\{\frac{NM\delta_1\tau_s - \tau}{\kappa_1}, \frac{\frac{\delta_2}{k}\tau_s - \tau}{\kappa_2}\right\}.$$

Let τ_s^* be the value of τ_s for which $\tau_1 = \tau_2$ (see Fig. 3). Then, if $\tau_s \geq \tau_s^*$, the constrained optimal update period is

$$T^* = \begin{cases} T_0, & \text{if } T_0 \in [\tau_1, \tau_2], \\ \tau_1, & \text{if } T_0 < \tau_1, \\ \tau_2, & \text{if } T_0 > \tau_2. \end{cases}$$

Section V below illustrates the effects of T on the transient behavior of the network using numerical simulations.

4.5. System Design Considerations

In addition to the update period discussed in the previous section, the other design issues to be addressed are the following:

4.5.1. The Parameter k: As discussed above, the robust stability problem consists of choosing k such that asymptotic stability is maintained when L takes any value in $\mathcal{L}$. In this case, the poles of the system will remain within the unit disk $\mathcal{D}$ for any $L \in \mathcal{L}$. The robust performance problem, however, consists of finding k such that some transient performance specified in terms of an acceptable range of settling time and percent overshoot is achieved for any $L \in \mathcal{L}$. This problem consists of keeping the poles of the system in some domain within the unit disc when L takes different values in $\mathcal{L}$. This domain $\mathcal{D}_0 \in \mathcal{D}$ corresponds to pole locations that meet the desired transient performance. Theorem 4.1 implies that the choice of any $k \in (0, k^*)$ is appropriate for robust stability. However, the robust performance problem may not admit a solution, i.e., there may not exist any k that ensures some given robust performance. More specifically, the more stringent the desired performance is, the lower the likelihood that it can be met.

To illustrate this point, let us consider the simple case of $D = 0$, i.e., when only local traffic traverses the link. In this case, l_0 denotes the number of throttled connections and satisfies $1 \leq l_0 \leq \bar{l}_0$. The vector of control gains for the case $D = 0$ is $Q_k = (k\alpha_0, k\alpha_1)$ with $\alpha_0 = 2$ and $\alpha_1 = -1$, and pole locations are determined by the product kl_0. It can be easily shown that stability is achieved if $0 < kl_0 < 4/3$ and, as a result, robust stability is guaranteed if $0 < k < (4/3)/\bar{l}_0$ (i.e., $k^* = (4/3)/\bar{l}_0$). It can also be shown that any achievable transient performance described by pole locations in a given domain $\mathcal{D}_0$ strictly included in $\mathcal{D}$ requires that $\delta_1 < kl_0 < \delta_2$ with $0 < \delta_1 < \delta_2 < 4/3$. Therefore, robust performance can be met only if $\delta_1 < k < \delta_2/\bar{l}_0$, which requires that $\bar{l}_0 < \delta_2/\delta_1$. Thus, if the specification of the transient performance (i.e., δ_1 and δ_2) is such that $\bar{l}_0 \geq \delta_2/\delta_1$, robust performance is not achievable. Moreover, the more stringent the performance specification is, which results in a smaller interval (δ_1, δ_2), the harder it is to achieve robust performance since δ_2/δ_1 gets closer to 1. Thus, whereas robust stability is always achievable, robust performance may not, and one has to rely on simulations to choose a good k in $(0, k^*)$ so that the transient performance satisfies some given specifications as much as possible.

4.5.2. The Parameter q^0: As mentioned above in connection with (15), parameter q^0 in (20) limits the sending rate of connections having all links in their path underloaded. Indeed, without such an upper bound, when link i_0 is underloaded (traffic demand less than the link capacity), $q_{i_0}(n)$ will be increasing. When a new connection starts while q_{i_0} has become too large, its input rate, determined by (19), will not be restricted enough which might lead to a long queue of outstanding packets. Therefore, q^0 should be chosen so that the initial buffer overshoot (i.e., when the link first becomes overloaded) will not be too large.

Another way of reducing this overshoot is to smooth out the traffic demand so that whenever a connection becomes

active, its input rate to the network increases gradually from zero until it reaches the demand level. This slow-start approach [27], while having the advantage of reducing the likelihood of large queues, has a drawback in that it tends to prolong the settling time and results in a slow convergence.

