On Fair Rate Allocation Policies with Minimum Cell Rate Guarantee for ABR Service in ATM Networks

Yiwei Thomas Hou,[a*] Henry H.-Y. Tzeng,[b] and Vijay P. Kumar[c]

[a]Dept. of Electrical Engineering, Polytechnic University, Brooklyn, NY 11201, USA.

[b]Bell Laboratories, Lucent Technologies, Room 4G-526, 101 Crawfords Corner Road, Holmdel, NJ 07733, USA.

[c]Bell Laboratories, Lucent Technologies, Room 4G-533, 101 Crawfords Corner Road, Holmdel, NJ 07733, USA.

A novel concept in available bit rate (ABR) service model as defined by the ATM Forum is the minimum cell rate (MCR) support for each ABR virtual connection. In this paper, we present a fundamental study of rate allocation policies with MCR guarantee. In particular, we define three network bandwidth allocation policies that guarantee MCR requirements. For each policy, we present a centralized algorithm to compute rate allocation. The practical significance of our policies are substantiated with our further development of distributed algorithms for ABR service. The performance of our ABR algorithms are demonstrated with simulation results.

1. INTRODUCTION

A key performance issue associated with ABR service is fair allocation of network bandwidth for each virtual connection (VC). The ATM Forum has adopted the max-min fairness criterion for ABR service [2]. Prior efforts on max-min fair rate allocation for ABR service such as [3,5,6] did not address the fairness issue in the context of MCR requirement. For connections with MCR requirement, a set of new policies need to be defined. Our paper focuses on this fundamental problem for ABR service.

We consider three policies, namely *MCRadd*, *MCRprop*, and *MCRmin* to support MCR constraints. These three policies were informally described in [4,9] for the simple single node case. Each policy strives to achieve a different fairness objective with MCR guarantee for each connection. In particular, the MCRadd policy allocates each VC session with its MCR plus a max-min fair share from the remaining network capacity; the MCRprop policy allocates network bandwidth in proportional to each VC's MCR;[2] MCRmin policy guarantees each VC session with either its MCR or a max-min fair share, whichever

*Part of this work was completed while the first author spent the summer of 1996 at Bell Labs, Lucent Technologies, Holmdel, NJ. Mr. Hou is supported by a National Science Foundation (NSF) Graduate Research Traineeship at the New York State Center for Advanced Technology in Telecommunications (CATT), Polytechnic University, Brooklyn, NY, USA.

[2]We assume nonzero MCR requirement for the MCRprop policy throughout the paper.

is greater. In this paper, we formally define these three policies. We also present a centralized bandwidth assignment algorithm to achieve each policy.

Even though centralized algorithms are essential for our understanding on how each policy allocates network bandwidth for each VC, the practical significance of these policies would be limited if we can not develop distributed algorithms to achieve each policy in the context of ABR traffic management. Therefore, we propose a set of heuristic algorithms consistent with the ABR traffic management specifications in [1]. We demonstrate the effectiveness of our ABR algorithms with simulation results based on benchmark network configurations suggested by the ATM Forum.

The remainder of this paper is organized as follows. Section 2 first summarizes key results on max-min fairness, and then defines three rate allocation policies with MCR guarantee. We also present a centralized algorithm for each policy. Section 3 shows the distributed ABR implementations for each policy. In Section 4, we present simulation results. Section 5 concludes this paper.

2. FAIR RATE ALLOCATION POLICIES WITH MCR GUARANTEE

Before we define our rate allocation policies, we shall briefly summarize key results on max-min fairness for the special case when there is no MCR requirement [2].

2.1. Preliminaries

In our model, a network $\mathcal{N}$ is characterized by a set of links $\mathcal{L}$ and sessions $\mathcal{S}$.[3] Each session $s \in \mathcal{S}$ traverses one or more links in $\mathcal{L}$ and is allocated a specific rate r_s. The (aggregate) allocated rate F_ℓ on link $\ell \in \mathcal{L}$ of the network is

$$F_\ell = \sum_{s \in \mathcal{S} \text{ traversing link } \ell} r_s \ .$$

Let C_ℓ be the capacity of link ℓ. A link ℓ is *saturated* or *fully utilized* if $F_\ell = C_\ell$. A rate vector $r = (\cdots, r_s, \cdots)$ is *feasible* if the following two constraints are satisfied:

$$r_s \geq 0 \quad \text{for all } s \in \mathcal{S},$$
$$F_\ell \leq C_\ell \quad \text{for all } \ell \in \mathcal{L}.$$

