# Dynamic Control of Session Input Rates in Communication Networks

ELI M. GAFNI AND DIMITRI P. BERTSEKAS, FELLOW, IEEE

*Abstract*—**We consider a distributed iterative algorithm for dynamically adjusting the input rate of each session of a voice or data network using virtual circuits so as to exercise flow control. Each session origin periodically receives information regarding the level of congestion along the session path and iteratively corrects its input rate. In this paper we place emphasis on voice networks, but the ideas involved are also relevant for dynamic flow control in data networks.**

**The algorithm provides for the addition of new and termination of old sessions and maintains at all times feasibility of link flows with respect to capacity constraints. Fairness with respect to all sessions is built into the algorithm and a mechanism is provided to control link utilization and average delay per packet at any desired level.**

## I. Introduction

THE purpose of this paper is to propose and investigate a flow control algorithm for adjusting session rates in a data or voice network. This algorithm is motivated by a voice coder scheme introduced in [1] and called "embedded coding." In this scheme a segment of talkspurt is coded into packets of different priority levels. The higher priority packets contain the "core" of the speech while the lower priority packets contain the information that "fine tunes" it. Traditional voice flow control mechanisms either block the initiation of a call or discard small segments of it while it is already in progress. By contrast the embedded coding scheme dynamically trades off between voice quality and congestion by discarding the lower "priority" packets either at the point of congestion or the point of entry. The level of congestion at which the gaps between the segments, delivered by the traditional schemes, render the speech unintelligible is much lower than the one at which the embedded coding scheme delivers unintelligible information. This flexibility in exercising flow control makes the embedded coding scheme attractive.

Alleviation and prevention of congestion by discarding lower priority packets at the point of entry seems to be superior to discarding them at the point of congestion. The latter amounts to a waste of network resources. But, it would not be advisable to forgo the capability of discarding lower priority packets at the point of congestion because of the time delay involved in making the entry points aware of downstream congestion. Based on this we believe that both capabilities should be used. The rates at the entry points will be determined based on longer time averages of congestion levels while the capability of discarding packets at the point of congestion will serve to alleviate intolerable momentary bottlenecks. The rates at the entry point will be adjusted so that the capability of discarding packets at the point of congestion will not be exercised too often.

Manuscript received May 24, 1983; revised April 5, 1984. Paper recommended by D. P. Looze, Past Chairman of the Computational Methods and Discrete Systems Committee. This work was conducted at the M.I.T Laboratory for Information and Decision Systems and was supported in part by the Defense Advanced Research Projects Agency under Contract ONR/N00014-75-C-1183.

E. M. Gafni is with the Department of Computer Science, University of California, Los Angeles, CA 90024.

D. P. Bertsekas is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139.

In this paper we discuss a method of determining the input rates at the entry point. To this end we will ignore the capability of discarding packets within the network in order to simplify the analysis. Throughout the paper we implicitly assume that we are dealing with a virtual circuit network where communication between a user pair is established by creating a *session involving a path that remains fixed throughout the duration of the user pair conversation.* The method of choice of the session path (i.e., the routing algorithm) is not considered in this paper. We propose an "on-line" iterative algorithm that will solve a static problem. The hope is that the algorithm converges fast enough relative to the session initiation and termination process, and as a result will be able to "track" its variation keeping the rates in the ballpark of the optimal rates at all times.

The criterion used to determine input rates is based on the notion of "fair allocation" introduced in Section II. Roughly speaking, the objective is to maximize the smallest session rate, and once this is achieved to maximize the second smallest rate, etc. In Section III we introduce the algorithm, describe its convergence properties, and discuss implementation issues.

The idea of the algorithm is to adjust the input rates of sessions on the basis of the current level of congestion along the session path. The necessary information is collected by a control packet sent periodically by each session origin along the path similarly as in flow control methods investigated by simulation in [1]. This method of adjusting input rates seems also suitable for other situations where fast reaction to momentary congestion is needed. For example, when the number of users in the network is small but some of these users transmit in traffic intensive, intermittent, but relatively long bursts, then a dynamic method of flow control is needed. The ideas of this paper may provide an alternative to the usual end-to-end windowing schemes [11] or other techniques [7], [8] in such situations. Some research along this direction may be found in [12].

## II. Problem Formulation

Consider a network with nodes $1,2,\cdots,N$ and a set of directed links $\mathcal{L}$. Each link $a \in \mathcal{L}$ has a capacity $C_a$ associated with it—a positive number. Let $\mathcal{S}$ denote a set of sessions taking place between nodes. Each session $s \in \mathcal{S}$ has an origin node associated with it and traverses a subset of links denoted by $\mathcal{L}_s$. Note that we do not restrict the session to have a single destination, so the set of links $L_s$ may be for example a tree rooted at the origin node of $s$ and used for broadcasting messages throughout the network. We denote by $\mathcal{S}_a$ the set of sessions traversing a link $a \in \mathcal{L}$. If $r_s$ is the input rate of session $s$ (in data units/s), then the flow $F_a$ of a link $a \in \mathcal{L}$ is given by

$$F_a = \sum_{s \in \mathcal{S}_a} r_s. \tag{1}$$

The problem broadly stated is to choose a vector of session input rates $r = (\cdots, r_s, \cdots)$ which results in a set of "satisfactory" link flows $\{F_a \mid a \in \mathcal{L}\}$, and at the same time maintains a certain degree of "fairness" for all sessions.

It is customary to consider as one of the characteristics of a fair allocation of resources in a network the feature that it is indifferent to the geographical separation of the session's origin and destinations. Although there might be different priorities assigned to sessions, these priorities are not assigned on the basis of geographical distance. Moreover, two sessions of the same priority should obtain the same rate, if the rate of one can be traded for the rate of the other without overloading the network or reducing the rate of any other session. This is in the spirit of making the network "transparent" to the user.

To capture the notion of fairness and priority as presented above we define the notiön of fair allocation.

For a vector $x = (x^1, x^2, \cdots, x^n)$ in the Euclidean space $R^n$, we consider the vector $\bar{x} = (\bar{x}^1, \bar{x}^2, \cdots, \bar{x}^n)$ the coordinates of which are the same as those of $x$ but are rearranged in order of increasing value, i.e., we have $\bar{x}^1 \leqslant \bar{x}^2 \cdots \leqslant \bar{x}^n$ and with each $i = 1, \cdots, n$ we can associate a distinct $i'$ such that $\bar{x}^i = x^{i'}$. We call $\bar{x}$ the *increasing permutation* of $x$. Given a subset $X$ of $R^n$ we will say that a vector $x \in X$ is a *fair allocation* over $X$ if for every vector $y \in X$ the increasing permutation $\bar{x}$ of $x$ is lexicographically greater or equal to the increasing permutation $\bar{y}$ of $y$, i.e., if $\bar{y}^j > \bar{x}^j$ for some $j$, then there exists an $i<j$ such that $\bar{y}^i < \bar{x}^i$.

If we view $X$ as a "feasible" set, a fair allocation vector $x$ over $X$ solves a hierarchy of problems. The first problem is to maximize the minimal coordinate of vectors in $X$. The second problem is to maximize the second minimal coordinate over all vectors which solve the first problem, etc.