4.5.3. The Threshold x^0: The choice of the buffer threshold x^0 affects both the delay and the throughput performance of the network. Indeed, as it was shown above, when the capacity of a link is fully utilized, its steady state buffer occupancy is equal to x^0. Therefore, smaller x^0 results in smaller steady state queueing delay τ_q, since $\tau_q = hx^0\tau_s$, where h is the number of hops in the connection's path. However, in high speed networks, because of the large delay-bandwidth product, it is expected that the propagation delay will be the dominant component of the end-to-end delay.

On the other hand, smaller values of x^0 might lead to under-utilization of the link capacity which results in a decrease in the network throughput. This could happen during the transient periods of control when, for instance, one of the connections sharing a link is terminated. During the time when the controller is increasing the admission rate of the other connections in order to utilize the bandwidth left over by the terminated flow, the link might become idle if there is not enough buffered packets to keep it fully utilized. Therefore, x^0 has to be as small as possible but large enough to ensure full utilization of the network capacity during the transients.

4.5.4. Routing: We have assumed in (vi) a connection-oriented static routing where every connection has a single route fixed in advance. Dynamic connection-oriented routing allows rerouting of connections to avoid congested routes. Dynamic routing and congestion control interact with each other in the sense that, as the routing algorithm is more successful in evenly spreading the load through the network by rerouting traffic through less congested paths, the congestion control algorithm allows more traffic into the network. However, dynamic routing by itself cannot avoid congestion and control of sources' input rate is necessary when the total demand is larger than the network capacity.

The congestion control mechanism presented here can operate under dynamic routing as long as route changes are slower than the time constant of the congestion controller. In this case, when routing is updated, the new routing information is broadcast to all network nodes in order to update their controller parameters since they depend on the vector $\bar{L}$ and the dimension and entries of this vector are functions of the routing. This updating can be avoided by carrying out the design for worst case $\bar{L}$ to obtain a controller which is robust with respect to routing. Moreover, when the dynamic routing, satisfying the above slowly-varying requirement, is not connection-oriented but sessions within each connection are routed through different paths in order to further spread the load, the congestion controller works as well but the fairness properties of the algorithm are significantly altered and become very much dependent on routing.

4.5.5. Monitoring of the Transmission Rate: Every connection (ab) is admitted to the network with a rate r_{ab} determined by (14). This equation implies, implicitly, that the traffic sources are cooperative in the sense that they transmit with the rate r_{ab} and keep any excess traffic at the source. An alternative to the cooperative source assumption is to enforce the transmission rate, defined by (14), at the network interface using a policing device similar to the leaky bucket [28], [29] with a token generation rate equal to r_{ab} and a token buffer of size 1.

4.5.6. Delivery of Control Information: As stated in assumption (x), the control signals (admission rates q_i's) are sent to the traffic sources along the reverse direction of the traffic. One way of implementing this is to have, in every packet, a control field in which the source node of a link, say link i with node a as a destination node, inserts its control update q_i along with an identifier for the link. As this information is forwarded backward along the path of every potential connection, any intermediate node, say node a_1, that receives this information from its neighboring node a_2 strips this information and replaces it by the admission rate of link a_1a_2 if it is smaller. Node a_1 uses this result to monitor the locally generated traffic which is destined to node a. Finally, node a_1 inserts this result in the control field of the packet(s) ready to be transmitted along the reverse paths of connections having node a as the destination node. Thus, the control information is forwarded promptly and does not incur any queueing delay. Consequently, the feedback delay is the sum of the propagation delay along the links plus transmission and processing delays at intermediate nodes.

Note that for a given network and a given width assigned to the control field in each packet (which determines how many different q_i's can be inserted) there exists a lower bound on the update period T so that the rate with which the control information is generated does not exceed the capacity allocated for its transmission. This lower bound depends on the network topology and the routing, and can be evaluated using standard information-theoretic considerations.

V. Design Example And Simulations

This section presents a design example and simulation results. We start by describing the network and the input traffic. Then, we carry out both the robust and adaptive designs followed by an examination of the effects of the update period T. Finally, we illustrate the performance under randomness in the input traffic.

5.1. The Network and the Input Traffic

Consider a network where links have a delay-bandwidth product of 10 packets, i.e., the propagation delay over a link corresponds to the time for transmitting 10 packets. This is the case, for instance, of 45 Mbps channels when 1 kbyte packets are sent over a 340-mile link. Let τ_s be the transmission time of a packet and let c be the transmission capacity in packets/slot where the slot duration is equal to the update period T ($T = c\tau_s$).

