A New Approach for Asynchronous Distributed Rate Control of Elastic Sessions in Integrated Packet Networks

Santosh P. Abraham and Anurag Kumar

Abstract— **We develop a new class of asynchronous distributed algorithms for the explicit rate control of elastic sessions in an integrated packet network. Sessions can request for minimum guaranteed rate allocations (e.g., MCRs in the ATM context), and, under this constraint, we seek to allocate the max-min fair rates to the sessions. We capture the integrated network context by permitting the link bandwidths available to elastic sessions to be stochastically time varying. The available capacity of each link is viewed as some statistic of this stochastic process (e.g., a fraction of the mean, or a large deviations Equivalent Service Capacity (ESC)). For fixed available capacity at each link, we show that the vector of max-min fair rates can be computed from the root of a certain vector equation. A distributed asynchronous stochastic approximation technique is then used to develop a provably convergent distributed algorithm for obtaining the root of the equation, even when the link flows and the available capacities are obtained from on-line measurements. The switch algorithm does not require per connection monitoring, nor does it require per connection marking of control packets. A virtual buffer based approach for on-line estimation of the ESC is utilised. We also propose techniques for handling large variations in the available capacity owing to the arrivals or departures of CBR/VBR sessions. Finally, simulations results are provided to demonstrate the performance of this class of algorithms in the local and wide area network context.**

I. INTRODUCTION

Traffic generated by store-and-forward data transfer applications is often called *elastic*, as such data transfer sessions can be served at varying rates even within each session. Hence, such traffic is amenable to handling by a "best-effort" service in the network; i.e., the service expects the data flow to adapt to the time varying available bandwidth. Elastic traffic can be economically supported by utilising the bandwidth left over after serving *stream* type traffic, which carries temporally sensitive information, typically real-time audio and video. Rate adaptation of elastic sessions requires some kind of feedback between the network and the session sources. This feedback can be implicit (via acknowledgments or packet loss indications, as in Internet's TCP), or explicit (via control packets circulating between the network and the session sources, as in the ABR service in ATM networks).

In this paper, we develop a new class of algorithms for the *explicit rate control of best-effort sessions* in integrated packet networks. The network model that we work with goes beyond the models used in existing work in two important ways: (1) We allow each elastic session to request a minimum transfer rate from the network. To this end, we adopt *an extension of the usual max-min fair (MMF) bandwidth sharing concept*. (2) The service integration aspect is incorporated by *modelling the available bandwidth (for best effort service) at each link as a stochastic process*, the motivation being that higher priority stream traffic takes away a random amount of the bandwidth.

The conventional notion of max-min fairness (see [6]) does not consider the case where some sessions may demand a minimum throughput. In [14] the authors define fair allocation over a constraint set as the lexicographically maximum vector in this set. This is a natural generalisation of the usual MMF concept, and we adopt it in this paper. This MMF notion has also been used in the ABR context in [21] and [16].

The formulation that leads to MMF rates assumes that the link capacities available to sessions are fixed numbers. We reconcile this with our model of stochastic available link capacities by defining for the available capacity random process of a link, say l (in the set of links $\mathcal{L}$), a statistic that is link l's available *capacity* C_l. We consider two such statitistics in this paper: a fraction (e.g., 0.95) of the mean available capacity, and a large deviation Equivalent Service Capacity (ESC). We then seek the MMF rate vector for the problem in which the fixed capacity of each link l is taken to be C_l.

Next we consider the development of distributed asynchronous algorithms for computing the MMF rate allocation. Instances of the distributed algorithm need to operate at each output port of each packet switch, in such a way that the MMF rate is computed and communicated to each session source.

It is now well recognized that the predominant use of the best-effort service in packet networks is for "web downloads" and email. A large proportion of the elastic sessions involve only a few kilobytes of data, and hence are short lived, lasting no more than a few round trip times. From the point of view of the MMF formulation this results in a rapidly changing session topology. It is clearly infeasible to design an accurate and responsive distributed explicit rate control for such a situation. There is, however, another approach to handling elastic sessions, and that is to set up *elastic virtual paths* between various network edge points (for example, between the edge routers of an enterprise interconnected by a packet network). The elastic sessions (e.g., TCP controlled sessions) between the clients and servers "behind" these edge points then share the elastic virtual paths. The rates allocated to the virtual paths can be dynamically controlled by an algorithm such as the one we develop. These elastic virtual paths can be expected to be long lived. Thus in the design of the algorithm that we propose our aim is to be able to track the MMF rates in the presence of

1. Short time scale variations in the available capacity of links due to intrinsic rate variations of the higher priority stream traffic
2. Propagation delays and asynchronous updates
3. Long time scale variations in the available capacity of links

Based on research supported by a grant from Nortel Networks.

Anurag Kumar is with the Dept. of Electrical Communication Engg., Indian Institute of Science, Bangalore 560 012, INDIA; e-mail: anurag@ece.iisc.ernet.in. Santosh Abraham is now with Lucent Technologies Bell Labs, Holmdel, N.J., USA; e-mail spabraham@lucent.com

due to the entry/exit of large bandwidth streaming traffic sessions

With the above objectives in mind, our approach to developing the algorithms is the following. We first show that the MMF solution (for the fixed link capacities, C_l) can be calculated from the solution of a set of coupled equations, one for each link. Since the capacities C_l are statistics of random processes, only noisy estimates of $C_l, l \in \mathcal{L}$, can be obtained on-line. Hence to obtain a sequence of iterates that converge to the MMF rates, *we take recourse to a distributed asynchronous stochastic approximation algorithm* (see [8]). The structure of the stochastic approximation iteration ensures provable convergence in the presence of asynchrony and delays. *The algorithm has a simple update step, requires no explicit information exchange between switches, does not require per flow monitoring at the switches, or even per flow marking of control packets, and hence can yield an efficient implementation.*

To compensate for the effect of longer term changes in available capacity (owing to arrival and departures of CBR/VBR connections) auxilliary capacity change detection methods are used to reset the gains of the stochastic approximation algorithm.

As mentioned above we examine two statistics of a link capacity process as the target available capacity of the link. A fraction of the mean capacity is a simple naive approach, for comparison. The motivation of such an approach may be to maintain the occupancy of the link at a certain level. In order to control switch output buffer occupancy, however, we can define a more sophisticated measure called the Equivalent Service Capacity (ESC). This is the dual of the equivalent bandwidth concept for a source with a stochastic sending rate into a queue with a fixed service rate. The ESC is the constant input rate that can be applied to a queue with a stochastic service process, such that the queue length is constrained below a specified level with a large probability, thus ensuring low loss probabilities and low delay, while ensuring good utilisation of the time varying service rate. An on-line measurement based estimation algorithm is outlined for computing the ESC. The ESC estimates are then used in the stochastic approximation algorithm. Thus the session rates will converge to the MMF rates, calculated with respect to the ESCs of all the links, and the flows into the link buffers will keep the queue lengths small.

We shall use the ATM networking framework for illustration purposes in this paper. The best-effort sessions in ATM networks are expected to be carried using the ABR service. The ABR protocols incorporate special RM (Resource Management) cells that enable the communication of an explicit rate value to a session source. We have sought to use a minimal number of features provided by the ABR framework, thus keeping the discussion relevant to other packet network technologies that may provide for feedback based control of session rates.

We have provided a section discussing various issues arising in the implementation of the algorithm, and we present a simulation study with networks having different delay parameters. One of the issues discussed is the choice of an initial gain for the algorithm. It is clear that in situations with large round trip times, any feedback control mechanism is adversely affected. In order to avoid large transients in the cases with large round trip times a low starting gain is used in the initial phase of a control cycle.

Early work on MMF rate control in packet networks was done in the context of packet voice sessions; see [15], [14], [25]. The basic framework is the one described in [14]. The design of explicit rate control algorithms for elastic sessions, in the ATM/ABR service context, has received much attention in the literature in the last five to six years. In [7] there is a comprehensive survey of the issues, and the state of the art in rate control algorithms until that date. Early efforts to develop explicit rate MMF algorithms attempted basically to implement variations of the well known centralised algorithm (see [6]) in a distributed fashion; the algorithms reported in [9] and [18] are important examples of this approach. A combination of clever heuristics gave rise to the ERICA algorithm [17], which was adopted almost as a benchmark by the ATM forum, and has seen many implementations. In our work, we have shown the MMF rate allocation problem as being equivalent to obtaining the root of a certain vector equation, and have then developed a provably convergent algorithm using the distributed stochastic approximation approach. Other control theoretic approaches include the work reported in [21] and [27].

The paper is organised as follows. In Section I-A we provide a summary of the basic network model, and the various model related notation that runs through the entire paper. In Section II we review the basic theory of MMF rate allocation, for a network with fixed link capacities, and provide a way to think about the computation of MMF rates that will be useful in the development of our algorithms. We compare MMF with other fairness proposals in the literature. In Section III we show that the MMF vector can be calculated from the root of a set of couple equations, one equation for each link. In Section IV we examine the question of what is meant by "available capacity" when the actual available capacity of a link is a random process. The available capacity of a link is a statistic of that random process. The MMF vector that we seek is the one for which the capacity of each link is taken to be its available capacity. In Section V we show how an asynchronous distributed stochastic approximation can be used to solve the root finding problem above, even when the link flows and the available capacities of links are available only as noisy on-line estimates. In Section VI, we describe a virtual buffers based approach for estimating the available capacity based on the ESC concept. In Section VII we discuss some implementation issues, choice of parameter values, and provide techniques for handling changes in the available capacities of links owing to arrivals and departures of CBR/VBR sessions. In Section VIII we provide a detailed simulation study of the performance of our algorithm in an example network.

A. The Model and Notation

We assume that a session comprises a source and a destination node; sessions from the source node traverse a *fixed sequence of links* to reach the destination node. Thus the network topology, the link capacities, the sessions and their routes are all given and static. These assumptions are standard in formalisations of such problems; see, for example, [14].

The cell stream from each source is viewed as a *fluid*. We assume that each source has an *infinite* backlog of fluid, and can transfer it to the network at any specified rate (note that a maximum transfer rate from a source can be easily incorporated by augmenting the network topology with a source access link with capacity equal to the source transfer rate limit). Every link has an *available capacity* to be shared among the elastic sessions that use that link; to begin with, we take this available capacity to be a constant for each link.

Our notation parallels that used in [14]. If A is a set, then $\mid A \mid$ denotes the size of, or the number of elements in, the set A. ϕ denotes the empty set. If $(x_1, x_2, \ldots, x_n)$ is a real valued vector, then $(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n)$ denotes the elements of the vector ordered in ascending order. The following notation describes the network model:

$\mathcal{S}$ the set of sessions
$\mathcal{L}$ the set of links
C_l the capacity of link $l \in \mathcal{L}$ (this is to be viewed as the capacity of link l available to best-effort sessions; initially we view this as a given constant value for each link)
$\mathcal{C}$ denotes the ordered set $(C_l, l \in \mathcal{L})$
$\mathcal{L}_s$ the set of links used by session $s \in \mathcal{S}$
$\mathcal{S}_l$ the set of sessions through link $l \in \mathcal{L}$
r_i the rate of the ith session, $1 \leq i \leq \mid \mathcal{S} \mid$; $r = (r_1, r_2, \ldots, r_{|\mathcal{S}|})$ denotes the *rate vector*
μ_s the minimum cell rate for session $s \in \mathcal{S}$
$\mathcal{M}$ the set $\{\mu_s : s \in \mathcal{S}\}$

For a rate vector r, and $l \in \mathcal{L}$ denote the total flow through link l by $f_l(r) = \sum_{s \in \mathcal{S}_l} r_s$.
Note that the 4-tuple $(\mathcal{L}, \mathcal{C}, \mathcal{S}, \mathcal{M})$ characterises an instance of the bandwidth sharing problem. Thus we will say, for example, that the rate vector r is feasible for $(\mathcal{L}, \mathcal{C}, \mathcal{S}, \mathcal{M})$, or that r is the max-min fair rate vector for $(\mathcal{L}, \mathcal{C}, \mathcal{S}, \mathcal{M})$, etc.