Definition 1 A rate vector r is *max-min fair* if it is feasible, and for each $s \in \mathcal{S}$ and every feasible rate vector $\hat{r}$ in which $\hat{r}_s > r_s$, there exists some session $t \in \mathcal{S}$ such that $r_s \geq r_t$ and $r_t > \hat{r}_t$. □

Definition 2 Given a feasible rate vector r, a link $\ell \in \mathcal{L}$ is a *bottleneck link* with respect to r for a session s traversing ℓ if $F_\ell = C_\ell$ and $r_s \geq r_t$ for all sessions t traversing link ℓ. □

Theorem 1 A feasible rate vector r is max-min fair if and only if each session has a bottleneck link with respect to r.[4] □

[3] From now on, we shall use the terms "session", "virtual connection", and "connection" interchangeably throughout the paper.

[4] For a proof of Theorem 1, see [2].

2.2. Rate Allocation Policies with MCR Guarantee

We are now ready to define our rate allocation policies to support MCR requirements. We also present a centralized algorithm for each policy. For the sake of feasibility, we assume that the sum of VCs' MCR requirements traversing any link does not exceed that link's capacity, i.e. $\sum_{\text{all } s \in \mathcal{S} \text{ traversing } \ell} \text{MCR}_s \leq C_\ell$ for every $\ell \in \mathcal{L}$. This assumption is enforced by admission control at call setup time for each connection. Furthermore, we say that a rate vector $r = (\cdots, r_s, \cdots)$ is *MCR-feasible* if the following two constraints are satisfied:

$$r_s \geq \text{MCR}_s \quad \text{for all } s \in \mathcal{S},$$
$$F_\ell \leq C_\ell \quad \text{for all } \ell \in \mathcal{L}.$$

2.2.1. Policy 1: MCRadd

The MCRadd fairness policy first allocates each session $s \in \mathcal{S}$ with its MCR and then applies max-min fairness algorithm for all sessions on the *remaining* network capacity. The rate allocation of each session is its MCR plus a max-min fair share from the network with the remaining capacity. Formally, this policy is defined as follows.

Definition 3 A rate vector r is *MCRadd fair* if it is MCR-feasible, and for each $s \in \mathcal{S}$ and every MCR-feasible rate vector $\hat{r}$ in which $\hat{r}_s > r_s$, there exists some session $t \in \mathcal{S}$ such that $r_s - \text{MCR}_s \geq r_t - \text{MCR}_t$ and $r_t > \hat{r}_t$. □

We define a new notion of bottleneck link as follows.

Definition 4 Given an MCR-feasible rate vector r, a link $\ell \in \mathcal{L}$ is an *MCRadd-bottleneck link* with respect to r for a session s traversing ℓ if $F_\ell = C_\ell$ and $r_s - \text{MCR}_s \geq r_t - \text{MCR}_t$ for all sessions t traversing link ℓ. □

It can be shown that the following theorem is true.

Theorem 2 An MCR-feasible rate vector r is MCRadd fair if and only if each session has an MCRadd bottleneck link with respect to r. □

The following centralized algorithm computes the rate allocation for each session in any network $\mathcal{N}$ such that the MCRadd fairness policy is satisfied.

Algorithm 1

Initial conditions:

$k = 1$, $\mathcal{S}^1 = \mathcal{S}$, $\mathcal{L}^1 = \mathcal{L}$,
$r_s^0 = \text{MCR}_s$, for every $s \in \mathcal{S}$,
$F_\ell^0 = \sum_{\text{all } s \in \mathcal{S} \text{ traversing } \ell} \text{MCR}_s$, for every $\ell \in \mathcal{L}$.

1. $n_\ell^k :=$ number of sessions $s \in \mathcal{S}^k$ traversing link ℓ, for every $\ell \in \mathcal{L}^k$.

2. $a^k := \min_{\ell \in \mathcal{L}^k} \dfrac{(C_\ell - F_\ell^{k-1})}{n_\ell^k}$.