For this design example, we assume that the network topology, and the input traffic, are such that the following properties are satisfied:

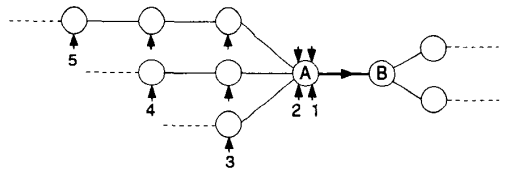


Fig. 4. Network topology: AB is the overloaded link.

TABLE II
ACTIVITY OF THE INPUT TRAFFIC

Connection #	1	2	3	4	5
# of hops	0	0	1	2	3
initiated at time	50	250	100	400	550
terminated at time	700	1000	850	1000	1000
input rate ($\times c$)	0.6	0.7	0.6	0.5	0.4

a) Only one link is overloaded.
b) The input traffic is such that

$$n_0 = 4,\ n_1 = 3,\ n_2 = 2,\ n_3 = 1,\ n_i = 0,\ i > 3,$$

where n_i is the maximum number of connections that can flow through the overloaded link with the source node located i hops from this link.

These properties are satisfied, for instance, by the network and the traffic shown in Fig. 4 and Table II, respectively.

Fig. 4 shows only a subset of the network nodes along with the sources of the five connections labeled according to the first column of Table II. This table describes the activity of these sources during the 1000 time slots of the experiment where the slot duration, T_1, is defined below. The presence of the other five unlabeled connections as potential traffic sources is taken into account in the design of the controller. The destination nodes, which are not shown in Fig. 4, can be any of the nodes downstream of node A. Note that the input traffic becomes heavy, i.e., larger than the link capacity, starting from time $t = 100T_1$.

5.2. Robust Design

In this section, we first carry out the robust design for a fixed update period T_1 (Section 5.2.1). Then, we investigate the effect of reducing the update period (Section 5.2.2). Finally, in Section 5.2.3, we evaluate the effect of the update period on the transient response.

5.2.1. Initial Design: It follows from Fig. 4 that the largest round trip propagation delay, RTD, corresponds to six hops: three hops in each of the forward and reverse paths of connection 5. Since the delay bandwidth product is assumed to be 10 and the transmission time of a packet is τ_s, this implies that $RTD = 6 \times 10\tau_s = 60\tau_s$. We choose the update period to be equal to the largest round trip propagation delay, $T_1 = 60\tau_s$, i.e., $c = 60$, and choose $x^0 = 30$ packets and $q^0 = c$.

With a slight abuse of notation, let $\hat{\tau}_{iA}$ denote the delay, measured in time slots T_1, that connections originating i hops from node A incur until they arrive at node A and let d_i be the corresponding round trip delay. Then, neglecting the processing delay, and using (7), (8), and (25) we have, respectively,

$$\begin{cases} \hat{\tau}_{0A} = 0 \\ \hat{\tau}_{1A} = \frac{\tau_s + \tau_p}{c\tau_s} = \frac{11\tau_s}{60\tau_s} = \frac{11}{60} \\ \hat{\tau}_{2A} = 2\hat{\tau}_{1A} = \frac{22}{60} \\ \hat{\tau}_{3A} = 3\hat{\tau}_{1A} = \frac{33}{60} \end{cases} \Rightarrow$$

$$\begin{cases} d_{0A} = 0,\ \theta_{0A} = 0 \\ d_{1A} = \lceil \frac{11}{60} \rceil = 1,\ \theta_{1A} = 1 - \frac{11}{60} = \frac{49}{60} \\ d_{2A} = \lceil \frac{22}{60} \rceil = 1,\ \theta_{2A} = 1 - \frac{22}{60} = \frac{38}{60} \\ d_{3A} = \lceil \frac{33}{60} \rceil = 1,\ \theta_{3A} = 1 - \frac{33}{60} = \frac{27}{60} \end{cases} \Rightarrow \begin{cases} d_0 = 0, \\ d_1 = 2, \\ d_2 = 2, \\ d_3 = 2. \end{cases}$$

Thus, as defined by (28), D is equal to 2. Therefore,

$$G_k = (k\alpha_0, k\alpha_1, \beta_0, \beta_1, \beta_2)^T,\ k \in (0, k^*), \qquad (54)$$

where, as follows from Theorem 4.1,

$$\alpha_0 = \frac{D+4}{2(D+1)} = 1,$$

$$\alpha_1 = -\frac{D+2}{2(D+1)} = -2/3,$$

$$\beta_0 = \frac{3D}{2(D+1)} = 1,$$

$$\beta_1 = \frac{D-4}{2(D+1)} = -1/3,$$

$$\beta_2 = \frac{D-6}{2(D+1)} = -2/3,$$

and

$$\bar{L} = (\bar{l}_0, \bar{l}_1, \bar{l}_2)^T,$$

where, as it follows from (45),

$$\bar{l}_0 = n_0(1 - \theta_{0A}) = 4,$$
$$\bar{l}_1 = n_1\theta_{1A} + n_2\theta_{2A} + n_3\theta_{3A} = 4.17,$$
$$\bar{l}_2 = n_1(1 - \theta_{1A}) + n_2(1 - \theta_{2A}) + n_3(1 - \theta_{3A}) = 1.83.$$

With this $\bar{L}$, the solution of the optimization problem of Theorem 4.2 is $k^* = 0.156$.