II. MMF Bandwidth Sharing with Minimum Session Rates

A. The MMF Rate Vector and Bottleneck Links

We adopt the generalisation of the notion of MMF rate allocation that is defined in [14].

Definition II.1: A rate vector r is **feasible** for the problem $(\mathcal{L}, \mathcal{C}, \mathcal{S}, \mathcal{M})$ if for all $s \in \mathcal{S}, r_s \geq \mu_s$ and for all $l \in \mathcal{L}$, $f_l(r) = \sum_{s \in \mathcal{S}_l} r_s \leq C_l$. □
Note that the set of feasible vectors is non-empty iff for all $l \in \mathcal{L} \quad \sum_{s \in \mathcal{S}_l} \mu_s \leq C_l$. We will assume that this is so, *with strict inequality*, in all the following discussions. Recalling the notation in Section I-A, we define the MMF rate vector as follows.

Definition II.2: Let $x = (x_1, x_2, \cdots, x_n), y = (y_1, y_2, \cdots, y_n) \in \mathcal{R}^n$. Then y is defined as **lexicographically larger** than x (denoted $>_{lex}$) if $\tilde{y}_1 > \tilde{x}_1$, or if $\tilde{y}_1 = \tilde{x}_1$ then $\tilde{y}_2 > \tilde{x}_2$, etc. If $\tilde{y}_i = \tilde{x}_i,\ 1 \leq i \leq n$, then $y =_{lex} x$. □

Definition II.3: A feasible rate vector r is **max-min fair (MMF)** for $(\mathcal{L}, \mathcal{C}, \mathcal{S}, \mathcal{M})$ if r is lexicographically the largest among all the feasible rate vectors. □
The following definition is the extension of the one in [6] to the problem with MCRs.

Definition II.4: Given a rate vector r, a link l is said to be a **bottle-neck link** for a session j if
(i) link l is saturated, i.e., $f_l(r) = C_l$, and
(ii) for all the sessions $s \in \mathcal{S}_l$, such that $r_s > \mu_s$, $r_s \leq r_j$; i.e., every session in l, that is not at its minimum rate, has flow no more than that of session j, or equivalently

$$r_s \leq \max\{\mu_s, r_j\}$$

□
The following theorem then relates the definition of MMF to the notion of bottleneck links.

Theorem II.1: If r is a feasible rate vector, then the following statements are equivalent:
(i) r is max-min fair.
(ii) Every session $s \in \mathcal{S}$ has a bottle-neck link.

Proof:
$(i) \Rightarrow (ii)$ Let r be MMF. Let $\exists s \in \mathcal{S}$ such that s does not have a bottle-neck link. Then, for each $l \in \mathcal{L}_s$ do one of the following
(a) if $C_l > f_l(r)$, then let $\delta_l = C_l - f_l(r)$
(b) if $C_l = f_l(r)$ and $\exists k \in \mathcal{S}_l$ such that $r_k > \max\{\mu_k, r_s\}$, then let $\delta_l = r_k - \max\{\mu_k, r_s\}$
Finally, let $\delta = \min_{l \in \mathcal{L}_s} \delta_l$. Now add δ to r_s. If the minimising δ corresponds to a case (a), then the net effect is to increase r_s without affecting any other rate; we thus have a lexicographically larger rate vector. If δ corresponds to a case (b) then subtract it from the corresponding r_p with $r_p > r_s$; notice that by doing this we still have $r_p \geq r_s$. We have not affected any r_p with $r_p \leq r_s$, or violated the minimum rate of any session. The new rate vector is lexicographically larger. In each case we have a contradiction to r being MMF.
$(ii) \Rightarrow (i)$ We are given that r is such that every session has a bottleneck link. The only way to get a lexicographically larger vector than r is to strictly increase the rate of some session, say s. Now consider the bottleneck link l_s for s. In order to increase r_s, the rate of some session $p \in \mathcal{S}_{l_s}$ with $r_p \leq r_s$ will have to be decreased. The resulting rate vector cannot be lexicographically larger than r; i.e., r is the MMF rate vector. □

B. A Useful Characterisation of the MMF Rate Vector

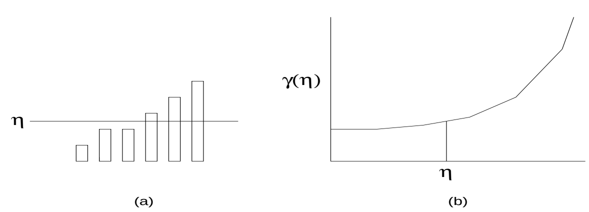

Fig. 1. Two depictions of the function $\gamma(\eta)$; the vertical bars in (a) are the minimum rate values arranged in ascending order; the break points in the piecewise linear curve in (b) are these minimum rate values; $\gamma(0) = \sum_{i=1}^{N} \mu_i$.

In order to better understand this notion of max-min fairness consider the fair allocation problem for N sessions with minimum rates $(\mu_i, i = 1, \ldots, N)$ on a single link with capacity C $(> \sum_{i=1}^{N} \mu_i)$. Define, for $\eta \geq 0$ the function $\gamma(\eta) = \sum_{i=1}^{N} \max(\mu_i, \eta)$, and solve for η^* such that $\gamma(\eta^*) = C$. Let the allocation r_i for each session $i, i = 1, \ldots, N$ be given by $r_i = \max(\mu_i, \eta^*)$.

It is useful to picture the function $\gamma(\eta)$ as shown in Figure 1. In part (a) of the figure, the μ_i values are arranged in ascending order and shown as vertical bars of increasing height; the value of η is shown as a line cutting across these bars; for this value of η each session with $\mu_i \leq \eta$ gets the rate η, and every other session gets the rate μ_i; the total flow is $\gamma(\eta)$. With part (a) of the figure in mind, we plot the function $\gamma(\eta)$ vs. η in part (b); the function is piecewise linear and the slope of the function at an η is the number of sessions with minimum rate less than η. Observe that with η as shown in part (a) of Figure 1, and $r_i = \max(\mu_i, \eta)$ $1 \leq i \leq N$, η is the minimum flow over all the sessions. *This minimum flow is maximised by setting $\eta = \eta^*$ defined above*. Observe that this allocation has the property that a session can increase its rate only if some session with lower rate reduces its rate, or if some session's rate falls below it minimum rate.

For a network of links, define the $\mid \mathcal{L} \mid \times \mid \mathcal{S} \mid$ matrix A with 0-1 elements, whose element in row l and column s is 1 if the session s flows through link l; otherwise that element is 0. Let $\underline{C}$ denote a column vector of link capacities with row indices corresponding to those of the matrix A. Let $\underline{\mu}$ denote the column vector of minimum rate values with row indices corresponding to those of the matrix A. Let $\underline{x}$ denote the column vector of the amounts by which the flows in a feasible vector r are more than the corresponding minimum rate values. In terms of this notation, consider the linear program

$$\begin{array}{rcl} & \max & \eta \\ & A(\underline{x} + \underline{\mu}) & \leq & \underline{C} \\ \forall s \in \mathcal{S} & x_s + \mu_s & \geq & \eta \\ & \underline{x}, \eta & \geq & 0 \end{array}$$

The form of the solution to this linear program is related to the problem for a single link that we discussed above. Consider the same problem for each link in the network, and let η_l be associated with link l. Now define, for each link l, $\gamma_l(\eta_l) = \sum_{s \in \mathcal{S}_l} \max(\mu_s, \eta_l)$, and solve $\gamma_l(\eta_l) = C_l$, for every $l \in \mathcal{L}$, to yield $\eta_l, l \in \mathcal{L}$. Pick the smallest of these $\eta_l, l \in \mathcal{L}$. The value η so obtained can be shown to be the solution of the above linear program (the complementary slackness conditions can be checked; we do not provide details here). A saturated link (or $\mathrm{argmin}_{l \in \mathcal{L}} \eta_l$) will be the bottleneck link for all the sessions through it. Consider now the network with all saturated links removed and the bandwidth utilised by their sessions in the other links subtracted from the links' capacities and form the same linear program in the reduced network. Continue this until all sessions have at least one saturated link. Notice that the rate allocation of the sessions so obtained is such that every session has at least one bottleneck link and hence the allocation is MMF; the details are provided in [5]. The algorithm just outlined is a centralised MMF rate computing algorithm and parallels the one without minimum session rates given in [6].

C. Comparison of MMF with other Fairness Notions

Other fairness notions have been proposed in the literature. Here we provide a comparison between these in the context of a single link, of capacity C, with n sessions, where the jth session has MCR μ_j. First consider "MCR plus equal share of the balance of the capacity" and "MCR plus a share of the balance proportional to the MCR". It is easy to see that there exist nonnegative nondecreasing functions $g_j(\eta), \eta \geq 0, 1 \leq j \leq n$, with $g_j(0) = \mu_j$, such that the fair rates can be obtained from the solution of the equation $C - \sum_{1 \leq j \leq n} g_j(\eta) = 0$. For max-min fairness, $g_j(\eta) = \max\{\mu_j, \eta\}$ (as discussed above). For the other two notions $g_j(\eta) = \mu_j + \alpha_j \eta$, where $\alpha_j = 1$ and $\alpha_j = \frac{\mu_j}{\sum_{1 \leq i \leq n} \mu_i}$, respectively. Given the solution η of the above equation, for MMF the fair rate of source j is $\max\{\mu_j, \eta\}$, and for the other two fairness notions the rate of source j is $\mu_j + \alpha_j \eta$. Notice that in each case the problem for the network is still to solve a certain equation to obtain a number η. The advantages of MMF are the following:

1. If max-min fairness is used, the network simply needs to feed back η. In the ATM/ABR context, this number plays the role of the explicit rate (ER), and can be placed in the ER field of the returning resource management (RM) cells. Note that when an RM cell passing through a link has to be updated *the updation does not need to be aware of the session to which the cell belongs*; the same value of ER works for every session. Not having to do a session parameter look up is considered by implementers as a useful saving in RM cell processing time. Sources then follow their *normal* source behaviour and compute their rates as $\max\{\mu_j, \eta\}$; thus no change in the normal ABR source behaviour is needed to accommodate this notion of fairness.
2. On the other hand if either of the other two notions of fairness is used, the switches need to feedback either (i) $\alpha_j \eta$, or (ii) just η; source j will then compute its rate as $\mu_j + \alpha_j \eta$. In (i) a session parameter look-up will need to be done at the switch for each RM cell to determine the value of α_j; in (ii) each source will need to know the value of α_j which will depend on which other sessions it is sharing the link with. The former is inefficient, and the latter is impractical. Further, the source computation $\mu_j + \alpha_j \eta$ changes the way ER is used in ABR source behaviour.

Kelly [19] formulates the bandwidth sharing objective as the vector of rates that maximises the total user utility. The utility that a rate provides to a user is modelled as a strictly concave, increasing, and continuously differentiable function. The notion of *proportional fairness* is obtained. Consider again the single link example of this section with session (or user) MCRs, and note that if all users have the same utility function (e.g., $\log(r_s + 1)$), then the utility maximising rate allocation is the MMF allocation. This is easily seen from Figure 1; if any of the sessions that have MMF rate greater than η^* was given additional rate, thereby reducing the rate of a session with MMF rate η^*, then the total utility will decrease.

In view of the above arguments we can assert that max-min fairness is a useful bandwidth sharing notion to adopt. We also believe that, based on the $g_j(\cdot)$ functions introduced above, everything else that we do subsequently in this paper goes through even if we do not use MMF; it is just that the η values that the algorithm yields have to be used differently.