3. $r_s^k := \begin{cases} r_s^{k-1} + a^k & \text{if } s \in \mathcal{S}^k, \\ r_s^{k-1} & \text{otherwise.} \end{cases}$

4. $F_\ell^k := \sum_{\text{all } s \in \mathcal{S} \text{ traversing } \ell} r_s^k$, for every $\ell \in \mathcal{L}^k$.

5. $\mathcal{L}^{k+1} := \{\ell \mid C_\ell - F_\ell^k > 0, \ell \in \mathcal{L}^k\}$.

6. $\mathcal{S}^{k+1} := \{s \mid s \text{ does not traverse any link in } (\mathcal{L} - \mathcal{L}^{k+1})\}$.

7. $k := k + 1$.

8. If $\mathcal{S}^k$ is empty, then $r^{k-1} = (\cdots, r_s^{k-1}, \cdots)$ is the rate vector satisfying the MCRadd fairness policy and this algorithm terminates; otherwise, go back to Step 1. □

2.2.2. Policy 2: MCRprop

The MCRprop fairness policy allocates a rate for each session proportional to its MCR and achieves max-min fairness on the normalized rate (with respect to its MCR) for each session. Formally, this policy is defined as follows.

Definition 5 A rate vector r is *MCRprop fair* if it is MCR-feasible, and for each $s \in \mathcal{S}$ and every MCR-feasible rate vector $\hat{r}$ in which $\hat{r}_s > r_s$, there exists some session $t \in \mathcal{S}$ such that $\frac{r_s}{\text{MCR}_s} \geq \frac{r_t}{\text{MCR}_t}$ and $r_t > \hat{r}_t$. □

We define a new notion of bottleneck link as follows.

Definition 6 Given an MCR-feasible rate vector r, a link $\ell \in \mathcal{L}$ is an *MCRprop-bottleneck link* with respect to r for a session s traversing ℓ if $F_\ell = C_\ell$ and $\frac{r_s}{\text{MCR}_s} \geq \frac{r_t}{\text{MCR}_t}$ for all sessions t traversing ℓ. □

It can be shown that the following theorem is true.

Theorem 3 An MCR-feasible rate vector r is MCRprop fair if and only if each session has an MCRprop-bottleneck link with respect to r. □

Since the centralized algorithm for MCRprop fairness policy is very similar to Algorithm 1, we omit to show it here due to space limitation.

2.2.3. Policy 3: MCRmin

A rate vector r is *MCRmin fair* if it is MCR-feasible and for each session s, one cannot generate a new MCR-feasible rate vector by increasing the allocated rate r_s without decreasing the allocated rate of some other session t with a rate r_t already less than or equal to r_s in the rate vector r. Formally, this policy is defined as follows.

Definition 7 A rate vector r is *MCRmin fair* if it is MCR-feasible, and for every $s \in \mathcal{S}$ and every MCR-feasible rate vector $\hat{r}$ in which $\hat{r}_s > r_s$, there exists some session $t \in \mathcal{S}$ such that $r_s \geq r_t$, and $r_t > \hat{r}_t$. □

Remark 1 Note that the MCRmin fairness definition above is similar to the max-min fairness definition (Definition 1) except the additional requirement that r must be MCR-feasible. □

We define a new notion of bottleneck link as follows.

Definition 8 Given an MCR-feasible rate vector r, a link $\ell \in \mathcal{L}$ is an *MCRmin-bottleneck link* with respect to r for a session s traversing ℓ if $F_\ell = C_\ell$ and $r_s \geq r_t$ for every session t traversing link ℓ for which $r_t > \text{MCR}_t$. □

It can be shown that the following theorem is true.

Theorem 4 An MCR-feasible rate vector r is MCRmin fair if and only if each session has an MCRmin-bottleneck link with respect to r. □

The following centralized algorithm computes rate allocation for each session in any network $\mathcal{N}$ such that the MCRmin fairness policy is satisfied.

Algorithm 2

Initial conditions:

$k = 1$, $\mathcal{S}^1 = \mathcal{S}$, $\mathcal{L}^1 = \mathcal{L}$,
$r_s^0 = \text{MCR}_s$, for every $s \in \mathcal{S}$,
$F_\ell^0 = \sum_{\text{all } s \in \mathcal{S} \text{ traversing } \ell} \text{MCR}_s$, for every $\ell \in \mathcal{L}$.