Hayden [4] proposed an algorithm which results in a rate vector $r = (\cdots, r_s, \cdots)$ which is a fair allocation over the set defined by

$$F_a \leqslant \rho C_a, \qquad \forall a \in \mathcal{L} \tag{2}$$

where $\rho$ is some constant between 0 and 1. Jaffe [3] proposed an algorithm which obtains a rate vector $r$ such that the vector $(\cdots, \beta_s r_s, \cdots)$ is a fair allocation over the set defined by

$$\beta_s r_s \leqslant C_a - F_a, \qquad \forall s \in \mathcal{S},\ a \in \mathcal{L}_s, \tag{3}$$

$$F_a \leqslant C_a,\ \forall a \in \mathcal{L}, \qquad r_s \geqslant 0,\ \forall s \in \mathcal{S} \tag{4}$$

where $\beta_s$ is some positive constant that characterizes the priority of session $s$.

The rationale behind the fair allocation problem based on (2) is quite simple: we maximize the minimum session rate while not allowing the flow of any link to be more than some given fraction of its capacity. The rationale behind (3), (4) is somewhat more sophisticated. Primarily, it enables us to establish preferences among sessions, and to accommodate fluctuations of a session rate which depend linearly on the rate as we will demonstrate shortly.

While Jaffe's algorithm is not iterative and as a result is unsuitable for distributed operation, Hayden's algorithm may result in transient flows that are larger than some link capacities (for an example, see [4, p. 39]).

Our purpose in this paper is to propose and analyze an iterative algorithm that solves a problem that is more general than Jaffe's [3], maintains at all times feasibility of link flows with respect to capacities, and is suitable for distributed operation. To this end we generalize the set defined by (3), (4) as follows.

For each link $a \in \mathcal{L}$ and session $s \in \mathcal{S}$ let $g_a: R^+ \to R^+$ and $\beta_s: R^+ \to R^+$ be functions mapping the nonnegative portion of the real line $R^+$ into itself. We are interested in finding a rate vector $r$ such that the vector $(\cdots, \beta_s(r_s), \cdots)$ is a fair allocation over the set defined by

$$\beta_s(r_s) \leqslant g_a(C_a - F_a), \qquad \forall s \in \mathcal{S},\ a \in \mathcal{L}_s \tag{5}$$

$$F_a \leqslant C_a, \qquad \forall a \in \mathcal{L},\ r_s \geqslant 0,\ \forall s \in \mathcal{S}. \tag{6}$$

A vector $r$ with this property will be called a *fair allocation rate.*

We make the following assumptions regarding the functions $g_a(\cdot)$ and $\beta_s(\cdot)$.

*Assumption A:* For all $a \in \mathcal{L}$, $g_a(\cdot)$ is monotonically nondecreasing and, for all $s \in \mathcal{S}$, $\beta_s(\cdot)$ is continuous, monotonically increasing, and maps $R^+$ onto $R^+$. (This implies also that the inverse $\beta_s^{-1}(\cdot)$ exists, is continuous, monotonically increasing, and maps $R^+$ onto $R^+$.)

*Assumption B:* The function $H_{sa}(\cdot)$ defined by

$$H_{sa}(f) = \beta_s^{-1}[g_a(f)], \qquad \forall s \in \mathcal{S},\ a \in \mathcal{L},\ f \in R^+ \tag{7}$$

is convex and differentiable on $R^+$ and satisfies

$$H_{sa}(0) = 0.$$

Assumption $\beta$ is not very restrictive. It is satisfied in particular if both $\beta_s^{-1}(\cdot)$ and $g_a(\cdot)$ are convex, differentiable, and monotonically increasing on $R^+$, and $g_a(0) = 0$. Also the convexity assumption in Assumption B can be replaced by a concavity assumption without affecting the convergence result of the next section, but this will not be pursued further.

The introduction of the nonlinear functions $\beta_s(\cdot)$ and $g_a(\cdot)$ allows us to assign different priorities to different sessions in a more flexible manner than in (3), and allows additional freedom in mathematically expressing algorithmic design objectives. As an example let us provide justification for the use of a particular form for $g_a$ in the case where each session is a voice conversation.

Suppose that the length of time over which each session rate is averaged is short relative to the "time constant" of the counting process of the number of off-hook speakers which are currently in talkspurt mode. Since about 30 percent of a talkspurt is silence and some segments of the talkspurt need more encoding than others, we view the bit rate generated by the vocoder for session $s \in \mathcal{S}$ as a stochastic process with mean $r_s$—the rate assigned to user $s \in \mathcal{S}$. We thus implicitly assume that the vocoder has the means of dynamically reconfiguring to the demands of the voice to achieve the desired average rate. Suppose that we want to reserve excess capacity on each link so as to be able to accommodate a variation at least as large as the standard deviation of the flow on the link. Assume that the standard deviation of the rate of each session $s \in \mathcal{S}$ is $\gamma \cdot r_s$ where $0 < \gamma < 1$. For a fixed link $a \in \mathcal{L}$ let $s' \in \mathcal{S}$ be such that

$$s' = \arg\max_{s \in \mathcal{S}_a} r_s.$$

Then, by the independence of the rates of different sessions, we have, assuming $F_a \leqslant C_a$, that the standard deviation $\sigma(F_a)$ of the flow $F_a$ satisfies

$$\sigma(F_a) = \sqrt{\sum_{s \in \mathcal{S}_a} [\sigma(r_s)]^2} = \gamma \sqrt{\sum_{s \in \mathcal{S}_a} r_s^2}$$

$$\leqslant \gamma \sqrt{\left(\sum_{s \in \mathcal{S}_a} r_s\right) r_{s'}} \leqslant \gamma \sqrt{C_a r_{s'}}.$$

Suppose we take in (5)

$$\beta_{s'}(r_{s'}) = r_{s'}, \quad g_a(C_a - F_a) = \frac{1}{\gamma^2 C_a}(C_a - F_a)^2. \tag{8}$$

Then from (5), (7), and (8) we obtain

$$\sigma(F_a) \leqslant \gamma \sqrt{C_a r_{s'}} \leqslant \gamma \sqrt{C_a g_a(C_a - F_a)} = C_a - F_a.$$

We are thus guaranteed to be able to accommodate the standard deviation of the flow resulting from the fair allocation.

In the second interpretation, the length of time over which the rate is averaged is relatively long with respect to the "time constant" of the counting process of the number of off-hook speakers in talkspurt mode. In this case we deal concurrently with all the off-hook sessions and want to be able to accommodate the standard deviation around the mean of the process (i.e., the instantaneous effect of the number of speakers at the talkspurt mode is washed out by the long time average). Let $q$ be the fraction of time a speaker is in the talkspurt mode and assume his rate while in the talkspurt mode is constant. Then using notations as before

$$\sigma(F_a)=\left[\sum_{s\in \mathcal{S}_a}(r_s/q)^2\cdot q(1-q)\right]^{1/2}$$
$$\leqslant\left(\frac{1-q}{q}C_a r_s'\right)^{1/2}$$
$$\leqslant\left(\frac{1-q}{q}C_a\right)^{1/2}[g_a(C_a-F_a)]^{1/2}.$$

Again, by choosing $g_a$ as in (8) with $\gamma=(q/1-q)^{1/2}$ we obtain $\sigma(F_a)\leqslant C_a-F_a$.