III. Max-Min Fair Allocation as the Solution of a Vector Equation

A centralised algorithm outlined above and discussed in [2] restates the problem of finding the max-min fair rate vector as one of obtaining the appropriate link control parameters (LCPs). We now show that a correct (not necessarily unique) vector of link control parameters is a solution of a certain vector equation.

Theorem III.1: For the max-min fair rate allocation problem $(\mathcal{L}, \mathcal{C}, \mathcal{S}, \mathcal{M})$ let $\tilde{\mathcal{L}}$ denote the set of links that are not bottlenecks for any session ($\tilde{\mathcal{L}}$ is the set of links that remain unsaturated at the termination of the centralised algorithm discussed at the end of Section II-B). Consider any vector $(\eta_l,\ l \in \mathcal{L})$ such that

$$(i) \qquad \min_{j \in \mathcal{L}_s} \eta_j \;=\; \min_{j \in \mathcal{L}_s \setminus \tilde{\mathcal{L}}} \eta_j \quad \text{for all } s \in \mathcal{S}$$

$$(ii) \qquad \sum_{s \in \mathcal{S}_l} \max(\mu_s, \min_{j \in \mathcal{L}_s} \eta_j) \;=\; C_l \quad \text{for all } l \in \mathcal{L} \setminus \tilde{\mathcal{L}}$$

Then the allocation obtained as $r_s = \max(\mu_s, \min_{j \in \mathcal{L}_s} \eta_j)$ is the max-min fair allocation.

Proof: From a centralised algorithm (see [2]) it is clear that such a vector $(\eta_l,\ l \in \mathcal{L})$ exists. By Theorem II.1 it is sufficient to show that with $r_s, s \in \mathcal{S}$ as defined in the theorem, every session $s \in \mathcal{S}$ has a bottleneck link. Consider any $s \in \mathcal{S}$. Let $l_s \in \mathcal{L}_s \setminus \tilde{\mathcal{L}}$ [1] be such that $\eta_{l_s} = \min_{j \in \mathcal{L}_s} \eta_j$. The link l_s is saturated, by hypothesis (ii) of the theorem. Also $r_s = \max(\mu_s, \eta_{l_s})$. It follows that $\forall q \in \mathcal{S}_{l_s}$

$$r_q = \max(\mu_q, \min_{j \in \mathcal{L}_q} \eta_j) \leq \max(\mu_q, \eta_{l_s}) \leq \max(\mu_q, r_s)$$

Hence by Definition II.4, l_s is a bottleneck link for $s \in \mathcal{S}$.
□

Consider the case when there are no nonbottleneck links, i.e., $\tilde{\mathcal{L}}$ is empty and every link is a bottleneck for at least one session. Let $\underline{\eta} = (\eta_l,\ l \in \mathcal{L})$ be the vector of link control parameters and $\underline{C} = (C_l,\ l \in \mathcal{L})$ the vector of link capacities. Define a vector function $\underline{f}(\underline{\eta}) = (f_l(\underline{\eta}),\ l \in \mathcal{L})$ with $f_l(\underline{\eta}) = \sum_{s \in \mathcal{S}_l} \max(\mu_s, \min_{j \in \mathcal{L}_s} \eta_j) \quad \forall l \in \mathcal{L}$. For each value of $\underline{\eta}$, $f_l(\underline{\eta})$ is just the total flow in link l. Then by Theorem III.1, the max-min fair allocation can be obtained by solving

$$\underline{f}(\underline{\eta}) = \underline{C}$$

We now seek a distributed algorithm for solving the above vector equation. Intuitively, a method for deriving a distributed algorithm can be the following. Given a link whose capacity is not fully utilised, we increase the link control parameter (with each session adjusting its rate according to $r_s = \max(\mu_s, \min_{j \in \mathcal{L}_s} \eta_j)$ until the capacity is fully utilised. Similarly, given a link with total flow through it exceeding the available link capacity, we decrease the link control parameter to maintain feasibility. The important issue in such an approach is to choose the increase/decrease steps so that convergence to the max-min fair rates is ensured.

IV. Available Capacity for Elastic Sessions

In the MMF formulation presented we have assumed that the available capacity at each link is given and is constant. As discussed above, in a real network the elastic session transmissions are scheduled only when there are no guaranteed class (stream traffic) packets to send. The capacity available to elastic flows can thus be modeled as a stochastic process having variations that can be viewed as being over two time scales: (1) Short time scale variations due to the intrinsic rate variations in the stream traffic; and (2) Long time scale variations due to the entry/exit of large bandwidth stream type sessions.

In order to apply the MMF formulation in this scenario we are required to assign a fixed number to each link as its *available capacity* (i.e., C_l for link l) for the elastic sessions through it. The fair session rates are obtained as a max-min fair share of the C_l's in the links they span. A naive choice for C_l would be the mean of the stochastic available capacity. It is clear from elementary queueing theory that when the total input rate for a stochastic server is equal to the mean of the stochastic service rate, then queue lengths increase to infinity. A simple alternative would be to choose C_l to be a fraction (say 0.95) of the mean. The choice of such a scaling factor is not clear as the queue length process depends on the higher moments of the stochastic service process. Another approach is to choose C_l to be the total *input* rate that ensures that queue lengths are constrained. We shall study a large deviations theory based formulation for obtaining such a value in Section VI; this will be called Equivalent Service Capacity (ESC).

V. Distributed Algorithms

The distributed algorithms available in the literature for the computation of a rate allocation can, in general, be classified as one of the following: (1) algorithms that require information about session bottlenecks; and (2) algorithms that do not require information about session bottlenecks.

Algorithms of the first type essentially follow from the centralised algorithm (see Section II-B, [6], and [2]). Each link computes its LCP using the available capacity and information whether sessions are bottlenecked in it or at other links. Such algorithms thus require a link to monitor the individual flows through it, and may also require explicit exchange of information between links. For an example of such an algorithm see [9]. In algorithms of the second type, the links do not explicitly track the session bottlenecks. *The approach we pursue is for each link to update η_l (cf. Section III) without explicit communication between the computing nodes.*

A. Asynchronous Distributed Algorithms based on Stochastic Approximation

We have shown in Section III that the max-min fair rate allocation problem can be cast into one of solving for the solution of a vector equation in the link control parameters. Each link can only monitor its own available capacity and its total flow, and can update its own link control parameter. At each iteration k of our algorithm, link l has an available capacity estimate which is a random variable $C_l(k)$, such that $C_l(k) = C_l + \beta_l(k) + u_l(k)$, where C_l is a constant (the unknown available capacity defined according to one of the notions discussed in Section IV), $\beta_l(k)$ is the bias in the estimate (that goes to zero as $k \to \infty$), and $u_l(k)$ is zero mean and accounts for the short term variations and measurement "noise". The term $\beta_l(k)$ is required since the available capacity process need not start in its stationary distribution at the beginning of the sequence of iterations; this term goes to zero if the available capacity process is assumed to converge to a stationary regime[2]. Further, we assume that each link l can measure the total flow through it, yielding the measurement $\hat{f}_l(k) = f_l(k) + v_l(k)$, where $v_l(k)$ is zero mean measurement noise.

[1] $\mathcal{L}_s \setminus \tilde{\mathcal{L}}$ is not empty because every session has at least bottleneck link.

[2] Longer term large variations (e.g., due to arrivals and departures of stream sessions) are viewed as causing a change in the law of the available capacity process for elastic traffic. Such variations are considered separately in later sections.

Using the Stochastic Approximation Approach: Motivated by the discussion in Section III, an iterative algorithm is now required to drive $C_l - f_l(k)$ to zero, given only measurements $C_l(k) - \hat{f}_l(k) = C_l - f_l(k) + \beta_l(k) + \omega_l(k)$, where $\omega_l(k) = u_l(k) - v_l(k)$. Such problems of finding the root of an equation given noisy observations are handled using stochastic approximation. We write C_l^{max} as the total link bandwidth (or some other bound on C_l), and use the notation $[x]_a^b$ for $\min\{b, \max\{x, a\}\}$. The stochastic approximation update we propose to use is

$$\eta_l(k+1) = \left[\eta_l(k) + a_l(k)(C_l(k) - \hat{f}_l(k))\right]_0^{C_l^{max}} \quad (1)$$

The sequence of gains $a_l(k)$, for eacl l, is a decreasing sequence and satisfies the following properties for each link l $\sum_{k=1}^{\infty} a_l(k) = \infty$ $\sum_{k=1}^{\infty} a_l(k)^2 < \infty$. The tapering gains of the update formula would suppress the effect of the noise term $\omega_l(k)$. Interestingly, it has also been shown [8] that this property of the gains also relieves us of the requirement of synchrony in the updates, hence $f_l(k)$ can be of the form $f_l(k) = \sum_{s \in \mathcal{S}_l} \max(\mu_s, \min_{j \in \mathcal{L}_s} \eta_j(k - \tau_s^{lj}(k)))$, where $\tau_s^{lj}(k)$ accounts for the fact that the flow from session s during the measurement interval is based on the value of η_j that is $\tau_s^{lj}(k)$ iterations old.

The truncation of each iterate is used to keep it within a bounded set. Under certain technical assumptions, the algorithm can be proved to yield sessions rates that converge to the max-min fair rate corresponding to $C_l, l \in \mathcal{L}$. The proof is highly technical in nature and is available in [5]. A sketch of the proof is as follows. First note that it can be shown under certain technical assumptions that the effects of asynchrony and delays are suppressed by the tapering gains of the stochastic approximation procedure [8], hence reducing the problem to the synchronous case [3]. For the synchronous case the proof consists of two parts. As in the case of proofs of stochastic approximation algorithms [24], it is first shown that the evolution of the sequence of vector link control parameters, $\eta_l(k), l \in \mathcal{L}$, is asymptotically equivalent to the solution of the following ordinary differential equation.

$$\dot{\eta}_l(t) = \lim_{\Delta \to 0} \frac{[\eta_l(t) + \Delta(C_l - f_l(\eta(t))]_0^{C_l} - \eta_l(t)}{\Delta}$$

It can then be shown that the stable solution of this differential equation is the vector η that yields the desired max-min fair rates; see [5].

VI. Equivalent Service Capacity and its Estimation

Consider a queue with an infinite buffer with a constant input rate and a stochastic service process that is stationary and ergodic. The ESC is defined to be the maximum constant input rate into the queue such that the overflow probability for a given threshold is bounded above by a given value. Thus ESC is the dual of the effective bandwidth formulation for a source with stochastic rates (see [26]).

A. Equivalent Service Capacity

With reference to Figure 2, let λ denote the constant arrival rate, and let $D(t_1, t_2)$ denote the potential number of services[3] in an interval (t_1, t_2), with $D(t)$ denoting $D(0, t)$. Assuming the service process to be stationary, and that its log moment generating function exists, we write, for $\theta > 0$,

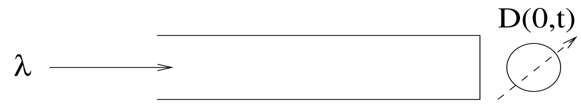

Fig. 2. An infinite buffer queue with a constant arrival rate and stochastic service rate.

$$\Gamma_{-D}(\theta) = \lim_{t \to \infty} \frac{1}{t} \log E \exp(-\theta D(t)).$$

Assuming that the stationary queue length process exists, for a given service process $\{D(t)\}$, and fixed arrival rate λ, let $Q(\lambda)$ denote the stationary queue length. We now state the main theorem without proof; the reader is referred to [26] for the proof in the discrete case. For the continuous case a proof can be developed as in [11].