1. Sort all the sessions in $\mathcal{S}^k$ in n sets ($1 \leq n \leq |\mathcal{S}^k|$):

   $u_1, u_2, \cdots, u_n$

   such that (1) every session in the same set has the same rate; and (2) rate values in these sets are in increasing order, i.e.

   $r_{s_1 \in u_1} < r_{s_2 \in u_2} < \cdots < r_{s_n \in u_n}$

2. $n_\ell^k :=$ number of sessions $s \in u_1$ traversing link ℓ, for every $\ell \in \mathcal{L}^k$.

3. $$a^k := \begin{cases} \min\{\min_{\ell \text{ traversed by } s \in u_1} \frac{(C_\ell - F_\ell^{k-1})}{n_l^k}, \ (r_{t \in u_2} - r_{s \in u_1})\} & \text{if } n > 1, \\ \min_{\ell \text{ traversed by } s \in u_1} \frac{(C_\ell - F_\ell^{k-1})}{n_l^k} & \text{if } n = 1. \end{cases}$$

4. $$r_s^k := \begin{cases} r_s^{k-1} + a^k & \text{if } s \in u_1, \\ r_s^{k-1} & \text{otherwise.} \end{cases}$$

5. $F_\ell^k := \sum_{\text{all } s \in \mathcal{S} \text{ traversing } \ell} r_s^k$, for every $\ell \in \mathcal{L}^k$.

6. $\mathcal{L}^{k+1} := \{\ell \mid C_\ell - F_\ell^k > 0, \ell \in \mathcal{L}^k\}$.

7. $\mathcal{S}^{k+1} := \{s \mid s \text{ does not traverse any link } \ell \in (\mathcal{L} - \mathcal{L}^{k+1})\}$.

8. $k := k + 1$.

9. If $\mathcal{S}^k$ is empty, then $r^{k-1} = (\cdots, r_s^{k-1}, \cdots)$ is the rate vector satisfying the MCRmin fairness policy and this algorithm terminates; otherwise, go back to Step 1. □

2.2.4. Example

We use the following example to illustrate how our centralized algorithms allocate network bandwidth for each policy.

In this network configuration (Fig. 1), the output port links of SW1 (Link 12) and SW2 (Link 23) are bottleneck nodes for sessions. Specifically, VC sessions s1, s2 and s3 share Link 12 while s1 and s4 share Link 23. Again, the link capacity is assumed to equal to 1 unit. The MCR requirement for each session is listed in Table 1. Using centralized algorithm for each policy, we obtain the rate allocation for each session under each policy in Table 1.

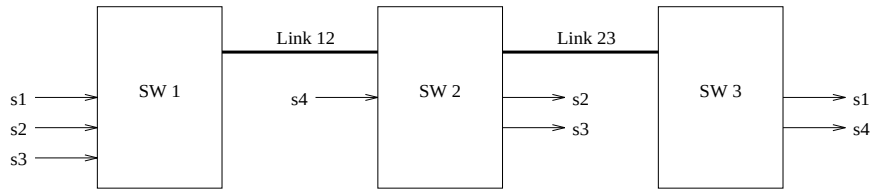

Figure 1. The three-node network configuration.

Table 1
Rate allocation for each session under each policy for the three-node network configuration.

Sessions	MCR Requirement	Policy		
		MCRadd	MCRprop	MCRmin
$s1$	0.35	0.517	0.70	0.35
$s2$	0.10	0.266	0.20	0.325
$s3$	0.05	0.217	0.10	0.325
$s4$	0.10	0.483	0.30	0.65

Although the centralized algorithms presented in this section are helpful for our understanding on how each policy works to perform network-wide bandwidth assignment, they cannot be directly applied to a distributed traffic management environment for ABR

service. To show the practical merit of our defined rate allocation policies, we will develop distributed algorithms conforming to the ATM Forum ABR traffic management specifications [1] to achieve these policies in the next section.

3. DISTRIBUTED HEURISTIC ALGORITHMS

Our ABR implementations for MCRadd, MCRprop, and MCRmin are all based on the *Intelligent Marking* technique, originally proposed in [6] and further refined in [7,8]. The key idea of this technique is to let each congested switch estimate "optimal" cell rate for each VC bottlenecked at the switch with a small number of computations and without having the switch keeping track of each VC's state information (so called per-VC accounting). Using simple feedback mechanisms, this estimated rate will be employed to adjust the cell rates of the sources. It has been shown in [7,8] that this algorithm provides max-min fair allocation.