The behavior of the network with G_k defined by (54) for $k = 0.15$ and $k = 0.075$ is shown in Fig. 5(a) and (b), respectively. These figures were obtained by a numerical solution of (18)–(20) with corresponding G_k's. The results can be summarized as follows:

a) Smaller values of k result in a slower response and larger buffer overshoot. We show in the next subsection that faster response and smaller buffer overshoot can be achieved by adopting smaller update periods.
b) When the traffic is heavy (starting at time $t = 100$), the buffer is kept at $x^0 = 30$, with fluctuations whenever a connection is initiated (at t =100, 250, 400, and 550) or terminated (at $t = 700$ and 850).
c) When the input traffic changes, the rate $q(n)$ converges to a new steady state where the link bandwidth is reallocated in order to accommodate all incoming traffic in a fair way.

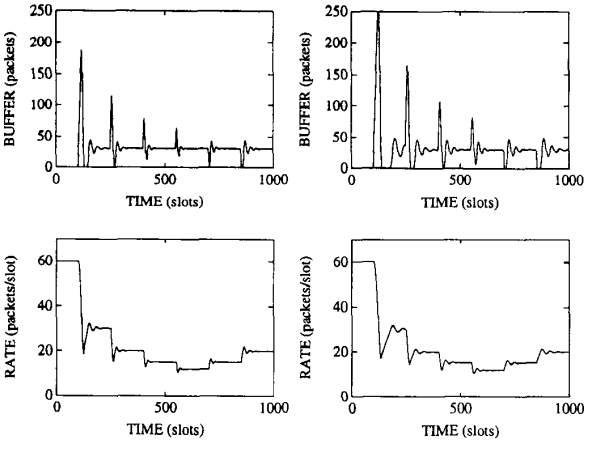


Fig. 5. Performance under robust design with $T_1 = 60\tau_s$: (a) $k = 0.15$; (b) $k = 0.075$.

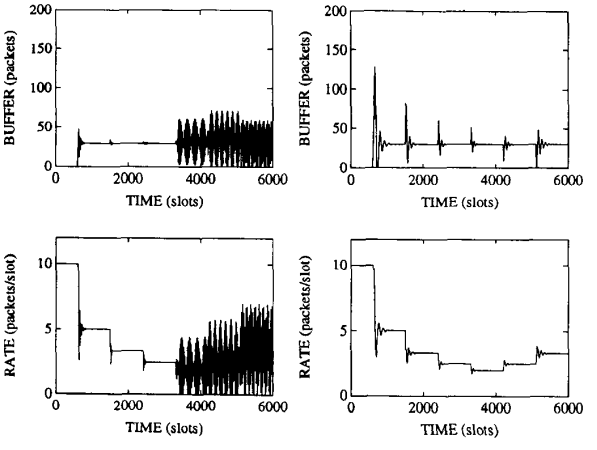


Fig. 6. Performance under robust design with $T_2 = 10\tau_s$: (a) when controller of Fig. 5(a) is used, (b) when appropriate gain (55) is used.

5.2.2. Effect of the Update Period T: Controller (54) was derived for the update period $T_1 = 60\tau_s$. It follows from Fig. 5 that this T results in quite large overshoot and slow response. It might seem reasonable to decrease T in order to eliminate these problems. According to the theory developed in Sections III and IV, this, however, is not the case: the utilization of a controller designed for a larger T in a network with smaller T may bring about an instability. This effect is illustrated in Fig. 6(a) where G_k of (54) with $k = 0.15$ is used when T is chosen to be $T_2 = 10\tau_s$ instead of $60\tau_s$. Obviously, the oscillatory behavior observed is unacceptable in most applications.