Theorem VI.1: $\lambda < \frac{-\Gamma_{-D}(\theta)}{\theta} \iff \lim_{B \to \infty} \frac{1}{B} \ln(P(Q(\lambda) > B)) \leq -\theta$ $\square$

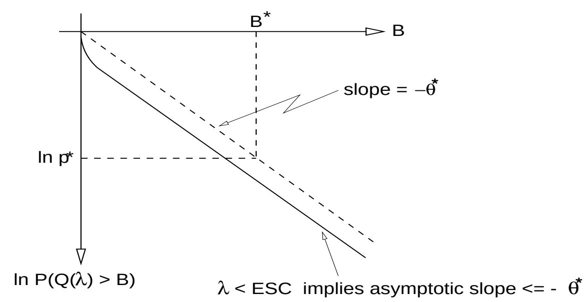

Fig. 3. Relation between $P(Q(\lambda) > B)$ and the QoS requirement (B^*, p^*) when $\lambda < \mathrm{ESC}$.

Let B^* denote the buffer threshold, and denote by p^* the desired upper bound on the probability $P(Q(\lambda) > B^*)$; we will denote this QoS requirement by the tuple (B^*, p^*). Define $\theta^* = \frac{-\ln p^*}{B^*}$. We infer from Theorem VI.1 (see also Figure 3) that, for large B^*,

$$\ln P(Q(\lambda) > B^*) < \ln p^* \quad \text{if} \quad \lim_{\mathrm{B} \to \infty} \ln(\mathrm{P(Q(\lambda) > B)}) \leq -\theta^* \quad \text{iff} \quad \lambda < \frac{-\Gamma_{-\mathrm{D}}(\theta^*)}{\theta^*}$$

Definition VI.1: The *Equivalent Service Capacity* of the service process $\{D(t)\}$, for the QoS (B^*, p^*), is the arrival rate λ^* of a deterministic process such that the asymptotic slope of $\ln(P(Q(\lambda^*) > B)$ is $-\theta^*$.

From the foregoing, it is clear that $\frac{-\Gamma_{-D}(\theta^*)}{\theta^*}$ is the ESC for the service process $\{D(t)\}$ for the QoS (B^*, p^*).

B. Estimating the ESC using a Virtual Buffer

Define the function $\psi(\lambda)$ by

$$(\psi(\lambda))^{-1} = \lim_{B \to \infty} \frac{-1}{B} \ln(P(Q(\lambda) > B))$$

It follows that the ESC is the solution of the equation $\psi(\lambda) = \frac{1}{\theta^*} =: \psi^*$. Denote the mean of the service rate process to be $\overline{\mu}$, and let $\underline{\mu}$ denote the minimum service rate. It follows that $\psi(\lambda) = 0$ if $\lambda \leq \underline{\mu}$, and $\psi(\lambda) \to \infty$ as $\lambda \to \overline{\mu}$; see Figure 4.

Assumption VI.1: $\psi(\lambda)$ is continuous, λ^* exists and is unique.

[3]The best-effort queue will typically be the lowest priority queue at the output of a switch; $D(t_1, t_2)$ then represents the number of cell times for which the

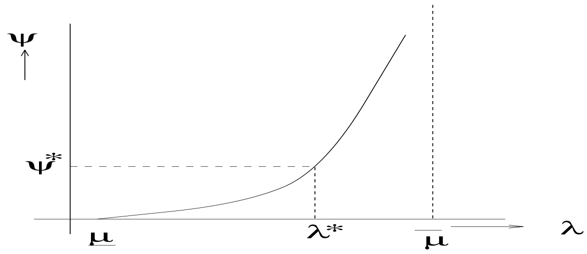

Fig. 4. A sketch of $\psi(\lambda)$ as a function of λ, showing the desired QoS $\psi^* = \frac{1}{\theta^*}$, and the corresponding ESC λ^*.

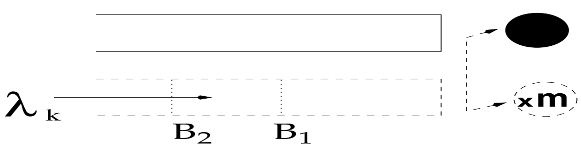

Fig. 5. A virtual buffer is associated with the best-effort output queue whose ESC has to be estimated. The available service applied to the virtual buffer is scaled by m; see text.

As in [12], it can be shown that Assumption VI.1 holds if the process $\{D(t)\}$ is a Markov modulated service rate process.

An estimation procedure motivated by the above discussion would be as follows. We take recourse to the virtual buffer technique proposed in [10]. Consider a best-effort output queue associated with a link at a node. A virtual buffer (simply a counter) is associated with this output queue; see Figure 5. The ESC estimate is updated periodically. In the kth update interval the virtual buffer is fed by a cell arrival rate equal to the current ESC estimate λ_k. The virtual buffer is served by the available capacity service process. We approximate $\psi(\lambda_k)$ by making measurements on the virtual buffer occupancy process. We then iterate λ_k to λ_{k+1} using the error between $\psi(\lambda_k)$ and ψ^*.

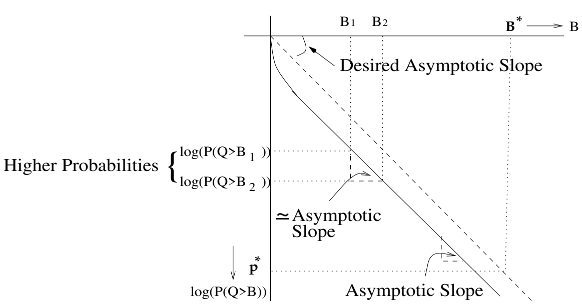

Fig. 6. The slope of the asymptote can be approximated by considering two points on the asymptote.

Measuring $\psi(\lambda)$: In Figure 6 we illustrate the motivation for a two threshold based method for obtaining an estimate of $\psi(\lambda)$. Letting $Q(\lambda)$ denote the stationary queue length in the virtual buffer, we assume that the plot of $\log P(Q(\lambda) > B)$ vs. B is asymptotically linear. Hence the ESC is approximated by that λ^* for which the curve $\log P(Q(\lambda^*) > B)$ vs. B has an asymptotic slope $1/\psi^*$. We note here that a correction to account for the fact that the asymptote is affine and not linear can also be incorporated by making measurements on a zero size virtual buffer, but we have not done this in the work reported here. We consider two smaller thresholds B_1 and B_2 (see Figure 5)that are "on the asymptote" (see Figure 6) and use the following approximation.

$$\text{Asymptotic Slope} \simeq \frac{\log \mathrm{P}(\mathrm{Q}(\lambda) > \mathrm{B}_2) - \log \mathrm{P}(\mathrm{Q}(\lambda) > \mathrm{B}_1)}{\mathrm{B}_2 - \mathrm{B}_1} \tag{2}$$

Since B_1 and B_2 are smaller buffer thresholds than B^*, accurate estimates of $P(Q(\lambda) > B_1)$ and $P(Q(\lambda) > B_2)$ can be obtained from measurements over shorter intervals.

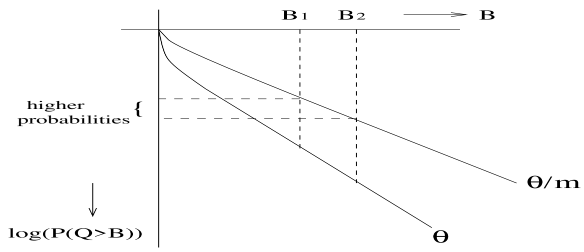

Fig. 7. By the use of a speed-up factor the probability of occurrence of the required event increases and hence the estimation from measurements is more accurate

A Speed-up Technique for Measuring $\psi(\lambda)$: We use a simple scaling property of the log-moment generating function to derive a new technique for reducing the length of the measurement intervals. For $m > 1$, we let $\hat{D}(0,t) = mD(0,t)$, i.e., $\hat{D}(0,t)$ denotes the maximum number of services in $(0,t)$ for a service process that is m times faster than $D(0,t)$. Note that

$$\Gamma_{-\hat{D}}(\theta/m) = \Gamma_{-D}(\theta) \quad \Rightarrow \quad \frac{\Gamma_{-\hat{D}}(\theta/m)}{\theta/m} = m\frac{\Gamma_{-D}(\theta)}{\theta} \tag{3}$$

Equation (3) states that the ESC for the $\hat{D}$ service process with QoS requirement θ/m is m times the ESC of the D service process with QoS requirement θ. Note that since $\theta/m < \theta$, the events that we are searching for occur with higher probability (see Figure 7). Thus the interval for which the test rate has to be applied to obtain a fairly accurate estimate of this probability is reduced. Also note that a better estimate of the asymptotic slope is obtained from larger values of the buffer thresholds B_1 and B_2. The use of the speed-up technique enables us to choose larger values of buffer thresholds when a larger speed-up factor m is used. In the implementation, this scaling can be approximated by decreasing the count of the virtual buffer by m at the potential departure epochs from the actual queue; see Figure 5.

B.1 Stochastic Approximation Algorithm for Estimating the ESC

For clarity, we present here the algorithm for a speed-up $m = 1$. The algorithm for higher speed-up factors can be similarly derived (see Algorithm VIII.1). Define $h(\lambda)$ by $h(\lambda) = \psi(\lambda)/\psi^*$. Note that we are searching for the root of $h(\lambda) - 1 = 0$. Let $\hat{\psi}(\lambda_k)$ denote the estimate of $\psi(\lambda_k)$ obtained from the measurement at the virtual buffer (over a measurement interval) when the input test rate is λ_k (see Equation 2). Further, let $g_k = \hat{\psi}(\lambda_k)/\psi^*$. Thus g_k is a "noisy" observation of $h(\lambda_k)$. We assume that the noise (denoted by ω_k) is additive and that it satisfies some properties. Write g_k as follows $g_k = h(\lambda_k) + \omega_k$. Define the $\sigma-$field $\mathcal{F}_k$ by $\mathcal{F}_k = \sigma\{\lambda_0, \ldots, \lambda_k, \omega_0, \omega_1, \ldots, \omega_{k-1}\}$.

Assumption VI.2: For all k, $E(\omega_k \mid \mathcal{F}_k) = 0$, and $E(\omega_k^2 \mid \mathcal{F}_k)$ is bounded uniformly in k.

link is free to serve cells from the best-effort queue.

The algorithm we use behaves as if the function $h(\lambda)$ is linear with unknown slope ν, i.e., $h(\lambda) = \nu\lambda$. At the k^{th} iteration we form a least squares estimate of ν (call it ν_k) and set $\lambda_k = 1/\nu_k$. The procedure is derived from the recursive form of the least squares estimate of ν_k, i.e.,

$$\nu_k = \nu_{k-1} + \frac{\lambda_{k-1}}{\sum_{i=0}^{k-1} \lambda_i^2}(g_k - 1) \tag{4}$$

In order to keep the value of λ_k computed in a bounded set we truncate ν_k to lie in a set $[\underline{\nu}, \overline{\nu}]$. Equation 4 is replaced by

$$\nu_k = \left[\nu_{k-1} + \frac{\lambda_{k-1}}{\sum_{i=0}^{k-1} \lambda_i^2}(g_k - 1)\right]_{\underline{\nu}}^{\overline{\nu}} \tag{5}$$

As pointed out in [23], the algorithm is of stochastic approximation type, as the value of ν_k is computed by adding to ν_{k-1} an error term scaled by a decreasing gain $\lambda_{k-1}/(\sum_{i=0}^{k-1} \lambda_i^2)$ that goes to zero as k increases. We now state the convergence theorem. This theorem is very similar to Theorem 5.1 of [23], and the proof of the theorem is a direct application of Theorem 5.3.1 of [24]; we refer the reader to [23] for details of the proof.

Theorem VI.2: Given Assumption VI.1 and Assumption VI.2, we have, with probability one, $\lambda_k \to \lambda^*$ as $k \to \infty$.