The point we want to make by the above arguments is that there is often a need to allow $g_a$ to be a nonlinear function, which may depend also on $C_a$, rather than only on the excess capacity as (3) implies. The exact role of $g_a$ is up to the network designer to decide, and our formulation allows him a great deal of flexibility in this regard.

It is possible to show that *Assumptions A and B guarantee existence and uniqueness of a fair allocation rate.* The proof given below is constructive and is based on a finitely terminating algorithm, basically the one of [3]. However this algorithm, in contrast with the one of the next section, is not suitable for distributed, on-line operation since it must be restarted each time an old session is terminated or a new one is initiated.

Consider first the problem of finding a vector $r=(\cdots,r_s,\cdots)$ that maximizes

$$\min_{s\in\mathcal{S}}\beta_s(r_s)$$

over the feasible set

$$R_0=\{r\mid (5)\text{ and }(6)\text{ are satisfied}\}.$$

This is the first problem in the hierarchy of problems solved by a fair allocation rate, and can be solved simply by observing that its optimal value [i.e., $\max_{r\in R_0}\min_{s\in\mathcal{S}}\beta_s(r_s)$] is equal to

$$w_1^*=\max\left\{w\mid w\leqslant g_a\left[C_a-\sum_{s\in\mathcal{S}_a}\beta_s^{-1}(w)\right],\ a\in\mathcal{L}\right\}$$

This follows easily from the fact that both $g_a$ and $\beta_s^{-1}$ are monotonically nondecreasing. Denote

$$\mathcal{L}^*(1)=\left\{a\in\mathcal{L}\mid w_1^*=g_a\left[C_a-\sum_{s\in\mathcal{S}_a}\beta_s^{-1}(w_1^*)\right]\right\}\tag{9a}$$
$$S^*(1)=\{s\in\mathcal{S}\mid\mathcal{L}_s\cap\mathcal{L}^*(1)\neq 0\}.\tag{9b}$$

For any fair allocation rate $(\cdots,r_s,\cdots)$ the rate of the sessions in $\mathcal{S}^*(1)$ is equal to $\beta_s^{-1}(w_1^*)$, i.e.,

$$r_s=\beta_s^{-1}(w_1^*),\qquad\forall s\in\mathcal{S}^*(1)$$

while $\mathcal{L}^*(1)$ may be viewed as the set of bottleneck links the presence of which does not allow us to increase $\min_{s\in\mathcal{S}}\beta_s(r_s)$ beyond the level $w_1^*$. Therefore, for the purposes of determining further a fair allocation rate vector, the rates of the sessions $s\in\mathcal{S}^*(1)$ are fixed at $\beta_s^{-1}(w_1^*)$ and we can consider a reduced network whereby the links $a\in\mathcal{L}^*(1)$ and sessions $s\in\mathcal{S}^*(1)$ are eliminated while the capacity $C_a$ of each link $a\notin\mathcal{L}^*(1)$ is replaced by

$$C_a-\sum_{s\in\mathcal{S}^*(1)\cap\mathcal{S}_a}\beta_s^{-1}(w_1^*).$$

If $\mathcal{S}^*(1)=\mathcal{S}$ we are done; otherwise we can consider the problem of maximizing the minimal value of $\beta_s(r_s)$ in the reduced network similarly as earlier. This will determine a new optimal value $w_2^*$ with $w_2^*>w_1^*$, a new set of bottleneck links $\mathcal{L}^*(2)$, and a set of sessions $\mathcal{S}^*(2)$ such that

$$\beta_s(r_s)=w_2^*,\qquad\forall s\in\mathcal{S}^*(2)$$

in any fair allocation vector. If $\mathcal{S}^*(1)\cup\mathcal{S}^*(2)=\mathcal{S}$ we are done; otherwise we can proceed by constructing a reduced network and continue in the same manner as earlier until we exhaust all sessions. This argument constructs a fair allocation rate $r^*$ and shows that it is uniquely defined in terms of the scalars $w_1^*$, $w_2^*$, $\cdots$, and the corresponding sets $\mathcal{S}^*(1)$, $\mathcal{S}^*(2)$, $\cdots$. Note also that the session rates $r_s^*$, $s\in\mathcal{S}$ and associated ink flows $F_a^*$, $a\in\mathcal{L}$ satisfy

$$r_s^*=\min_{a\in\mathcal{L}_s}H_{sa}(C_a-F_a^*),\qquad\forall s\in\mathcal{S}.\tag{10}$$

The algorithm of the next section is based on this property.

We can also show the reverse property, namely, that if a rate vector $r^*$ satisfies (10), then it is a fair allocation rate. To see this let $\tilde r=(\cdots,\tilde r_s,\cdots)$ satisfy (10) or equivalently

$$\beta_s(\tilde r_s)=\min_{a\in\mathcal{L}_s}g_a(C_a-\tilde F_a),\qquad\forall s\in\mathcal{S}.\tag{11}$$

Observe that from the definition (9) of $w_1^*$ and (11) we obtain

$$w_1^*\geqslant\tilde w_1$$

where

$$\tilde w_1=\min_{s\in\mathcal{S}}\min_{a\in\mathcal{L}_s}g_a(C_a-\tilde F_a).$$

Let $\tilde a\in\mathcal{L}$ be any link such that

$$g_{\tilde a}(C_{\tilde a}-\tilde F_{\tilde a})=\tilde w_1.$$

Then from (11) we have

$$\tilde r_s=\beta_s^{-1}(\tilde w_1),\qquad\forall s\in\mathcal{S}_{\tilde a}.$$

The inequality $\tilde w_1\leqslant w_1^*$ implies that $\tilde F_{\tilde a}=\Sigma_{s\in\mathcal{S}_{\tilde a}}\tilde r_s\leqslant\Sigma_{s\in\mathcal{S}_{\tilde a}}r_s^*=F_{\tilde a}^*$ where $r^*$ is the fair allocation rate. But this implies that

$$\tilde w_1=g_{\tilde a}(C_{\tilde a}-\tilde F_{\tilde a})\geqslant g_{\tilde a}(C_{\tilde a}-F_{\tilde a}^*)\geqslant w_1^*.$$

Therefore, we must have $\tilde w_1=w_1^*$ and it follows that the vector $\tilde r$ solves the first problem in the hierarchy of the fair allocation problem. Proceeding similarly as earlier we can show that $\tilde r$ solves all the problems in the hierarchy of the fair allocation problem and is therefore a fair allocation rate.

We summarize the conclusions from the preceding arguments in the following proposition.

*Proposition 1:* Let Assumptions A and B hold.

1) There exists a unique fair allocation rate.