The way to improve the speed of the response is not only to reduce T but also to redesign the controller appropriately (Theorems 3.1, 4.1, and 4.2). Specifically, for $T_2 = 10\tau_s$, repeating the design steps described above, we obtain

$$\begin{cases} \hat{\tau}_{0A} = 0 \\ \hat{\tau}_{1A} = \frac{\tau_s+\tau_p}{c\tau_s} = 1.1 \\ \hat{\tau}_{2A} = 2\hat{\tau}_{1A} = 2.2 \\ \hat{\tau}_{3A} = 3\hat{\tau}_{1A} = 3.3 \end{cases} \Rightarrow$$

$$\begin{cases} d_{0A} = 0,\ \theta_{0A} = 0 \\ d_{1A} = 2,\ \theta_{1A} = 0.9 \\ d_{2A} = 3,\ \theta_{2A} = 0.8 \\ d_{3A} = 4,\ \theta_{3A} = 0.7 \end{cases} \Rightarrow \begin{cases} d_0 = 0, \\ d_1 = 4, \\ d_2 = 6, \\ d_3 = 8. \end{cases}$$

Thus, the maximum delay D is equal to 8 and therefore

$$G_k = (k\alpha_0, k\alpha_1, \beta_0, \beta_1, \ldots, \beta_8)^T,\ k \in (0, k^*), \qquad (55)$$

where,

$$\alpha_0 = 2/3,\ \alpha_1 = -5/9,$$
$$\beta_0 = 4/3,\ \beta_1 = 2/9,\ \beta_2 = 1/9,$$
$$\beta_3 = 0,\ \beta_4 = -1/9,\ \beta_5 = -2/9,$$
$$\beta_6 = -1/3,\ \beta_7 = -4/9,\ \beta_8 = -5/9,$$

and

$$\bar{L} = (\bar{l}_0, \bar{l}_1, \ldots, \bar{l}_8)^T,$$

where,

$$\bar{l}_0 = n_0(1-\theta_{0A}) = 4,\ \bar{l}_1 = 0,\ \bar{l}_2 = 0,$$
$$\bar{l}_3 = n_1\theta_{1A} = 2.7,\ \bar{l}_4 = n_1(1-\theta_{1A}) = 0.3,$$
$$\bar{l}_5 = n_2\theta_{2A} = 1.6,\ \bar{l}_6 = n_2(1-\theta_{2A}) = 0.4,$$
$$\bar{l}_7 = n_3\theta_{3A} = 0.7,\ \bar{l}_8 = n_3(1-\theta_{3A}) = 0.3.$$

The solution of the optimization problem gives $k^* = 0.160$.

Equations (18)–(20) with $T = 10\tau_s$ and G_k given in (55) with $k = 0.15$ have been solved numerically. The results, shown in Fig. 6(b), lead to the following conclusions:

a) Adopting a smaller update period T, along with the appropriate controller, results in a stable operation of the network with faster response and smaller buffer overshoot.
b) When the network operates with a controller designed for larger T, an instability could be brought about (see Fig. 6(a) where large oscillations take place when connection 5 is active).

Note that the total simulation time ($6000T_2$) is the same as in Fig. 5 ($1000T_1$) since $T_2 = T_1/6$.

5.2.3. Evaluation of the Transient Response: In order to evaluate the improvement in transient response due to the reduction of the update period from T_1 to $T_2 = T_1/6$, we examine the first four transients in Figs. 5(a) and 6(b) corresponding to connection initiations (the last two transients correspond to connection terminations and exhibit buffer undershoots rather than overshoots). Table III shows the improvements in the buffer overshoot $\Delta x = x - x^0$ and the settling time (time it takes the buffer to reach and remain within the interval [28.5, 31.5]) due to the decrease in T. This improvement, which is around 40%, clearly is not linear in T (reduced by 83%). Moreover, there are fundamental bounds on the achievable transient response due to the presence of a fixed propagation delay. Indeed, the fastest response that can be achieved is defined by the so-called dead-beat control [30], where the steady state is reached in exactly n time slots and n is the order of the system. This response is obtained when the poles of the system (38) and (39) are located at the origin of the complex plane. Obviously, when the robust controller is used, pole locations change as the input traffic pattern changes and, therefore, the dead-beat response constitutes only a lower

TABLE III
IMPROVEMENT OF BUFFER OVERSHOOT (IN PACKETS) AND SETTLING TIME (IN UNITS OF T_1)

		$T_1 = 60\tau_s$	$T_2 = 10\tau_s$	% Imp.
First Transient	Buffer Overshoot	156	98	37
	Settling Time	153	91	40.5
Second Transient	Buffer Overshoot	84	52	38
	Settling Time	102	61	40.1
Third Transient	Buffer Overshoot	49	31	37.6
	Settling Time	71	41	42
Fourth Transient	Buffer Overshoot	34	21	38
	Settling Time	41	31	24

bound on the achievable response time. From (31) and (32), the system order is equal to $D+3$ and, if a dead-beat controller were used, the time to reach steady state would be

$$t_s = (D+3)T = \left(\left\lceil \frac{\tau}{2T} \right\rceil + 3\right)T \geq \left(\frac{\tau}{T} + 3\right)T = \tau + 3T,$$

where τ is the largest round trip delay. Therefore, t_s cannot be made smaller than τ regardless of how small the update period T is.