C. Using ESC in the LCP Update Algorithm

In order to use the ESC notion in the MMF problem with stochastic available capacities, we simply take $C_l = \lambda_l^*$ in Equation 1. We estimate the ESC, λ^*, as described above. Theorem VI.2 then shows that the term $\beta_l(k)$ in Equation 1 does converge to zero, as is required for convergence of the link control parameters. We note here that in applying the ESC approach to the rate allocation problem, we have made the following assumptions in order obtain a tractable analysis and an implementable algorithm.

- When we formulated the notion of ESC of a link in Section VI-A, we assumed that the arrival process and the stochastic service process were independent. In a network, the stream sessions will often span several links, and hence the available capacity processes at these links will be dependent. If elastic sessions from one such link flow into the other, then the elastic flow in the second link and the available capacity process at that link are dependent. In high-speed networks, however, the number of stream sessions will be large and we assume that there will be enough mixing of stream sessions from different paths, at a link, so as to render the dependence weak.
- In formulating the notion of ESC, we had taken the flow of elastic sessions into a link to be a deterministic flow (i.e., not stochastic). While this is valid at the edge of the network, since avialable capacities are random, the elastic flows into nodes in the core of the network will no longer be deterministic. However, the flow into each link in the core will be the superposition of a large number of small independent (see previous bullet) flows; hence the variability relative to the flow rate can be reasonably assumed to be small.

VII. Implementation Issues

From the discussions in Section V it is clear that if the available capacity variations on each link are stationary, and hence the ESC of each link is fixed, then the rates will converge to the MMF rates, in spite of propagation delays and asynchrony. The fact that such convergence takes place during a stationary regime will be evident from the simulation results we present. However, in a practical network, the stationarity of available capacity assumption will be violated if there is an entry or exit of a large capacity CBR/VBR session. Such a change will result in a change in the ESCs of some links, and hence a change in the MMF rates. In such a situation the decreasing gains of the stochastic approximation algorithm will result in a sluggish response. We need to take some special actions to take care of this possibility. We address this problem by resetting/changing the stochastic approximation gain when large changes in available capacity take place. When the algorithm is reset we also need to concern ourselves with the choice of an initial gain. Finally, in order to conform to standards (we assume the ATM context) we need to adapt switch updates to the source end system behavior.

A. Adapting to Large Changes in Available Capacity

If the available capacity of some link reduces (owing to the entry of a session) the slow response of the algorithm would result in large queues building up. When the available capacity increases, the slow response would cause inefficient utilisation. We discuss the following two cases: (1) Adapting to local changes in available capacity, i.e., the available capacity for the link that the given switch controls changes due to the entry/exit of CBR/VBR sessions. (2) Adapting to remote changes in available capacity, i.e., the available capacity changes at a remote link that is not controlled by the given switch, but one that has sessions in common with the local link.

A.1 Adapting to Local Changes in Available Capacity

We consider a sudden change in available capacity for the ABR class owing to the entry or exit of a large bandwidth CBR/VBR session. Every CBR/VBR session must go through a bandwidth reservation phase at each switch on its path. Hence when a CBR/VBR connection arrives or terminates, every switch on its path knows about it. Our proposal consists of simply using the admission of a new CBR/VBR flow as a trigger for resetting the gain of the algorithm. At such epochs the gain of the algorithm is reset to the initial large value (i.e., $a_l(k)$ is increased). We have reported simulations *without* such resetting of gain in our paper [4].

Large queue length build-ups can be avoided in large round trip time (rtt) networks if the gains are increased *anticipatively*. During the connection setup phase, for a CBR/VBR session, a switch is aware of the entry of a session and the amount of bandwidth allocated to it. Thus the stochastic approximation gain can be increased a few round-trip times before the new session begins to send. Also, the bandwidth allocated to the new session is subtracted from the available capacity estimate. Hence the link control parameter begins to fall before the new session begins transmissions, limiting the elastic flow rates, and preventing large queues from building up when the new stream session actually begins transmission.

A.2 Adapting to Remote Changes in Available Capacity

Note that the max-min fair solution is a global solution across the network. Hence a change in capacity in one link can cause the fair flows in other links to change drastically. An increase in capacity at a link may increase the rate of a session. This would increase the total flow in another one of this session's links, which may be a bottleneck to other sessions, and thus increase the queue length built up at this link. We suggest the use of *queue length thresholds* to trigger an increase in the stochastic approximation gain in such situations. The increased gain of the stochastic approximation algorithm coupled with the negative difference between the available capacity and the input flow (see Equation 1) forces a quick reduction of the LCP [4].

Similarly, a decrease in capacity at a remote link may decrease the max-min rate of a session, thus decreasing the total flow through another link that this session passes through, and reducing the utilisation of available capacity at this latter link. We suggest the following "exponential forgetting" based low utilisation detection algorithm. When, the low utilisation detection algorithm triggers an increase in the stochastic approximation gain, the positive difference between available capacity and the input flow will cause a quick increase in the LCP [4].

Algorithm VII.1:
The algorithm has an averaging parameter h, and an available capacity Utilisation Threshold. At each update epoch do:

1. Compute the available capacity utilisation in the previous interval, i.e., $\rho =$ (Total time in the interval during which the ABR queue had at least one customer) / (Length of the update interval).
2. If $(\rho >$ Utilisation Threshold), set $\beta = \rho$, else set $\beta = h\beta + (1-h)\rho$.
3. If $(\beta <$ Utilisation Threshold), increase the gain of the stochastic approximation algorithm.

In the above algorithm, the choice of a Utilisation Threshold is an important issue when we use the ESC. Noting that ESC is less than the mean available capacity, the achievable utilisation of the available capacity is (ESC / Mean Available Capacity). It is clear that the Utilisation Threshold should be sufficiently less than this achievable utilisation to not cause false triggering of the above low utilisation detection algorithm. Since the ESC depends on the higher order statistics of the available capacity process, the link utilisation for a given mean available capacity can differ significantly depending on the higher moments of the available capacity process. Since ESC and mean available capacity can both be measured, however, a rule of thumb could be to set the Utilisation Threshold 0.1 less than the achievable utilisation as defined above. In our simulations we have found this rule of thumb to be adequate.

B. *Choice of an Initial Gain*

When the round trip time (rtt) is large compared to the update intervals at the links, several link updates take place in the time between a given update and the corresponding response from a session. Hence, if the value of a link's LCP is lower than the max-min value the multiple updates could force it to increase to a value much higher than the max-min value, and when the sessions send at such a high value, multiple updates would force the LCP to a value smaller than the max-min rate. Thus, the effect of these multiple updates is that there are oscillatory excursions in the computed LCPs when the initial gain is not sufficiently small. These potentially large oscillations in the LCP, which in turn could cause large oscillations in input rate, would trigger the mechanisms discussed in Section VII-A.2 resetting the gain of the algorithm often. The LCPs would then never converge to their max-min values. In order to overcome this problem, we choose the initial gain based on the following heuristic.

Let n_l denote the number of sessions through link l. Note that in the ATM/ABR context the number of sessions is known at each switch; further, recall that we are assuming long-lived elastic flows, as would be the case with edge-to-edge elastic virtual paths carrying an aggregate of short lived elastic sessions (see Section I). Hence, n_l can be assumed to be known. Let MRT denote the maximum round trip time among the sessions through link l, and let N_l denote the maximum number of link control parameter updates that can take place in an interval of length MRT. The initial gain is chosen to be $1/n_l N_l$. An initial gain less than $1/n_l N_l$ can also be chosen, hence even if the parameters n_l and N_l cannot be exactly determined upper bounds on them can be used (this would address the situation in which the session activity is time varying).

C. *Adapting to Source End System Behavior*

In the derivation of the stochastic approximation algorithm we assumed that the source immediately changes its rate to the rate fed back from the network. However, in the source behaviour specifications of the ATM Forum, the sources are required to increase the rate by a small fixed amount whenever a Resource Management (RM) cell with an Explicit Rate (ER) field greater than the source rate is received. The ER field indication is a dynamic upper bound on the source rate. Note that such small fixed increments in the session rates could result in fast growth of the LCP's if Equation 1 is implemented directly. We are thus required to bound the increment of the LCP so that the difference between rate increment at the session sources and the LCP is small. Let T denotes the update interval for the given switch. For a session s through the given switch, let N_s denote the number of RM Cells received by the source in the time T, and A_s denote the additive increment for a source s, then the maximum increment in an update interval at the source s (denoted by I_s) is $I_s = N_s \times A_s$.

Since the RM cells are sent "in-rate", note that N_s and hence I_s are proportional to the sending rate of the session. In order to prevent the link control parameter from increasing uncontrollably due to the slow increment at the sources, we need to ensure that if the link control parameter increases at an update, the increment is bounded above by a quantity that is close to the minimum of the I_s value of the sessions through the link. To obtain an estimate of the minimum I_s among the sessions s in through the given link, we assume that all the sessions that pass through the given link have the same fixed additive increment, i.e., $A_s = A$ for all sessions, and that A is known at the switches. Let NRM denote the number of data cells transmitted per RM cell, then the following can be used as an estimate of the number of RM cells received by a typical source (it is exact if the rates of all sessions through the links is equal): (Total Input Rate $\times T$)/(Number of Sessions $\times$ NRM). The LCP increment at the link is then bounded above by $I =$

$A \times$ (Total Input Rate $\times T$)/(Number of Sessions $\times$ NRM).

Note that as the input rate increases, the bound on the increment also increases. Hence the modified update equation (see Equation 1) is

$$\eta_l(k+1) = \min\left\{\eta_l(k) + I,\, \left[\eta_l(k) + \alpha_l(k)(C_l(k) - \hat{f}_l(k))\right]_0^{C_l^{max}}\right\}$$

VIII. A Simulation Study

A. Description of the Simulation Setup

A.1 Simulation Software

We have used the NIST ATM simulator for the experiments reported here. This simulation package provides users with modules for switches, broadband terminal equipment and links. A graphical user interface is also provided to enable users to build a given network topology and witness the progress of the simulation. The implementation of the modules closely follows the guidelines provided by the ATM Forum. The source code of the simulator is available via anonymous FTP at `ftp://isdn.csnl.nist.gov/atm-sim`. We extended the switch module introduced here to incorporate the stochastic approximation algorithm, the rate estimation algorithms and the other heuristic methods proposed in Section VII.

The simulated network is shown in Figure 8. It consists of 4 bottleneck links and six sessions. All the bottleneck links (labeled (1) to (4) in Figure 8) have a maximum capacity of 150Mb/s. All the flows are from "left to right". Hence switch SWi controls the flow through link i.

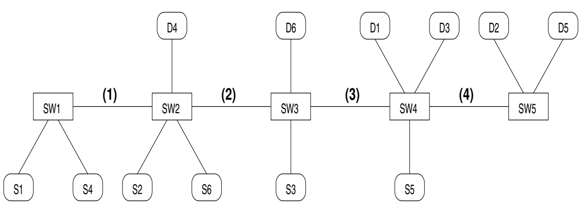

Fig. 8. The simulated network.

A.2 Simulating Stochastic Available Capacity

We use a Markov model for the service capacity available to the ABR sessions at each link. A Markov model has been chosen since it is easy to compute the exact ESC for such a model (this involves the computation of the dominant eigenvalue of a certain matrix; see, for example, [20]). We associate a two state $(0, 1)$ Markov chain with each controlled link in the network (Links 1 to 4 in Figure 8). Time at the switches is (conceptually) slotted into intervals whose size is the cell transmission time (at full rate i.e., 150Mb/s) on the link. The transitions of the Markov chain take place at these slot boundaries. The state 0 means that during that cell time a link is servicing higher priority traffic. A cell in the ABR queue is served in those slots for which the associated Markov chain is in state 1. We denote the $i \to i$ transition probability by p_i for $i \in \{0, 1\}$.