2) $r^*=(\cdots,r_s,\cdots)$ is a fair allocation rate if and only if it satisfies

$$\beta_s(r_s^*)=\min_{a\in\mathcal{L}_s}g_a(C_a-F_a^*),\qquad\forall s\in\mathcal{S}$$

or equivalently

$$r_s^*=\min_{a\in\mathcal{L}_s}H_{sa}(C_a-F_a^*),\qquad\forall s\in\mathcal{S}.$$

Some appreciation of the role of the functions $\beta_s(\cdot)$ and $g_a(\cdot)$ in influencing the allocation of communication resources in a fair allocation can be gained by making the simplifying assumption that $\beta_s(\cdot)$ and $g_a(\cdot)$ are linear of the form

$$\beta_s(r_s)=r_sb_s,\qquad\forall s\in\mathcal{S}\tag{12a}$$
$$g_a(C_a-F_a)=(C_a-F_a)q_a,\qquad\forall a\in\mathcal{L}.\tag{12b}$$

Let $r^* = (\cdots, r_s^*, \cdots)$ be a unique fair allocation rate and $F^* = (\cdots, F_a^*, \cdots)$ be the corresponding set of total link flows [cf. (1)]. Then from Proposition 1 and (12) we have

$$r_s^* \leqslant (C_a - F_a^*) q_a b_s^{-1}, \qquad \forall s \in \mathcal{S}, \quad a \in \mathcal{L}_s$$

with equality holding for all $a$ and $s$ in the "first level bottleneck" sets $\mathcal{L}^*(1)$, $\mathcal{S}^*(1)$ of (9). By adding the inequality above over all $s \in \mathcal{S}_a$ we obtain

$$F_a^* \leqslant (C_a - F_a^*) q_a \sum_{s \in \mathcal{S}_a} b_s^{-}$$

from which

$$\frac{F_a^*}{C_a} \leqslant \frac{q_a \sum_{s \in \mathcal{S}_a} b_s^{-}}{1 + q_a \sum_{s \in \mathcal{S}_a} b_s^{-1}} \qquad \forall a \in \mathcal{L} \tag{13a}$$

and

$$\frac{F_a^*}{C_a} = \frac{q_a \sum_{s \in \mathcal{S}_a} b_s^{-1}}{1 + q_a \sum_{s \in \mathcal{S}_a} b_s^{-1}} \qquad \forall a \in \mathcal{L}^*(1). \tag{13b}$$

(We note that by a similar calculation we can actually obtain the exact value of the utilization of any link $a$. It is given by

$$\frac{F_a^*}{C_a} = \frac{q_a \sum_{s \in \mathcal{S}_a^*} b_s^{-1}}{1 + q_a \sum_{s \in \mathcal{S}_a^*} b_s^{-1}} + \frac{1}{1 + q_a \sum_{s \in \mathcal{S}_a^*} b_s^{-1}} \frac{F_a^* - \bar{F}_a^*}{C_a}$$

where $\mathcal{S}_a^*$ is the subset of sessions $s \in \mathcal{S}_a$ for which link $a$ is a bottleneck link, i.e., $b_s r_s^* = (C_a - F_a^*) q_a$, and $\bar{F}_a^* = \Sigma_{s \in \mathcal{S}_a^*} r_s^*$. This formula generalizes (13).)

An important conclusion from (13) is that while the scalars $b_s$ control the session priorities, i.e., the relative allocations of rate among sessions sharing the same links, the scalars $q_a$ control the link utilizations, i.e., the level of congestion and average packet delay on individual links. Thus, if the desired maximum link utilization is $\rho$ we see from (13) that the scaling factors $q_a$ should be set at

$$q_a = \frac{\rho}{(1-\rho) \sum_{s \in \mathcal{S}_a} b_s^{-1}}$$

What is interesting about this formula is that it allows the "on-line" control of the utilization of every link *regardless of the number and priority of the sessions traversing the link.* This is particularly useful when, as is usually the case, the number and priority of sessions on each link is unknown *a priori* and changes over time. Thus, when a new session passing through a link $a$ is added or terminated, the corresponding factor $q_a$ is adjusted using the formula above in order to maintain the desired level of link utilization. It will be seen in the next section that *by adjusting the factors $q_a$ we can control not only the link utilizations but also the rate of convergence of the fair allocation algorithm.* It should be noted also that a major difficulty with traditional end-to-end window flow control with fixed window sizes is that the average delay per packet on a congested link is roughly proportional to the number of active windowed sessions traversing the link [9] and cannot be bounded via control of link utilization as in our scheme. It is also worth noting that adjustment of the function to guarantee any desired link utilization is possible even when it does not have the linear form (12b). However, the corresponding implementation is somewhat complicated.

We finally mention that in some situations it may be reasonable to consider, in addition to (5) and (6), the constraint

$$r_s \leqslant R_s, \qquad \forall s \in \mathcal{S} \tag{14}$$

where $R_s$ is given upper bound to the rate of session rate $s$. We may view $R_s$ either as a limit on rate imposed by technological restrictions or as a maximum desired rate by session $s$. The problem of finding a fair allocation rate over the set defined by (5), (6), and (14) can be reduced to the problem considered earlier by introducing for each $s \in \mathcal{S}$, an artificial link $a_s$ traversed only by session $s$ by setting the capacity $C_{a_s}$ of that link equal to

$$C_{a_s} = R_s + \beta_s(R_s)$$

and by selecting the function $g_{a_s}$ to be the identity. Then the constraint

$$\beta_s(r_s) \leqslant g_{a_s}(C_{a_s} - r_s) = C_{a_s} - r_s$$

becomes $r_s + \beta_s(r_s) \leqslant C_{a_s} = R_s + \beta_s(R_s)$ and is equivalent to (14). It can be easily shown [cf. (20)] that with the definitions above the algorithm of the next section maintains the inequality (14) after every iteration.

## III. The Algorithm

Let $r^k = (\cdots, r_s^k, \cdots)$ be the rate vector obtained after $k$ iterations and let $\{F_a^k\}$ be the corresponding set of total link flows. Assume that

$$0 \leqslant F_a^k < C_a, \qquad \forall a \in \mathcal{L}. \tag{15}$$

The new rate vector $r^{k+1} = (\cdots, r_s^{k+1}, \cdots)$ obtained at the $(k+1)$st iteration is given by

$$r_s^{k+1} = \min_{a \in \mathcal{L}_s} \{r_s^k + \gamma_a^k [H_{sa}(C_a - F_a^k) - r_s^k]\}, \qquad \forall s \in \mathcal{S} \tag{16}$$

where $\gamma_a^k$ is given by

$$\gamma_a^k = \frac{}{1 + \sum_{s \in \mathcal{S}_a} H'_{sa}(C_a - F_a^k)} \tag{17}$$

and $H'_{sa}(\cdot)$ denotes the first derivative of $H_{sa}(\cdot)$. In a practical implementation of the algorithm the link flows $F_a^k$ can either be measured (by taking a time average), or they can be mathematically computed as the sum of the session rates $r_s^k$, $s \in \mathcal{S}_a$. The session rates computed via (16), (17) will have to be translated into physical rates by software residing at the session origin nodes.

The following lemma shows among other things that property (15) is maintained by the algorithm at each iteration, and therefore *if the initial link flows $F_a^0$ are within the link capacities $C_a$ the same is true for all link flows generated by the algorithm.* This allows in particular the initiation of new sessions at zero rate and the termination of old ones without violating capacity constraints or otherwise disrupting the algorithm—a property not shared by Hayden's algorithm [4].

*Lemma 1:* Let Assumptions A and B hold and assume that the initial rate vector $r^0$ is such that

$$0 \leqslant r_s^0, \qquad \forall s \in \mathcal{S}, \quad 0 \leqslant F_a^0 < C_a, \qquad \forall a \in \mathcal{L}. \tag{18}$$

Then if $\{r^k\}$ is a rate sequence generated by the algorithm (16) with $\gamma_a^k$ given by (17) we have for all $k$

$$0 \leqslant r_s^k, \qquad \forall s \in \mathcal{S}, \quad 0 \leqslant F_a^k < C_a, \qquad \forall a \in \mathcal{L}. \tag{19}$$

Furthermore,

$$F_a^k \leqslant \sum_{s \in \mathcal{S}_a} H_{sa}(C_a - F_a^k), \qquad \forall a \in \mathcal{L}, \quad k \geqslant 1. \tag{20}$$

*Proof:* See Appendix A.