5.3. Adaptive Design

In this section we design an adaptive controller for both update periods considered above, T_1 and T_2.

In the case of T_1, the characteristic polynomial $P_\Gamma(\lambda)$ in (40) has dimension $D+2=4$ and therefore, the gains are specified by the choice of four desired closed loop poles. For instance, if we choose the desired characteristic polynomial as

$$P_\Gamma(\lambda) = \lambda^2(\lambda-\lambda_1)(\lambda-\lambda_2),\ \lambda_{1,2} = 0.4 \pm j0.3,$$

then, as it follows from (43), the adaptation of the controller gains

$$G = (\alpha_0, \alpha_1, \beta_0, \beta_1, \beta_2)^T$$

to changes in L takes the following form

$$G(n) = [M(L(n-1))]^{-1}\tilde{\Gamma}, \tag{56}$$

where

$$\tilde{\Gamma} = (1.2, -0.75, 0, 0, 0)^T.$$

In the case $T_2 = 10\tau_s$, we choose the desired behavior according to

$$P_\Gamma(\lambda) = \lambda^8(\lambda-\lambda_1)(\lambda-\lambda_2),\ \lambda_{1,2} = 0.4 \pm j0.3.$$

As a result, $G = (\alpha_0, \alpha_1, \beta_0, \beta_1, \ldots, \beta_8)^T$ is given by

$$G(n) = [M(L(n-1))]^{-1}\tilde{\Gamma}, \tag{57}$$

where

$$\tilde{\Gamma} = (1.2, -0.75, 0, 0, \ldots, 0)^T.$$

Networks with controllers (56) and (57) have been simulated by a numeric solution of (31) and (32). The results are shown in Fig. 7(a) and (b), respectively. From these figures we conclude:

a) The transient behavior of the network under adaptive control is much better than under robust control.

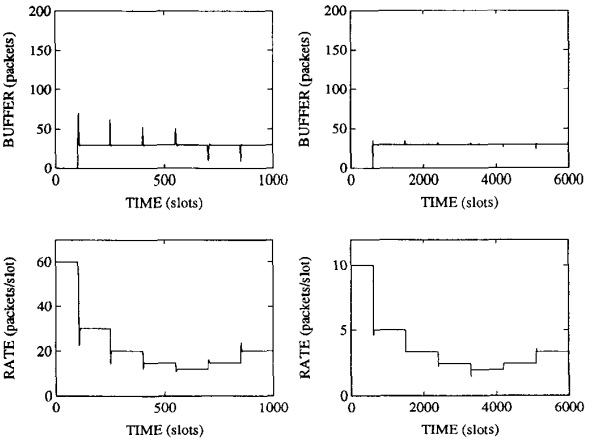


Fig. 7. Performance under adaptive design: (a) $T = T_1$; (b) $T = T_1/6$.

b) Higher updating rates result in faster response and smaller buffer overshoot.

c) The superior performance exhibited by the adaptive controller justifies its selection for implementation despite the extra computational time it requires.

5.4. Effect of Random Traffic

We have assumed throughout this paper that the input traffic is specified by a constant rate that may change from time to time in a random manner. In reality, however, the input traffic is random in the sense that the rate r^0_{ab} of connection ab is a random number $r^0_{ab}(n)$. In order to investigate the behavior of the control scheme developed in this paper, we assume that $r^0_{ab}(n)$, $n = 1, 2, \ldots$, is a sequence of independent random variables uniformly distributed over the range $[(1-\varepsilon)r^0_{ab}, (1+\varepsilon)r^0_{ab}]$. The behavior of the network with controller (55) and $k = 0.15$ is shown in Fig. 8(a), for $\varepsilon = 0.6$, and Fig. 8(b), for $\varepsilon = 0.3$. The results can be summarized as follows:

a) Due to the structure of the control protocol in (19), the randomness will not appear in the state trajectories if $r^0_{ab}(n)$ remains greater than $q(n)$ for all active connections ab. This is more likely to happen if more connections are being throttled since, as it follows from (35), the steady state q is inversely proportional to the number l of throttled flows. This is the reason why the plots in Fig. 8 exhibit more fluctuations when smaller number of connections are active.

b) During the time intervals when the randomness in the input is reflected in the dynamics, the state trajectories fluctuate but remain close to the deterministic trajectories shown in Fig. 6(b).

c) As the level of randomness increases, i.e., as ε gets larger, the level of fluctuations increases.