A.3 Simulated Scenarios

The update interval at all the switches for all the experiments is $1ms$. The simulations are performed for the following scenarios.

1. LAN Environment: The link distances are chosen to be 2km for each link. Thus the update interval is much larger than the round trip times. The execution in this case is very much like the synchronous case.
2. WAN1 Environment: The link distances are chosen to be 65km for each link, hence the maximum round trip among the sessions at any link is slightly less than twice the update interval.
3. WAN2 Environment: The link distances are very large (1000 km). The maximum round trip time among the sessions at any link is 30 times the update interval.

A.4 Available Capacity Estimation Algorithms

In each of the three scenarios described in Section VIII-A.3, 2 experiments are performed which differ in the notion of available capacity that is used, and the way available capacity is estimated.
1. Scaled mean capacity: The available capacity is estimated as 0.95 times the available capacity measured in the just previous inter update interval. This algorithm will be denoted by Alg. A.
2. ESC: The ESC is estimated using the virtual buffer based algorithm (cf. Section VI-B). This algorithm will be denoted by Alg. B.

We now describe the implementation of the ESC estimation algorithm. Some notation is required for the description. Let $k, k = 1, 2, \ldots$ index the update intervals, and let λ_k denote the sequence of rates applied to the virtual buffer. The speed-up factor is denoted by m. δ_v is a small positive number. Let B_1 and B_2 denote the two thresholds for the virtual buffer, and let $C_{ESC}(k)$ denote the k^{th} estimate of the ESC. The QoS requirement is an overflow probability less than P^* for a buffer threshold B^*. Define $\psi^* = B^*/(\log P^*)$. The following algorithm yields the estimate $C_{ESC}(k)$.

Algorithm VIII.1:

1 Compute $x_i, i = 1, 2$, as $x_i = \dfrac{\text{Amount of time the virtual buffer exceeds } B_i}{\text{Length of the update interval}}$

2 If $(x_2 < p_{th})$ (see Remarks below for an explanation of this step) $\lambda_k = \lambda_{k-1} + m\delta_v$

else $g_k = -\frac{B_2 - B_1}{m\psi^*(\log x_2 - \log x_1)}$, and $\lambda_k = \left(\left[\frac{1}{\lambda_{k-1}} + \frac{\lambda_{k-1}}{\sum_{i=0}^{k-1}\lambda_i^2}(g_k - 1)\right]_{\underline{\nu}}^{\overline{\nu}}\right)^{-1}$

3 $C_{ESC}(k) = \lambda_k/m$

Remarks:

1) At the beginning of an update interval, the queue length of the virtual buffer is set to zero.
2) In Step 1 of the algorithm, the fractions of time the virtual buffer queue length is greater than B_1 and B_2 are computed.
3) In Step 2, we first check if the fraction of time the queue length is greater than B_2 is smaller than a set threshold. If it is smaller, then we assume that the current estimate is much smaller than the actual ESC, so we simply increase the current estimate of the ESC by a fixed increment δ_v. This step enables a quick initial increase to a close range of the ESC, and avoids potential overflows when taking logarithms. If the fraction of time the queue length is greater than B_2 is greater than the set threshold then the stochastic approximation iteration is performed.
4) The speed-up m of the ABR service rate at the virtual buffer is implemented as follows. Every time a cell can be potentially served in the actual ABR queue the count of the virtual buffer is reduced by min(m, current count of virtual buffer).

A.5 Implementation of the LCP Update Algorithm

Let N_l denote the maximum number of link control parameter updates that can take place in an interval of length MRT (cf. Section VII-B). NRM = Number of data cells transmitted per RM cell, AIR = fixed additive increment at the sources, δ = maximum rate increment, n_l = number of sessions, f_l = estimate of the input rate, C_l = estimate of the available capacity, α_l = stochastic approximation gain used for the update, C_l^{max} = maximum permissible value for η_l.

Algorithm VIII.2:

At each link l, at each update, do the following

1. Estimate the input rate f_l into link l, i.e., f_l = (number of cells that arrived in the previous inter-update interval) / (length of inter-update interval).
2. Compute the maximum increment and the bound on η_l based on the maximum increment, i.e., $\delta = AIR \times (f_l/(n_l \times NRM))$, and $\eta_l' = \eta_l + \delta$
3. Update the estimate of the available capacity C_l.
4. If a large long term change in available capacity has taken place reset the gain: $\alpha_l = \alpha_l^0$.
5. If the queue length is greater than the preset queue threshold, increase the gain: $\alpha_l = \alpha_l^0$.
6. If low link utilisation has been detected, increase the gain: $\alpha_l = \alpha_l^0$.
7. Compute the (temporary) new value of the link control parameter η_l: $\eta_l \leftarrow \left[\eta_l + \alpha_l \left(\frac{C_l - f_l}{n_l N_l}\right)\right]_0^{C_l^{max}}$
8. Apply the bound on the increment of η_l and modify the stochastic approximation gain. If $\eta_l > \eta_l'$ then $\eta_l = \eta_l'$ and do not change α_l, else retain η_l computed in Step 7, but reduce the stochastic approximation gain α_l (see Equation 6 later in the paper).

Remarks: In Step 1, the current input rate is estimated. In Step 2, a bound on the increment is computed to account for the additive increment mechanism in ATM networks (cf. Section VII-C). In Steps 4 to 6 the gain of the algorithm is reset if the available capacity at a switch has changed, (cf. Section VII-A.1), or if any one of the remote capacity change mechanisms (cf. Section VII-A.2) have indicated a need to reset the gains. In Step 7, we have slightly modified the update step so that the increment term $(C_l - f_l)$ is divided by $n_l N_1$. Note that with $\alpha_l^0 = 1$, the initial gain will be exactly the amount suggested in Section VII-B. However this factor does not modify the structure of the algorithm. In Step 8 of the algorithm, we decrease the stochastic approximation gain only if the update was not limited by the bound on the increment. This is done to prevent the gain from decreasing to a small value when a large number of successive increments are limited by the bound on the increment. Note that as the input rate increases, the bound on the increment also increases, so the decrement on the stochastic approximation gain will in general be constrained only during the initial increment phase. Also note that the gain is always decremented when an update *decreases* the link control parameter.

A.6 Simulation Parameters

Queue Threshold for Gain Reset: (cf. Section VII-A.2) 200 cells at all switches.

Low Utilisation Threshold: Algorithm VII.1 is used for detecting low utilisation for the case when the available capacity at a remote link is reduced. The utilisation threshold used at SW1, SW3, and SW4 is 0.8, and 0.7 at SW2. As discussed in Section VII-A.2 the utilisation threshold should be less than the achievable utilisation of the available capacity. The available capacity of Link 2 changes during the simulation, and our software did not include a facility to change the value of this parameter during the simulation run, hence a lower value of utilisation threshold was used for SW2 throughout the simulation. Note from Table II that the achievable utilisation of Link 2 during the low rate phase is about 0.83 ($= \frac{45.2}{\frac{51.82}{0.95}}$), and hence a utilisation threshold of 0.8 will not be appropriate in this period.

Initial Gain and Decrement: The initial value of the stochastic approximation gain is computed using the heuristic given in Section VII-B. The values are given in Table I. Note that the number of sessions in Links 1,2,3 and 4 are 2,3,3, and 2 respectively. The update interval (T1) is 1ms on all the links; the maximum round trip propagation delays (T2) are 30μs, 2ms, and 30ms for the LAN, WAN1, and WAN2 environments, respectively. Table I also shows the computation of the initial gains.

Link	Initial Gain $1/(n_l \times \lceil T2/T1 \rceil)$		
	LAN	WAN1	WAN2
link 1	1/2	1/4	1/60
link 2	1/3	1/6	1/90
link 3	1/3	1/6	1/90
link 4	1/2	1/4	1/60

TABLE I
INITIAL GAINS

The successive decrements in the gains are carried out using the formula

$$\alpha \leftarrow \frac{1}{(1/\alpha) + 0.1} = \frac{\alpha}{1 + 0.1\alpha} \qquad (6)$$

Note that the sequence of gains generated by the decrement procedure given in Equation 6 satisfy the assumptions on the stochastic approximation gains given in Section V.

Session MCRs: The session MCRs are $(0, 0, 0, 0, 0, 20)$ for sessions $(1, 2, 3, 4, 5, 6)$ respectively.

Available Capacity and Link Details: In Table II, we give the details pertaining to link distances and the available capacity for ABR traffic. The transition probabilities of the DTMC that modulates the available capacity for the ABR traffic are given. In the estimation algorithm using a short term average, a utilisation factor of 0.95 is used.

The Equivalent Service Capacity (ESC) is computed for an overflow probability (P^*) of 10^{-3} for a threshold (B^*) of 10 cells; p_{th} is set to 10^{-6}, m is set to 5, and $\delta_v = 1$. The initial values of λ and the thresholds on ν scaled down by the speed up factor are also given in Table II. The choice of the initial value of λ is not very critical to the algorithm. If the initial value is too small, the probability of crossing the higher virtual buffer threshold is small. λ is then simply incremented by a fixed value (cf. Algorithm VIII.1, step 3). If the initial value is large, the

Switch	Available Capacity and ESC Estimator Parameters								Link Distances	
		DTMC								
Link	Time	p_0	p_1	$0.95\times$ Mean	ESC	Initial λ	$1/\overline{\nu}$	$1/\underline{\nu}$	Type	Distance
SW1 link 1	0 - 3sec	0.2	0.9	126.67	125.18	50	80	130	LAN	1 km
									WAN1	65 km
									WAN2	1000 km
SW2 link 2	0 - 1sec	0.2	0.9	126.67	125.18	50	80	130	LAN	1 km
	1 - 2sec	0.6	0.3	51.82	45.20	50	30	60	WAN1	65 km
	2 - 3sec	0.2	0.9	126.67	125.18	50	80	130	WAN2	1000 km
SW3 link 3	0 - 3sec	0.2	0.9	126.67	125.18	50	80	130	LAN	1 km
									WAN1	65 km
									WAN2	1000 km
SW4 link 4	0 - 3sec	0.2	0.9	126.67	125.18	50	80	130	LAN	1 km
									WAN1	65 km
									WAN2	1000 km

TABLE II

AVAILABLE CAPACITY MODEL PARAMETERS, ESC ALGORITHM PARAMETERS, AND LINK DISTANCES

algorithm will force the first iterate to the minimum permissible value.

We now remark on $\overline{\nu}$ and $\underline{\nu}$. Note that $1/\overline{\nu}$ and $1/\underline{\nu}$ are, respectively, lower and upper bounds to the ESC, λ. A conservative choice of the lower bound is used. Note that such a choice of the lower bound on λ does not significantly affect the rate of convergence of the algorithm since λ is simply incremented when it is much lower than the ESC (cf. Algorithm VIII.1, Step 3). The upper bound on λ may be chosen to be close to the mean available capacity. In the ATM context, the mean available capacity may be obtained by subtracting link rate from the mean bandwith reserved by the stream sessions; alternatively the mean available capacity can be measured. However, our studies indicate that the performance of the algorithm is not significantly affected by choosing a value slightly larger than the mean available capacity.

Remark on Choice of Parameters: The main objective of this simulation study is to demonstrate the efficacy of the algorithms presented here, and to observe how well the results obtained match the theoretically predicted values. To keep the simulation execution time small, however, and also to have sufficiently large number of updates within this time we chose an update interval of 1ms. The Markov chains chosen would have to make many transitions between states within this 1ms interval for fairly accurate results in the estimation algorithm. However, such Markov chains have ESC's close to the mean if the QoS requirement is not very stringent. In order to demonstrate the effect of a large difference between the ESC of the stochastic capacity and mean of the capacity, we have chosen a stringent QoS requirement, i.e., a threshold crossing probability of 10^{-3} for a buffer of size 10.