The idea behind the choice of expression (17) as well as the intuition behind Lemma 1 can be best explained by the use of Fig. 1. Let the function $G_a(\cdot): R^+ \to R^+$ be given by

$$G_a(F_a) = \sum_{s \in \mathcal{S}_a} H_{sa}(C_a - F_a). \tag{21}$$

The figure depicts the relations between $F_a^k$ and $F_a^{k+1}$, as if the network consisted of the single link $a$. $F_a^{k+1}$ is determined by intersecting the tangent to the graph of $G_a(F_a)$ at the point $(F_a^k, G_a(F_a^k))$, with the line $y = F^a$. The reader can easily convince himself that, for $k > 1$, $F_a^k$ must lie in the area where

$$F_a \leqslant G_a(F_a)$$

which gives rise to the lemma. Fig. 2 shows why just monotonicity of $G_a(\cdot)$ is not sufficient for the lemma to hold.

We can now state the main result of this paper.

*Proposition 2:* Under the assumptions of Lemma 1 the sequence $\{r^k\}$ converges to the fair allocation rate.

*Proof:* See Appendix A.

### *Rate of Convergence*

The rate of convergence of the algorithm will be derived assuming that the functions $\beta_s$ and $g_a$ are linear [cf. (12)], and that there is a single "bottleneck" link per session. In particular the following holds.

*Assumption C:* a) The functions $\beta_s(\cdot)$ and $g_a(\cdot)$ are linear of the form

$$\beta_s(r_s) = r_s b_s, \qquad \forall s \in \mathcal{S} \tag{22}$$

$$g_a(C_a - F_a) = (C_a - F_a) q_a, \qquad \forall a \in \mathcal{L} \tag{23}$$

where $b_s$, $q_a$ are positive scalars.

b) For every session $s \in \mathcal{S}$ there exists a unique link $a_s$ such that

$$r_s^* b_s = (C_{a_s} - F_{a_s}^*) q_{a_s} = \min_{a \in \mathcal{S}_s} (C_a - F_a^*) q_a \tag{24}$$

where $r^* = (\cdots, r_s^*, \cdots)$ is the unique fair allocation rate and $F^* = (\cdots, F_a^*, \cdots)$ is the corresponding set of total link flows.

Note that Assumption C implies both Assumptions A and B. The following proposition establishes a linear rate of convergence of the algorithm.

*Proposition 3:* Let Assumption C hold.

a) For every link $a$ in the "first level bottleneck set" [cf. (9)]

$$L^*(1) = \{a \in \mathcal{L} \mid w_1^* = (C_a - F_a^*) q_a \tag{25}$$

and every session $s$ in the corresponding set

$$S^*(1) = \{s \in \mathcal{S} \mid \mathcal{L}_s \cap L^*(1) \neq 0\} \tag{26}$$

there exists an integer $\bar{k}$ such that

$$F_a^k = F_a^*, \quad \forall k \geqslant \bar{k} \tag{27}$$

$$r_s^{k+1} - r_s^* = \frac{q_{a_s} \sum_{m \in \mathcal{S}_{a_s}} b_m^{-1}}{1 + q_{a_s} \sum_{m \in \mathcal{S}_{a_s}} b_m^{-1}} (r_s^k - r_s^*), \qquad \forall k \geqslant \bar{k}. \tag{28}$$

b) Let

$$\bar{\gamma} = \max_{s \in \mathcal{S}} \frac{q_{a_s} \sum_{m \in \mathcal{S}_{a_s}} b_m^{-1}}{1 + q_{a_s} \sum_{m \in \mathcal{S}_{a_s}} b_m^{-1}}. \tag{29}$$

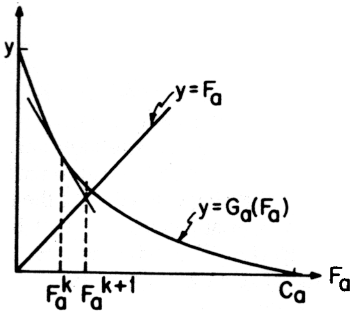

Fig. 1.

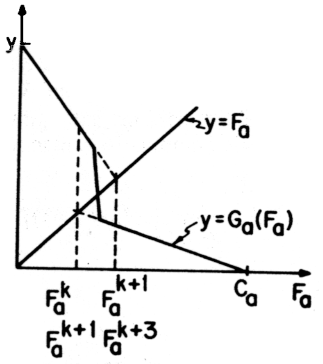

Fig. 2.

For every $\gamma \in (\bar{\gamma}, 1)$

$$|r_s^k - r_s^*| \leqslant 0(\gamma^k), \qquad \forall s \in \mathcal{S} \tag{30}$$

$$|F_a^k - F_a^*| \leqslant 0(\gamma^k), \qquad \forall a \in \mathcal{L} \tag{31}$$

where $0(\cdot)$ denotes a function such that for some constant $w$ and all $x \geqslant 0$ we have $|0(x)| \leqslant wx$.

*Proof:* See Appendix A.

A remarkable fact about the convergence ratio $\bar{\gamma}$ of (29) is that it is also an upper bound on all link utilizations [cf. (13)]. In fact for the links and sessions of the "first level bottleneck set" $\mathcal{L}^*(1)$ the convergence ratio is exactly equal to the link utilization. This shows that *by properly adjusting the link parameters $q_a$ we can control simultaneously both the link utilizations and the rate of convergence of the algorithm.*

### *A Variation of the Algorithm*

In iteration (16) we have assumed that updating of all the rates $r_s$ takes place simultaneously. It is possible to consider other related algorithms whereby a single rate $r_s$ is updated using (16), then the flows $F_a$ are updated to reflect the change in $r_s$, then another session rate is updated using (16) and so on until all the session rates are taken up cyclically in a fixed order. This one-session-at-a-time mode of operation is reminiscent of the Gauss–Seidel method for solving systems of equations and is perhaps better suited for distributed implementation. It is possible to show that the convergence result of this section holds for this modified algorithm as well.

A distributed asynchronous algorithm which is intermediate between the one above and the one of iteration (16) can be implemented as follows. Each session origin sends *at arbitrary times* (compare to the formalism of [5], [6]) along the session path a control packet containing the current rate of the session. As the packet travels to its destination the information needed to compute the right side of (16) is collected. (We assume here for simplicity a single destination per session and that each link $a$ on the session path maintains the current value of $F_a$ as the sum of all currently assigned session rates $r_s$, $s \in \mathcal{S}_a$, and the form of the function $H_{sa}(\cdot)$ for each $s \in \mathcal{S}_a$.) The destination returns the new rate to the session origin and the links along the session path.

Unfortunately the algorithm in the form just given is not guaranteed to converge and may produce link flows exceeding capacity. The difficulty is that some sessions may update their rates much faster than others and a situation may occur whereby some session may increase its rate well above its fair allocation on an empty link through several updates during the time that another session is still (due to communication delays) in the process of completing a single update. The latter session thinking that the link is empty may increase substantially its rate and find that the link has insufficient excess capacity.