Fig. 2 illustrates the behavior of the Intelligent Marking technique. For each queue of a switch, four variables LOAD, MCCR (Mean CCR), UCR (Upper Cell Rate), and EBR (Estimated Bottleneck Rate) are defined. The value of LOAD corresponds to the aggregated cell rate entering the queue normalized with respect to link capacity and is measured by the switch over a period of time. The value of MCCR contains an estimated average cell rate of all VCs traversing this queue; the value of UCR contains an estimated upper limit on the cell rate of all VCs traversing this queue; and the value of EBR contains an estimated bottleneck rate at this queue. Furthermore, two parameters TLR and α are defined for each queue, where the value of TLR is the target load ratio, and $0 < \alpha < 1$.

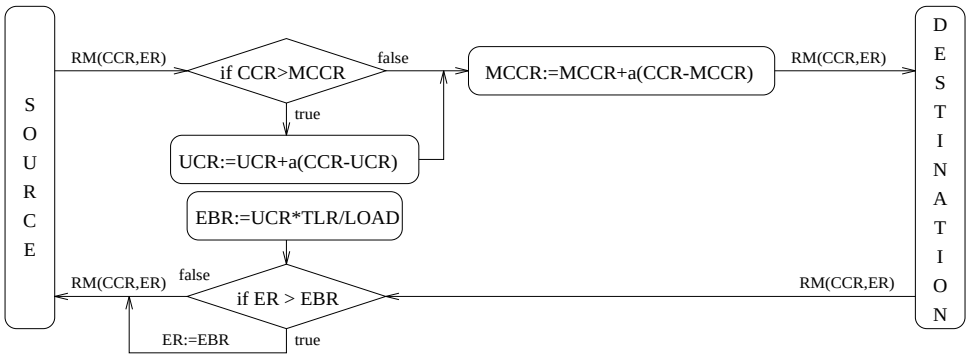

Figure 2. Switch behavior of Intelligent Marking protocol.

The Intelligent Marking algorithm is a heuristic algorithm. We will give an intuitive explanation on how it works. The RM cells from all VCs participate in exponential averaging for MCCR with MCCR := MCCR + α(CCR − MCCR) while only some VCs with greater than average rate (potentially VCs bottlenecked at this switch) participate in UCR averaging, which is used to estimate bottleneck link rate. Since there can be only one bottleneck rate at a link and it is no less than any of the VC's rate at steady state,

the final rate assignment for each VC converges to max-min fair rate (the "if" part of Theorem 1) for the entire network.

3.1. Heuristic Algorithm for MCRadd

Since the Intelligent Marking technique allocates max-min fair rate for each VC from network bandwidth when there is no MCR requirement [7,8] and our MCRadd policy allocates each VC with MCR plus a max-min fair share from the *remaining* network capacity, we can let the offsetted cell rate, CCR – MCR for each VC to participate in Intelligent Marking and estimate the MCRadd-bottleneck link rate from the remaining network bandwidth.

Fig. 3 illustrates the switch behavior of our heuristic algorithm. Similar to the Intelligent Marking algorithm, for each queue of a switch, four variables named LOAD, MFSR (Mean Fair Share Rate), Upper Fair Share Rate (Upper Fair Share Rate), and EBR (Estimated Bottleneck Rate) are defined. The LOAD is the same as before. The value of MFSR contains an estimated MCR-offsetted average rate of all VCs traversing this queue; the UFSR contains an estimated MCR-offsetted upper rate; and the value of EBR contains an estimated MCRadd-bottleneck link rate. The parameters TLR and α are defined the same as before.