B. Simulation Results and Discussion

Six simulation experiments were performed. Each simulation run was for 3 seconds.For each of the capacity estimation algorithms Alg. A and Alg. B (see Section VIII-A.4), a set of three experiments were performed using the environments LAN1, WAN1 and WAN2 (see Section VIII-A.3). We present the plots in four figures (Figures 9, 10, 11, and 12), each with 8 plots, organised into 2 rows and 4 columns.

1. In Figure 9 we show the time series of the available capacity estimate at each link. Since the estimate depends only on the estimation algorithm used and not on the link distances, we only need to present one set of plots for the three network scenarios (LAN, WAN1, WAN2). We also plot the analytically obtained value of the available capacity. For Alg. A, this is simply the mean obtained from the DTMC scaled down by 0.95. For Alg. B, the analytically obtained ESC, for a probability of 10^{-3} for the buffer size exceeding 10 cells, is plotted.
2. In Figures 10, 11 and 12 we plot, at SW2 and SW3, the time series of the link control parameters obtained from our algorithm, and their ideal values corresponding to the analytically computed values of available link capacity (i.e., either 0.95 times mean capacity, or the ESC), and also the time series of the buffer occupancies. We only plot the epochs where the buffer exceeded 10 cells at SW2 and SW3. For details of which plot corresponds to which quantity, see the captions.

Observations: The following is a summary of our observations from the experiments.

1. The second row of Figure 9 shows that the ESC estimation approach works quite well. Notice that while there are transient estimation errors at epochs where the ESC changes, the estimate converges to the analytically computed ESC.
2. It is evident from the dotted plots in the first and second columns of Figures 10, 11 and 12, that even though the link capacity process is changed for link 2 only (between 1 sec and 2 sec), this change affects the link control parameters of all the links.
3. Note from the first and third column plots in Figures 10, 11 and 12 that there are slow increment phases of the LCP at all links due to the constraints applied on the increment of the link control parameters (see Section VII-C).
4. In the intervals where the available capacity is “stationary” (i.e., where the parameters of the Markov chain modulating the available capacity are fixed) the LCPs (and hence the session rates) track the analytically obtained values closely. As would

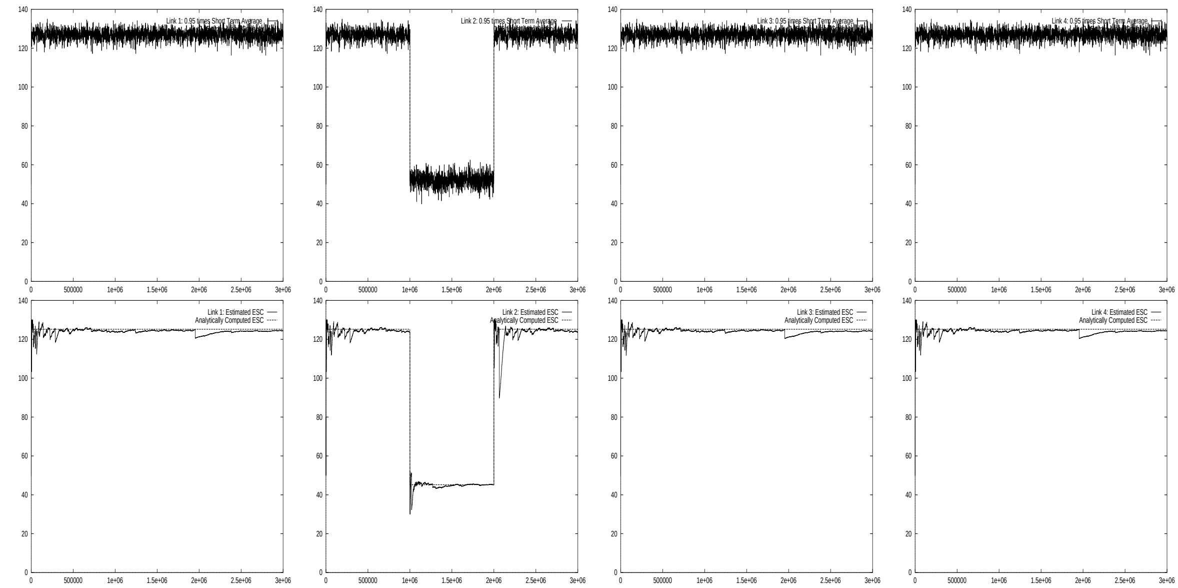

Fig. 9. Plots of available capacity estimates obtained using Algorithm A (i.e., 0.95 times mean; top row of plots), and Algorithm B (i.e., ESC; bottom row), and their analytically obtained values. For each plot: x-axis: time in microseconds; y-axis: available capacity in Mb/s. Each column of plots (cols 1 to 4) is for the corresponding link (links 1 to 4).

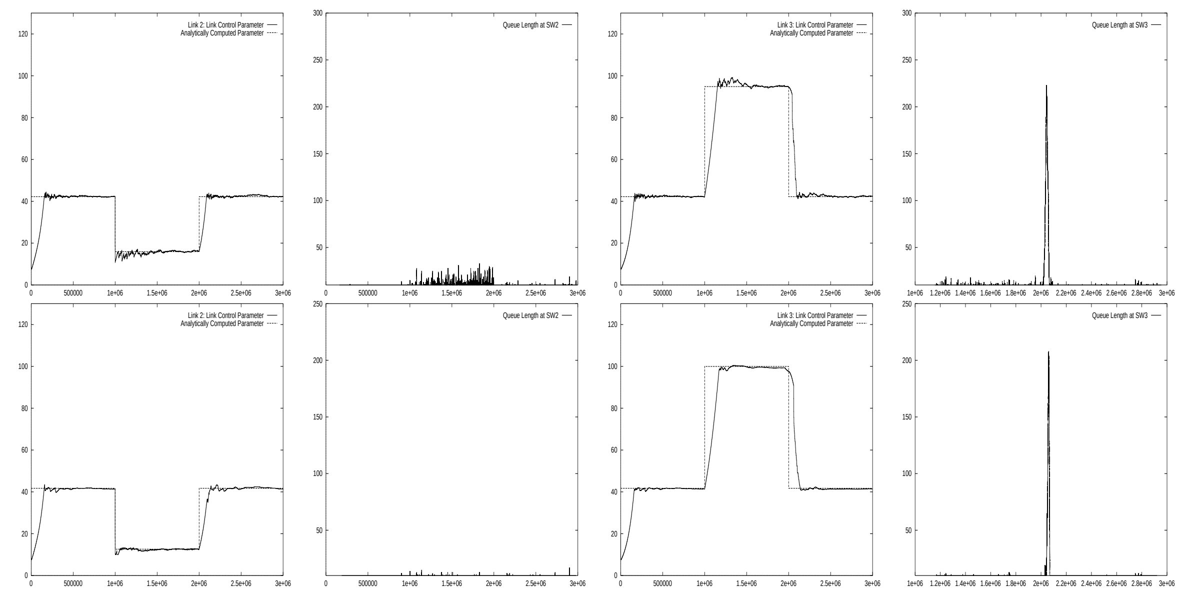

Fig. 10. LAN environment: Plots of the link control parameter and queue lengths at link 2 and link 3. Row1 and Row2 correspond to 0.95 $\times$ mean, and ESC respectively. Col1 and Col3: LCPs from the simulation and analytically obtained values at link 2, link 3 respectively; x-axis: time in microseconds, y-axis: LCP in Mb/s. Col2 and Col4: queue length at link 2, link 3 respectively; x-axis: time in microseconds, y-axis: queue length in cells.

be expected, the transient deviations from the ideal values last longer the larger the propogation delay.

5. Note that there is very small queue build up at SW2 in spite of the large capacity drop (around 70Mb/s) that takes place at the 1 second epoch (see the second columns of Figures 10, 11 and 12). The large queue build up has been prevented by an anticipative increase of the gain, and decrease in available capacity used in the algorithm (see Section VII-A).

6. The low utilisation detection algorithm (cf. Section VII-A.2) allow switches SW1, SW3 and SW4 to quickly adapt to the fall in available capacity at SW2.

7. In the experiments for WAN2, (the plot in the second row and second column of Figures 10, 11 and 12), note a queue build up at SW2 time 250mS. This is due to large propagation delays in the control loop causing an overshoot in the computed link control parameter. Since link 2 is the first bottleneck link for sessions 1,2 and 6, (note that link 3 is also a bottleneck for sessions 1 and 2), the effect of the overshoot is a temporarily higher input rate than the output rate at queue 2 causing queue build up. Note that a similar overshoot in LCP is observed at link 3; the flows into link 3 due to sessions 1 and 2 are constrained by the output rate of link 2, however, hence a similar queue length build up does not take place at link 3.

8. At the 2 second epoch, when the available capacity at link 2 returns to its behaviour before the 1 second epoch, the LCP of link 3 needs to drop back to the lower value. In the LAN

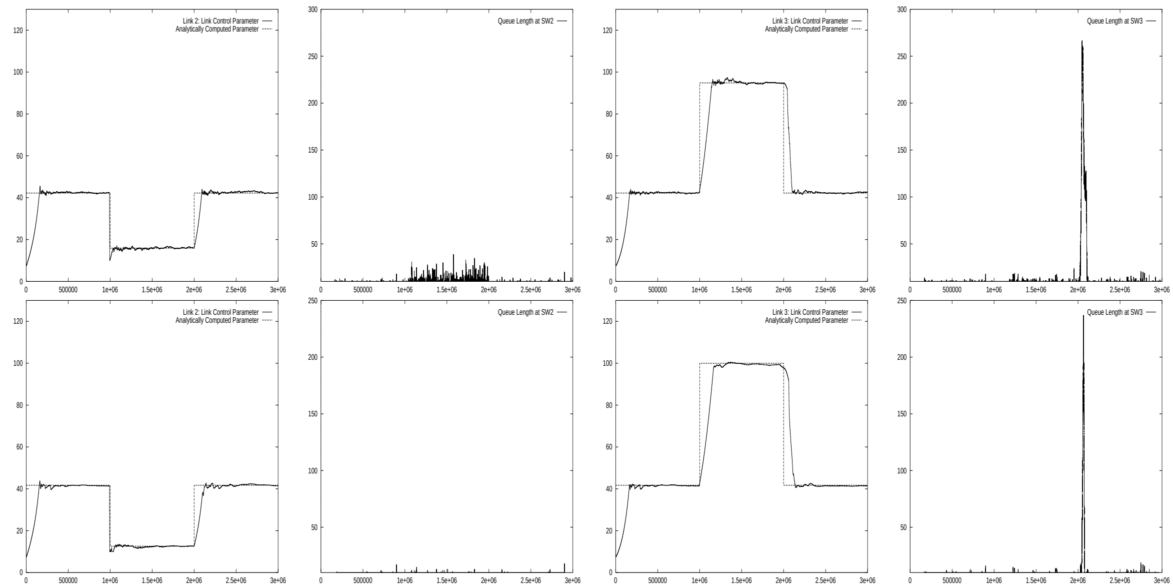

Fig. 11. WAN1 Environment: Plots of the link control parameter and queue lengths at link 2 and link 3. Row1 and Row2 correspond to 0.95 × mean, and ESC respectively. Col1 and Col3: LCPs from the simulation and analytically obtained values at link 2, link 3 respectively; x-axis: time in microseconds, y-axis: LCP in Mb/s. Col2 and Col4: queue length at link 2, link 3 respectively; x-axis: time in microseconds, y-axis: queue length in cells.