VI. CONCLUSIONS

This paper presents a theory for analysis and design of a congestion control system in packet switching networks. A case of a single congested node is considered. Within the framework of the controller structure and system architecture

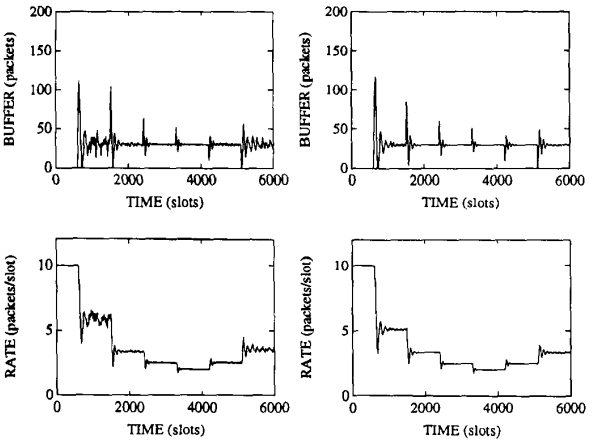


Fig. 8. Effect of random input: (a) level of randomness $\varepsilon = 0.6$; (b) $\varepsilon = 0.3$.

assumed (Section II), the design of the congestion controller is accomplished as follows:

1) Using Theorem 3.1, choose the order of the controller that guarantees the existence of stabilizing gains.
2) If the adaptive controller is used (i.e., the gains are adapted to the specific input traffic and network conditions), select the controller gains based on (56).
3) If the robust controller is used (i.e., the gain ensures stability for all admissible input traffics and network conditions), select the parameterized controller gain as indicated in Theorem 4.1 and a specific gain based on Theorem 4.2.

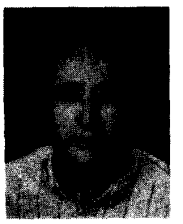
Both the adaptive and the robust designs, with the gains selected appropriately, ensure asymptotic stability of the network. Moreover, numerical simulations, reported in Section V, show that the adaptive controller is capable of insuring much higher performance characteristics as compared with the robust one. This may justify the additional computation burden necessary for the implementation of the adaptive approach.

ACKNOWLEDGMENT

The authors are grateful to the three anonymous reviewers for their thoughtful and constructive comments.