It is therefore necessary to impose an additional synchronization mechanism in the preceding algorithm. One simple possibility, close in spirit to recent work of Mosely [10], is to maintain at each link a status flag for each session whereby, after a session updates its rate, the flag is set. The session is not allowed to increase (but may reduce) its rate until all other sessions sharing the link also update their rates at least once. When this occurs all status flags are reset and the sessions can update their rates again in the usual manner. This amounts to replacing iteration (16) by the iteration

$$r_s^{k+1} = \min_{a \in \mathcal{L}_s} \{R_{sa}^k(r_s^k, F_a^k)\}$$

where

$$R_{sa}^k(r_s^k, F_a^k) = \begin{cases} r_s^k + \gamma_a^k[H_{sa}(C_a - F_a^k) - r_s^k] \\ \quad \text{if the flag of session } s \text{ at link } a \text{ is reset} \\ \min\{r_s^k,\ r_s^k + \gamma_a^k[H_{sa}(C_a - F_a^k) - r_s^k]\} \\ \quad \text{otherwise.} \end{cases}$$

This algorithm, even when it is implemented totally asynchronously, guarantees that no session can increase its rate more than once before all other sessions sharing a link with it perform at least one update, thereby guaranteeing that link flows will always stay within capacity. We refer to Mosely [10] for a convergence analysis and detailed discussion of asynchronous flow control algorithms.

## Appendix A

### Proof of Lemma 1

It suffices to show (19), (20) for $k = 1$. Consider the function $G_a: R^+ \to R^+$ defined by

$$G_a(F_a) = \begin{cases} \sum_{s \in \mathcal{S}_a} H_{sa}(C_a - F_a) & \text{if } 0 \leqslant F_a \leqslant C_a \\ 0 & \text{if } F_a > C_a. \end{cases}$$

From (16) and (17) we have

$$r_s^1 = \min_{a \in \mathcal{L}_s} \left\{ r_s^0 + \frac{1}{1 - G_a'(F_a^0)} [H_{sa}(C_a - F_a^0) - r_s^0] \right\} \qquad \forall s \in \mathcal{S}$$

or

$$r_s^1 = \min_{a \in \mathcal{L}_s} \left\{ \frac{H_{sa}(C_a - F_a^0) - r_s^0 G_a'(F_a^0)}{1 - G_a'(F_a^0)} \right\} \qquad \forall s \in \mathcal{S}. \quad \text{(A.1)}$$

Since $G_a(\cdot)$ in monotonically nonincreasing we have $G_a'(F_a^0) \leqslant 0$, and since also $H_{sa}(C_a - F_a^0) \geqslant 0$ we obtain from the hypothesis $r_s^0 \geqslant 0$ and (A.1)

$$r_s^1 \geqslant 0, \qquad \forall s \in \mathcal{S}, \qquad \text{(A.2)}$$

and therefore also

$$F_a^1 \geqslant 0, \qquad \forall a \in \mathcal{L}. \qquad \text{(A.3)}$$

From (A.1) we have

$$r_s^1 \leqslant \frac{H_{sa}(C_a - F_a^0) - r_s^0 G_a'(F_a^0)}{1 - G_a'(F_a^0)}, \qquad \forall s \in \mathcal{S},\ a \in \mathcal{L}_s$$

and by adding over all $s \in \mathcal{S}_a$ we obtain

$$F_s^1 \leqslant \frac{G_a(F_a^0) - F_a^0 G_a'(F_a^0)}{1 - G_a'(F_a^0)}, \qquad \forall a \in \mathcal{L}.$$

Since $1 - G_a'(F_a^0) > 0$ we obtain from the inequality above

$$F_a^1 \leqslant G_a(F_a^0) + (F_a^1 - F_a^0) G_a'(F_a^0).$$

Since $G_a(\cdot)$ is convex the right side of this inequality is less or equal to $G_a(F_a^1)$ and we obtain

$$F_a^1 \leqslant G_a(F_a^1). \qquad \text{(A.4)}$$

Since $G_a(\cdot)$ is monotonically nonincreasing and $G_a(F) = 0$ for $F \geqslant C_a$ we obtain from (A.4)

$$F_a^1 < C_a. \qquad \text{(A.5)}$$

From (A.2)-(A.5) we obtain (19) and (20).

### Proof of Proposition 2

Denote

$$r_s^* = \liminf_{k \to \infty} r_s^k, \qquad \forall s \in \mathcal{S}.$$

Fix $s \in \mathcal{S}$ and consider a subsequence $\{r_s^k\}_{k \in \mathcal{K}_s}$ converging to $r_s^*$. We have from (16)

$$r_s^* = \lim_{\substack{k \to \infty \\ k \in \mathcal{K}_s}} \min_{a \in \mathcal{L}_s} \{r_s^{k-1} + \gamma_a^{k-1}[H_{sa}(C_a - F_a^{k-1}) - r_s^{k-1}]\}.$$

Since $\mathcal{L}_s$ consists of a finite number of links we may assume (by passing to a subsequence of $\mathcal{K}_s$ if necessary) that there exists a link $a_s$ such that

$$r_s^* = \lim_{\substack{k \to \mathcal{K}_s \\ k \to \infty}} \{r_s^{k-1} + \gamma_{a_s}^{k-1}[H_{sa_s}(C_{a_s} - F_{a_s}^{k-1}) - r_s^{k-1}]\}. \qquad \text{(A.6)}$$

Since $\{F_{a_s}^{k-1}\}_{k \in \mathcal{K}_s}$ is bounded above and below we may assume (by passing to a subsequence of $\mathcal{K}_s$ if necessary) that for some $\bar{F}_{a_s}$

$$\lim_{\substack{k \to \mathcal{K}_s \\ k \to \infty}} F_{a_s}^{k-1} = \bar{F}_{a_s}.$$

Denote also

$$\bar{\gamma}_{a_s} = \frac{1}{1 - G_{a_s}'(\bar{F}_{a_s})} = \lim_{\substack{k \in \mathcal{K}_s \\ k \to \infty}} \gamma_{a_s}^{k-1}.$$

We have from (A.6)

$$r_s^* \geqslant (1 - \bar{\gamma}_{a_s}) \liminf_{\substack{k \to \mathcal{K} \\ k \to \infty}} r_s^{k-1} + \bar{\gamma}_{a_s} H_{sa_s}(C_{a_s} - \bar{F}_{a_s})$$

$$\geqslant (1 - \bar{\gamma}_{a_s}) r_s^* + \bar{\gamma}_{a_s} H_{sa_s}(C_{a_s} - \limsup_{k \to \infty} F_{a_s}^k)$$

and finally

$$r_s^* \geqslant H_{sa_s}(C_{a_s} - \limsup_{k \to \infty} F_{a_s}^k).$$

Since the choice of $s$ was arbitrary we conclude that for every $s \in \mathcal{S}$ there exists $a_s \in \mathcal{L}_s$ such that (A.7) holds.