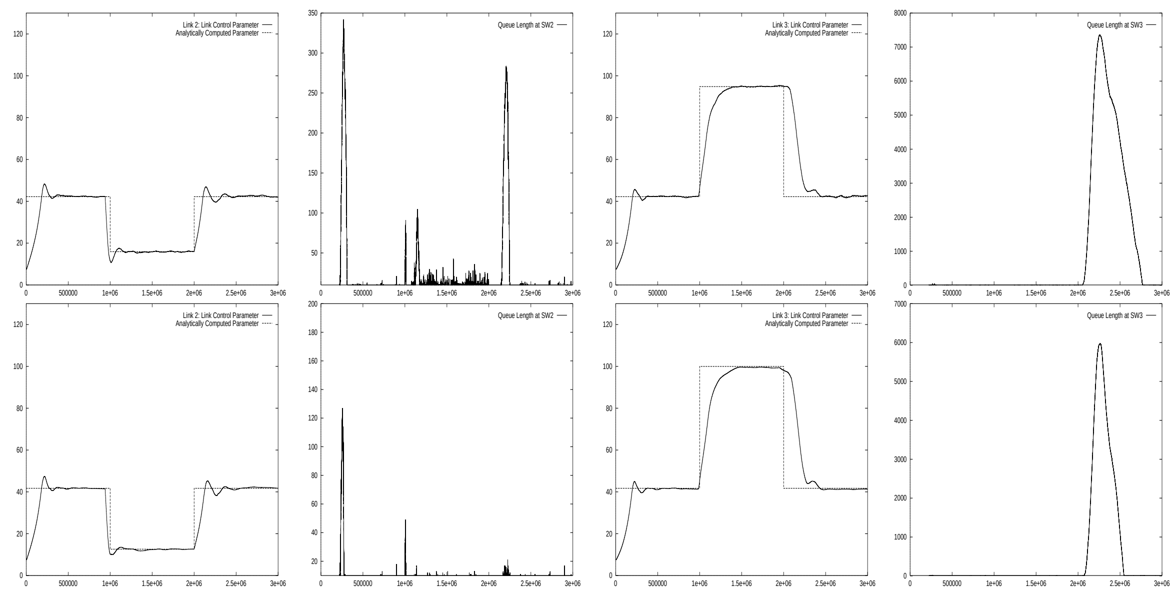

Fig. 12. WAN2 Environment: Plots of the link control parameter and queue lengths at link 2 and link 3. Row1 and Row2 correspond to 0.95 × mean, and ESC respectively. Col1 and Col3: LCPs from the simulation and analytically obtained values at link 2, link 3 respectively; x-axis: time in microseconds, y-axis: LCP in Mb/s. Col2 and Col4: queue length at link 2, link 3 respectively; x-axis: time in microseconds, y-axis: queue length in cells.

and WAN1 experiments, the queue threshold based detection of a remote capacity increase (cf. Section VII-A.2) enables quick adaptation of the link control parameters at SW1, SW3 and SW4 to the increase in the available capacity at SW2 at the 2 second epoch. However, in the WAN2 experiments, a large queue build up is observed at SW1, SW3 and SW4 at the 2 second epoch. The last column of Figure 12 shows this for link 3. The reasons include the following:

(a) A large number of "in transit" cells due to the large propagation delays.

(b) The small value of the initial gain (cf. Table I) that the stochastic approximation is reset to.

Adaptive rate control is known to be problematic when there are sudden changes in available capacity, and the propagation delays are large [22]. Further investigations needs to be carried out to obtain a strategy for adapting the gain in WAN2 environments to prevent such large queue build up.

9. We have also observed that during periods when the available capacity process is stationary, the link buffers are best controlled by the rate control algorithm that uses ESC as the measure of available capacity. This is to be expected as the ESC approach aims at a target buffer threshold crossing probability. In order to demonstrate the advantage of using an ESC estimate (Alg. B) instead of a scaled short term average (Alg. A), we compute the fraction of time the buffer size was above 10 cells during the interval between 1.2s and 2s (from the plots it is clear that the

LCP's are very close to their ideal values in this interval; also this is in the interval in which Link 2 has a different service process). These values are presented in Table III. Note that the experiments with ESC (Alg. B) give the best results (the probabilities of crossing a buffer level of 10 are close to or less than 10^{-3} in each case). It is also evident from the plots of the queue lengths, that the queue lengths observed when using an ESC estimate are smaller than the queue lengths observed when Alg. A is used. At SW2 a pronounced difference is observed between the buffer threshold crossing probability with the available capacity taken as 0.95 times the mean, and the buffer threshold crossing probability with the available capacity computed using the ESC techniques; note specially the second plot in the second column of Figure 12. This is due to the larger difference between the ESC and the mean during this interval (see Table II). Note that since the LCPs converge during a period in which the available capacity process is stationary, the link utilisation is the maximum possible subject to the buffer overflow constraint. This follows from the definition of the ESC.

Algorithm	Links	$P(Q > 10)$		
		LAN	WAN1	WAN2
Alg. A	SW1	0.005492	0.005125	0.004195
	SW2	0.124316	0.145884	0.134630
	SW3	0.011678	0.010251	0.007325
	SW4	0.004255	0.004931	0.004188
Alg. B	SW1	0.000427	0.000635	0.000575
	SW2	0.000959	0.000846	0.001040
	SW3	0.000479	0.001283	0.000412
	SW4	0.000208	0.000529	0.000451

TABLE III

FRACTION OF TIME THE BUFFER SIZE AT THE SWITCHES WAS ABOVE 10 CELLS IN TIME BETWEEN 1.2S AND 2S

IX. CONCLUSION

In this paper we have presented a new approach for designing explicit rate control algorithms for elastic sessions in packet networks. We began by studying a generalisation of the the max-min fairness framework that includes the case where sessions can demand a minimum rate guarantee. We then proposed a distributed stochastic approximation algorithm that computes the rate allocation while being robust to short term variations in available capacity, asynchrony between the processing elements and delays in the network. In order to ensure low losses, a framework for estimating the available capacity that inherently ensures low losses was proposed. An algorithm for estimating the available capacity was also presented. The distributed rate allocation algorithm was combined with capacity estimation techniques in simulation experiments to demonstrate the efficacy of the proposed approach in the ATM networks scenario.

The algorithms that we have developed have the nice property of not requiring per session rate monitoring, nor per session marking of control cells. Thus we expect that they may lead to more efficient implementations at high network speeds.

REFERENCES

[1] "ATM Forum Traffic Management Specification; version 4.0," April 1996.

[2] Santosh P. Abraham and Anurag Kumar, "Max-Min Fair Rate Control of ABR Connections with Nonzero MCRs", Proceedings IEEE *Globecom'97*, pp 498-502

[3] Santosh P. Abraham and Anurag Kumar, "A Stochastic Approximation Approach for Max-Min Fair Adaptive Rate Control of ABR Sessions with MCRs", *IEEE Infocom'98*, San Francisco, Mar-Apr, 1998.

[4] Santosh P. Abraham and Anurag Kumar, "A Simulation Study of an Adaptive Distributed Algorithm for Max-Min Fair Rate Control of ABR Sessions" *Proc. CCBR'98*, June 1998, Ottawa, Canada.

[5] Santosh P. Abraham, "Asynchronous Distributed Rate Control Algorithms for Best-Effort Sessions in Integrated Services Networks with Minimum Rate Guarantees", *Ph.D. Thesis*, ECE Department, Indian Institute of Science, Bangalore, India, October 1998.

[6] D. Bertsekas and R.G. Gallager, *Data Networks*, Prentice Hall, 1987.

[7] F. Bonomi, K.W. Fendick, "The Rate-Based Flow Control Framework for the Available Bit Rate ATM Service" *IEEE Network* March/April 1995, pp 25-39

[8] V. Borkar, "Asynchronous Stochastic Approximation" *SIAM J. Control and Optim.*, vol. 36, No. 3, May 1998, pp. 840-851

[9] Anna Charny, "An algorithm for rate allocation in a packet-switching, network with feedback", *Master's thesis*, MIT, Cambridge, May 1994

[10] C. Courcoubetis, G. Kesidis, A. Ridder, J. Walrand, and R. Weber, "Admission Control and Routing in ATM Networks using Inferences from Measured Buffer Occupancy", *IEEE Trans. Commn.*, Vol. 43, pp. 1774-1784, April 1995.

[11] N.G. Duffield, N. O'Connell, "Large Deviations and Overflow Probabilities for the General Single-Server Queue, with Applications", *Proc. Camb. Phil. Soc.* Vol. 118, 1995, pp.363-374.

[12] Anwar I. Elwalid and D. Mitra, "Effective Bandwidth of General Markovian Traffic Sources and Admission Control in High Speed Networks", *IEEE/ACM Trans. on Networking*, Vol. 1, No. 3, June 1993.

[13] C. Fulton, San-Qi Li and C.S. Lim, "UT: ABR Feedback Control with tracking", *Proc. IEEE INFOCOM'97*, April 1994, pp. 806-815.

[14] E. M. Gafni and D.P. Bertsekas, "Dynamic Control of Session Input Rates in Communication Networks," *IEEE Trans. Automatic Control*, Vol AC-29, No. 11, November 1984.

[15] H.P. Hayden, "Voice flow control in integrated packet networks," M.S. thesis, Dept. of Electrical Engg. and Computer Sci. Massachussetts Inst. Tech. Cambridge, MA, June 1981.

[16] Y.T. Hou, H.H.-Y. Tzeng, V.P. Kumar, "On Fair Rate Allocation Policies with Minimum Cell Rate Guarantees for ABR Service in ATM Networks" *Proc. ITC'15*, Washington, D.C., June 1997.

[17] R. Jain, S. Kalyanraman, R. Viswanathan and R. Goyal, " A Sample Switch Algorithm", *ATM Forum/95-0178,* February 1995.

[18] L. Kalampoukas, A. Varma, K.K. Ramakrishnan, "An Efficient Rate Allocation Algorithm for Packet-Switched Networks Providing Max-Min Fairness", *Preprint*

[19] F.P. Kelly, A.K. Maulloo, and D.K.H. Tan, "Rate Control for Communication Networks, Shadow Prices, Proportional Fairness, and Stability," *The Journal of Operational Research Society, Vol. 49, 1998.*

[20] George Kesidis, Jean Walrand and Cheng-Shang Chang "Effective Bandwidths for Multiclass Markov Fluids and Other ATM Sources" *IEEE/ACM Transactions on Networking*, Vol.1,No. 4,August 1993,pp 424-428.

[21] A. Kolarov, G. Ramamurthy, "A Control Theoretic Approach to the Design of an Explicit Rate Controller for ABR Service" *Proc INFOCOM'97*, April 1997, pp. 293-301.

[22] A. Kolarov, G. Ramamurthy, "A Control Theoretic Approach to the Design of Closed Loop Rate Based Flow Control For High Speed ATM Networks" *IEEE/ACM Transactions on Networking*, Vol.7,No. 5,October 1999,pp 741-753.

[23] Anurag Kumar, "Adaptive Load Control of the Central Processor in a Distributed System with a Star Topology" *IEEE Trans. on Computers*, Vol 38, no. 11, Nov 1989, pp. 1502-1512.

[24] H.J. Kushner, D.S. Clark, *Stochastic Approximation Methods for Constrained and Unconstrained Systems* Springer-Verlag 1978.

[25] J. Mosley, "Asynchronous Distributed Flow Control Algorithms," Ph.D. dissertation, Dep. Elec. Engg. Comp. Sci. Mass. Inst. Tech. , Cambridge, MA, June 1984.

[26] G. de Veciana, J. Walrand, "Effective Bandwidths: Call Admission, Traffic Policing and Filtering for ATM Networks", *Queueing Systems, Theory and Applications: QUESTA*, August 1993.

[27] G. de Veciana et al, *Proc. IEEE Globecom '98*, Sydney, Australia, November 1998.