REFERENCES

[1] M. Gerla and L. Kleinrock, "Flow control: A comparative survey," *IEEE Trans. Communications*, vol. COM-28, pp. 553–574, 1980.
[2] M. Schwartz, *Telecommunication Networks, Protocols, Modeling and Analysis*. Reading, MA: Addison-Wesley, 1987.
[3] A. S. Tannenbaum, *Computer Networks*. Englewood Cliffs, NJ: Prentice Hall, 1988.
[4] D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice Hall, 1992.
[5] R. Jain and K. Ramakrishnan, "Congestion avoidance in computer networks with connectionless network layer," in *Proc. Comp. Net. Symp.*, Washington, DC, Apr. 1988, pp. 134–143.
[6] L. Zhang, "A New Architecture for Packet Switching Network Protocols," Ph. D. dissertation, MIT, Cambridge, MA, 1989.
[7] S. Shenker, "A theoretical analysis of feedback flow control," in *ACM SIGCOMM*, Philadelphia, PA, 1990, pp. 156–165.
[8] S. Keshav, "A control-theoretic approach to flow control," *ACM Comp. Comm. Review*, vol. 21, no. 4, pp. 3–15, 1991.
[9] G. Ramamurthy and R. S. Dighe, "A multidimensional framework for congestion control in B-ISDN," *IEEE J. Select. Areas Comm.*, vol. 9, pp. 1440–1451, 1991.
[10] V. Jacobson, "Congestion avoidance and control," in *Proc. ACM SIGCOMM*, Stanford, CA, 1988, pp. 314–329.
[11] K. K. Ramakrishnan and R. Jain, "A binary feedback scheme for congestion avoidance in computer networks with a connectionless network layer," in *Proc. ACM SIGCOMM*, Stanford, CA, 1988, pp. 303–313.
[12] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Proc. ACM SIGCOMM*, 1989, pp. 1–12.
[13] L. Zhang, "Some thoughts on the packet network architecture," *ACM Comp. Comm. Review*, pp. 3–17, 1987.
[14] J. Bolot and A. Shanker, "Dynamical behavior of rate-based flow control mechanism," *ACM Comp. Comm. Review*, vol. 20, no. 2, pp. 35–49, 1990.
[15] A. Mukherjee and J. Stikwerda, "Analysis of dynamic congestion control protocols—A fokker-plank approximation," in *Proc. ACM SIGCOMM*, Zurich, Switzerland, 1991, pp. 159–169.
[16] D. Mitra, "Asymptotically optimal design of congestion control for high speed data networks," *IEEE Trans. Communications*, vol. 40, pp. 301–311, 1992.
[17] D. Mitra and J. B. Seery, "Dynamic adaptive windows for high speed data networks: Theory and simulations," in *Proc. ACM SIGCOMM*, Philadelphia, PA, 1990, pp. 30–40,.
[18] D. Mitra and J. B. Seery, "Dynamic adaptive windows for high speed data networks with multiple paths and propagation delays," in *Proc. IEEE INFOCOM*, 1991, pp. 39–48.
[19] K. W. Fendick, M. A. Rodrigues, and A. Weiss, "Analysis of a rate-based feedback control strategy for long haul data transport," *Performance Evaluation*, vol. 16, pp. 67–84, 1992.
[20] P. Mishra and H. Kanakia, "A hop by hop rate-based congestion control scheme," in *Proc. ACM SIGCOMM*, Baltimore, MD, 1992, pp. 112–123.
[21] J. Robinson, D. Friedman, and M. Steenstrup, "Congestion control in BBN packet switched networks," *ACM Comp. Comm. Review*, vol. 20, no. 1, pp. 76–90, 1990.
[22] Z. Haas and J. H. Winters, "Congestion control by adaptive admission," in *Proc. IEEE INFOCOM*, Bal Harbour, FL, Apr. 1991, pp. 560–569.
[23] D. Chiu and R. Jain, "Analysis of the increase and decrease algorithms for congestion avoidance in computer network," *Comp. Net. and ISDN Syst.*, vol. 17, pp. 1–14, 1989.
[24] L. Benmohamed, S. M. Meerkov, "Feedback Control of Congestion in Packet Switching Networks: The Case of a Single Congested Node," Control Group Rep. CGR-93-15, Univ. of Michigan, Nov. 1993.
[25] S. M. Meerkov, "Simplified description of slow markov walks—Part I," *Automation Remote Contr.*, vol. 33, pp. 404–414, 1972.
[26] J.-T. Lim and S. M. Meerkov, "Simplified description of slow-in-the-average markov walks," *Math. Analysis and Appl.*, vol. 158, no. 2, pp. 476–486, 1991.
[27] Z. Wang and J. Crowcroft, "A new congestion control scheme: Slow start and search (Tri-S)," *ACM Comp. Comm. Review*, vol. 21, no. 1, pp. 32–43, 1991.
[28] J. Turner, "New directions in communications (or which way to the information age?)," *IEEE Comm. Magazine*, vol. 24, pp. 8–15, 1986.
[29] M. Sidi, W. Liu, I. Cidon, and I. Gopal, "Congestion control through input rate regulation," in *Proc. IEEE GLOBECOM*, Dallas, TX, 1989, pp. 1764–1768.
[30] K. J. Astrom and B. Wittenmark, *Computer-Controlled Systems*. Englewood Cliffs, NJ: Prentice-Hall, 1990.



Lotfi Benmohamed (S'91-M'94) received the Ph. D. degree in electrical engineering: systems from The University of Michigan in 1993.

At present, he is a research scientist with the Advanced Communications Group at the National Institute of Standards and Technology, where he does research on high speed networking. His current research interests include congestion control and performance evaluation of communication networks.

Semyon M. Meerkov (M'78-SM'83-F'90) received the M. S. E. E. degree from the Polytechnic of Kharkov, Ukraine, in 1962 and the Ph. D. degree in system science from the Institute of Control Sciences, Moscow, Russia, in 1966.

He was with the Institute of Control Sciences until 1977. From 1979 to 1984 he was with the Department of Electrical Engineering, Illinois Institute of Technology, Chicago, IL. Since 1984 he has been with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI, where he is a Professor. He has held visiting position at UCLA (1978–1979) and Stanford (1991). His research interests are in systems and control with applications to manufacturing and communications.