Let $a_1 \in \mathcal{L}$ be such that

$$a_1 = \arg\min_{a\in\mathcal{L}} g_a(C_a - \limsup_{k\to\infty} F_a^k).$$

Using the monotonicity of $\beta_s^{-1}$ we obtain

$$H_{sa_s}(C_{a_s} - \limsup_{k\to\infty} F_{a_s}^k) \geq H_{sa_1}(C_{a_1} - \limsup_{k\to\infty} F_{a_1}^k)$$

and therefore from (A.7)

$$r_s^* \geq H_{sa_1}(C_{a_1} - \limsup_{k\to\infty} F_{a_1}^k), \qquad \forall s \in \mathcal{S}_{a_1}. \tag{A.8}$$

Summing (A.8) over all $s \in \mathcal{S}_{a_1}$ we obtain

$$\liminf_{k\to\infty} F_{a_1}^k \geq \sum_{s\in\mathcal{S}_{a_1}} \liminf_{k\to\infty} r_s^k = \sum_{s\in\mathcal{S}_{a_1}} r_s^*$$
$$\geq \sum_{s\in\mathcal{S}_{a_1}} H_{sa_1}(C_{a_1} - \limsup_{k\to\infty} F_{a_1}^k).$$

On the other hand from (20) we have

$$\limsup_{k\to\infty} F_{a_1}^k \leq \sum_{s\in\mathcal{S}_{a_1}} H_{sa_1}(C_{a_1} - \limsup_{k\to\infty} F_{a_1}^k).$$

It follows that the last two inequalities as well as (A.8) hold as equations, the entire sequence $\{F_{a_1}^k\}$ converges to $\Sigma_{s\in\mathcal{S}_{a_1}} r_s^*$ while each sequence $\{r_s^k\}$, $s \in \mathcal{S}_{a_s}$, converges to $r_s^*$.

Consider now a new network derived from the previous one by deleting link $a_1$, and all the sessions traversing it. We consider the algorithm executed in the same manner as before with the same initial rates for the remaining sessions but with the capacity of each link $a \in \mathcal{L}$ replaced by

$$C_a - \sum_{s\in\mathcal{S}_{a_1}} r_s^k.$$

This will result in the same rate sequence for the sessions $s \notin \mathcal{S}_{a_1}$ as in the original algorithm. A trivial modification of the argument used to show (20) in Lemma 1 shows that we will have

$$\limsup_{k\to\infty} F_a^k \leq \sum_{s\in\mathcal{S}_a} H_{sa}(C_a - \limsup_{k\to\infty} F_a^k) \qquad \forall a \in \mathcal{L}.$$

This relation can be used to repeat the argument given earlier in order to show the convergence of the sequences $\{r_s^k\}$ to $r_s^*$ for all sessions $s \notin \mathcal{S}_{a_1}$ traversing the link $a_2$ where

$$a_2 = \arg\min_{\substack{a\in\mathcal{L}\\ a\neq a_1}} g_a(C_a - \limsup_{k\to\infty} F_a^k).$$

By repeating this procedure we will eventually exhaust all links thereby showing that each rate sequence $\{r_s^k\}$, $s \in \mathcal{S}$ converges to $r_s^*$, each flow sequence $\{F_a^k\}$, $a \in \mathcal{L}$ converges to $F_a^* = \Sigma_{s\in\mathcal{S}_a} r_s^*$, and each stepsize sequence $\{\gamma_a^k\}$, $a \in \mathcal{L}$ converges to

$$\gamma_a^* = \frac{}{1 + \sum_{s\in\mathcal{S}_a} H'_{sa}(C_a - F_a^*)}$$

By taking limits in (16) we have for all $s \in \mathcal{S}$

$$r_s^* = \lim_{k\to\infty} \min_{a\in\mathcal{L}_s} \{r_s^k + \gamma_a^k[H_{sa}(C_a - F_a^k) - r_s^k]\}$$
$$= \min_{a\in\mathcal{L}_s} \lim_{k\to\infty} \{r_s^k + \gamma_a^k[H_{sa}(C_a - F_a^k) - r_s^k]\} \tag{A.9}$$

where the interchange of min and lim above is valid since all the sequences inside the braces converge and the number of elements of $\mathcal{L}_s$ is finite. From (A.9) we obtain for all $s \in \mathcal{S}$

$$\min_{a\in\mathcal{L}_s} \gamma_a^*[H_{sa}(C_a - F_a^*) - r_s^*] = 0.$$

Since $\gamma_a^* > 0$ for all $a \in \mathcal{L}_s$ we obtain

$$r_s^* = \min_{a\in\mathcal{L}_s} H_{sa}(C_a - F_a^*), \qquad \forall s \in \mathcal{S}.$$

The result now follows from Proposition 1. Q.E.D.

## Proof of Proposition 3

a) Let $a_1$ be any link in $\mathcal{L}^*(1)$. Since $r^k \to r^*$ and $F_a^k \to F^*$, Assumption C implies that for all $s \in \mathcal{S}_{a_1}$

$$\lim_{k\to\infty} b_s^{-1} q_{a_1}(C_{a_1} - F_{a_1}^k) = r_s^k$$
$$\lim_{k\to\infty} b_s^{-1} q_a(C_a - F_a^k) > r_s^k, \qquad \forall a \neq a_1.$$

Therefore, from (16) we obtain for all $k$ sufficiently large

$$r_s^{k+1} = r_s^k + \frac{b_s^{-1} q_a(C_s - F_a^k) - r_s^k}{1 + q_a \sum_{m\in\mathcal{S}_{a_1}} b_m^{-1}} \qquad \forall s \in \mathcal{S}_{a_1}.$$

We also have

$$r_s^* = b_s^{-1} q_{a_1}(C_{a_1} - F_{a_1}^*), \qquad \forall s \in \mathcal{S}_{a_1} \tag{A.10}$$

and combining these two equations we obtain

$$r_s^{k+1} - r_s^* = \frac{q_{a_1} \sum_{m\in\mathcal{S}_{a_1}} b_m^{-1}}{1 + q_{a_1} \sum_{m\in\mathcal{S}_{a_1}} b_m^{-1}} (r_s^k - r_s^*)$$
$$+ \frac{b_s^{-1} q_{a_1}(F_{a_1}^* - F_{a_1}^k)}{1 + q_{a_1} \sum_{m\in\mathcal{S}_{a_1}} b_m^{-1}} \tag{A.11}$$

We have

$$F_{a_1}^{k+1} - F_{a_1}^* = \sum_{m\in\mathcal{S}_{a_1}} (r_s^{k+1} - r_s^*), \quad F_{a_1}^k - F_{a_1}^* = \sum_{m\in\mathcal{S}_{a_1}} (r_s^k - r_s^*)$$

so by adding (A.11) over all $s \in \mathcal{S}_{a_1}$ we obtain for all $k$ sufficiently large

$$F_{a_1}^{k+1} - F_{a_1}^* = 0.$$

This proves (27) and from (A.11) we also obtain (28).

b) From part a) we have that (30) and (31) hold for $a \in \mathcal{L}^*(1)$ and $s \in \mathcal{S}^*(1)$. We will prove that they hold for $a$ and $s$ in the "second level bottleneck sets" $\mathcal{L}^*(2)$ and $\mathcal{S}^*(2)$, respectively. The same method of proof can then be used to show (30) and (31) for $a \in \mathcal{L}^*(3)$ and $s \in \mathcal{S}^*(3)$, etc., until all links and sessions are exhausted. To this end we will need the following lemma the proof of which is left for the reader.

*Lemma 2:* Consider two nonnegative sequences $\{x_k\}$, $\{u_k\}$ such that for some $p \in (0,1)$

$$x_{k+1} \leq p x_k + u_k, \qquad \forall k \geq 0$$

and for some $y$ and $t \in (p, 1)$ we have

$$u_k \leq y t^k, \qquad \forall k \geq 0.$$

Then for each $\gamma \in (t, 1)$ there exists $w$ such that

$$x_k \leq w \gamma^k, \qquad \forall k \geq 0.$$

Returning to the proof of Proposition 3b), let $a_2$ be any link in $\mathcal{L}^*(2)$. The set of sessions traversing $a_2$ consists of the two sets

$$B_{a_2} = \mathcal{S}^*(2) \cap \mathcal{S}_{a_2}$$

$$\bar{B}_{a_2} = \mathcal{S}^*(1) \cap \mathcal{S}_{a_2}.$$

For $s \in B_{a_2}$ we have similarly as earlier [cf. (A.11)] for all sufficiently large $k$

$$r_s^{k+1} - r_s^* = \frac{q_{a_2} \sum_{m \in \mathcal{S}_{a_2}} b_m^{-1}}{1 + q_{a_2} \sum_{m \in \mathcal{S}_{a_2}} b_m^{-1}} (r_s^k - r_s^*) + \frac{b_s^{-1} q_{a_2} (F_{a_2}^* - F_{a_2}^k)}{1 + q_{a_2} \sum_{m \in \mathcal{S}_{a_2}} b_m^{-1}} \tag{A.12}$$

Adding this equation over $s \in B_{a_2}$ and using the fact

$$F_{a_2}^k - F_{a_2}^* = \sum_{s \in B_{a_2}} (r_s^k - r_s^*) + \sum_{s \in \bar{B}_{a_2}} (r_s^k - r_s^*)$$

we obtain

$$\sum_{m \in B_{a_2}} (r_s^{k+1} - r_s^*) = \frac{q_{a_2} \left( \sum_{m \in \bar{B}_{a_2}} b_s^{-1} \right) \sum_{m \in B_{a_2}} (r_s^k - r_s^*) - q_{a_2} \left( \sum_{m \in B_{a_2}} b_s^{-1} \right) \sum_{m \in \bar{B}_{a_2}} (r_s^k - r_s^*)}{1 + q_{a_2} \sum_{m \in \mathcal{S}_{a_2}} b_s^{-1}} \tag{A.13}$$

Since $\bar{B}_{a_2} \subseteq \mathcal{S}^*(1)$ it follows from what has been shown already that for any $\gamma \in (\bar{\gamma}, 1)$

$$\left| \sum_{m \in \bar{B}_{a_2}} (r_s^k - r_s^*) \right| \tag{A.14}$$

where $\bar{\gamma}$ is given by (29). It follows from Lemma 1 and (A.13) that for any $\gamma \in (\gamma, 1)$

$$\left| \sum_{m \in B_{a_2}} (r_s^{k+1} - r_s^*) \right| \tag{A.15}$$

Adding (A.14) and (A.15) we obtain for any $\gamma \in (\bar{\gamma}, 1)$

$$|F_{a_2}^k - F_{a_2}^*| \leq 0(\gamma^k)$$

and from (A.12)

$$|r_s^{k+1} - r_s^*| \leq 0(\gamma^k).$$

This shows (30) and (31) for $a \in \mathcal{L}^*(2)$ and $s \in \mathcal{S}^*(2)$, and proceeding similarly we can show (30) and (31) for all $a \in \mathcal{L}$ and $s \in \mathcal{S}$. Q.E.D.

## References

[1] T. Bially, B. Gold, and S. Seneff, "A technique for adaptive voice flow control in integrated packet networks," *IEEE Trans Commun.*, vol. COM-28, pp. 325-333, 1980.

[2] E. M. Gafni, "The integration of routing and flow control for voice and data in a computer communication network," Ph.D. dissertation, Dep. Elec. Eng. Comput. Sci., Mass. Inst. Technol., Cambridge, MA, Aug. 1982.

[3] J. M. Jaffe, "Bottleneck flow control," *IEEE Trans. Commun.*, vol. COM-29, pp. 954-962, 1981.

[4] H. Hayden, "Voice flow control in an integrated network," M.S. thesis, Dep. Elec. Eng. Comput. Sci., Mass. Inst. Technol., Cambridge, MA, June 1981.

[5] D. P. Bertsekas, "Distributed asynchronous computation of fixed points," *Mathematical Programm.*, vol. 27, pp. 107-120, 1983.

[6] D. P. Bertsekas, "Distributed dynamic programming," *IEEE Trans. Automat. Contr.*, vol. AC-27, pp. 610-616, 1982.

[7] F. H. Moss and A. Segall, "An optimal control approach to dynamic routing in networks," *IEEE Trans. Automat. Contr.*, vol. AC-27, pp. 329-339, 1982.

[8] B. Hajek and R. G. Ogier, "Optimal dynamic routing in communication networks with continuous traffic," Coordinated Science Lab., Univ. Illinois, Urbana, Rep. 1982.

[9] R. G. Gallager, Class Notes on Data Communication Networks, Course 6.263, Dep. Elec. Eng. Comput. Sci., Mass. Inst. Technol., Cambridge, MA, 1983.

[10] J. Mosley, "Asynchronous distributed flow control algorithms," Ph.D. dissertation, Dep. Elec. Eng. Comput. Sci., Mass. Inst. Technol., Cambridge, MA, June 1984.

[11] M. Gerla and L. Kleinrock, "Flow control: A comparative survey," *IEEE Trans. Commun.*, vol. COM-28, pp. 553-574, 1980.

[12] D. Oshinski, "Use of fair rate assignment algorithms in networks with bursty sessions," M.S. thesis, Dep. Elec. Eng. Comput. Sci., Mass. Inst. Technol., Cambridge, MA, May 1984.

**Eli M. Gafni** received the engineering degree in electrical engineering from the Technion—Israel Institute of Technology, Haifa, Israel, the M.S. degree from the University of Illinois, Urbana-Champaign, in 1979, and the Ph.D. degree in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 1982.

From 1972 to 1977, he served in the Israeli Defense Force as a Technical Officer, where he participated in projects involving estimation and control. He is currently an Assistant Professor in the Department of Computer Science, University of California, Los Angeles. His research interests are in communication networks, distributed algorithms, and optimization.

**Dimitri P. Bertsekas** (S'70-SM'77-F'84) received the Diploma of Mechanical and Electrical Engineering from the National Technical University of Athens, Athens, Greece, in 1965, the M.S.E.E. degree from George Washington University, Washington, DC, in 1969, and the Ph.D. degree in electrical engineering from Massachusetts Institute of Technology, Cambridge, in 1971.

He has served on the faculty of Stanford University, Stanford, CA, and the University of Illinois, Urbana-Champaign. He is currently Professor of Electrical Engineering and Computer Science at M.I.T. His research interests are in optimization theory and algorithmic aspects of communication networks. He is the author of *Dynamic Programming and Stochastic Control* (New York: Academic, 1976) and *Constrained Optimization and Lagrange Multiplier Methods* (New York: Academic, 1982) and coauthor of *Stochastic Optimal Control: The Discrete Time Case* (New York: Academic, 1978).