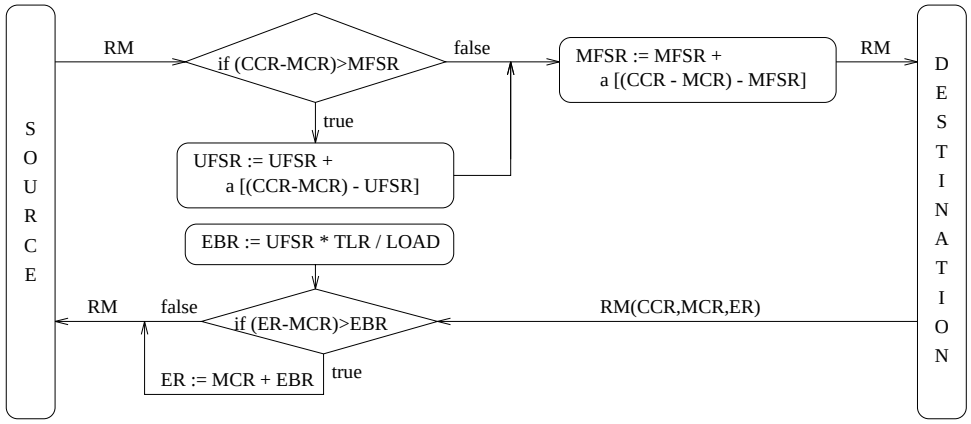

Figure 3. Switch behavior of heuristic implementation for the MCRadd policy.

3.2. Heuristic Algorithm for MCRprop

Similar to the heuristic implementation for the MCRadd policy, heuristic algorithm for MCRprop can be implemented by letting normalized cell rate (e.g. CCR/MCR, ER/MCR) of each VC at the switch to participate in Intelligent Marking.

Fig. 4 illustrates the switch behavior of our heuristic algorithms. Four variables named LOAD, NMR (Normalized Mean Rate), NUR (Normalized Upper Rate) and NBR (Normalized Bottleneck Rate) are defined for each output port of an ATM switch. The value of LOAD is the same as before. The value of NMR contains an estimated normalized (with respect to MCR) average rate for all VCs traversing this link; the value of NUR

contains an estimated normalized upper rate; and NBR contains an estimated normalized MCRprop-bottleneck link rate. Here, NMR, NUR and NBR are all dimensionless. TLR and α are defined the same as before.

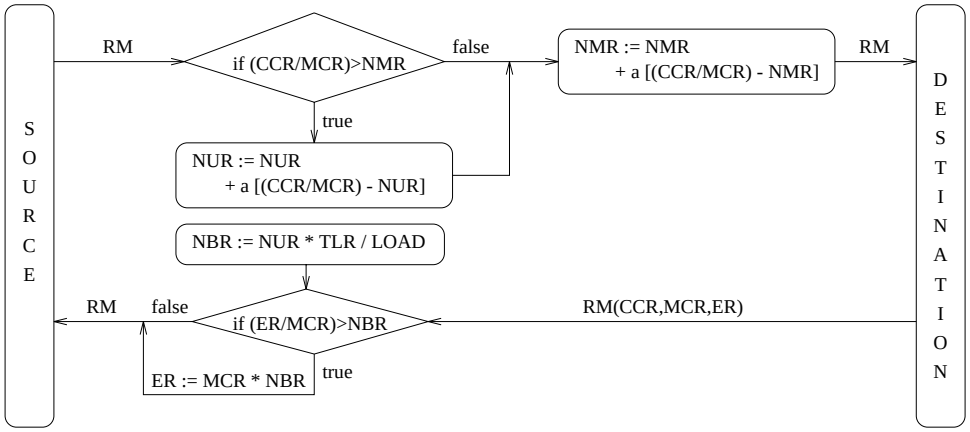

Figure 4. Switch behavior of heuristic implementation for the MCRprop policy.

3.3. Heuristic Algorithm for MCRmin

Since 1) Intelligent Marking at a switch computes a max-min fair share with zero MCR requirement [7,8]; and 2) SES defined in [1] generates cells always at a rate no less than MCR, we would expect that combined mechanisms of switch algorithm (Intelligent Marking) and SES may somehow serve the MCRmin policy. This is indeed the case and will be demonstrated by our simulation results in the next section.

4. SIMULATION RESULTS

Here we present a simulation study demonstrating the effectiveness of our heuristic ABR algorithms to achieve three rate allocation policies. Table 2 lists the parameters used in our simulation. The distance from source/destination to the switch is 100 m and the link distance between ATM switches is 10 km.

Due to space limitation, we will only present simulation results on the three-node network configuration shown in Fig. 1.

Figs. 5, 7, and 9 show the cell rates of each session under our heuristic ABR algorithms for MCRadd, MCRprop, and MCRmin policies, respectively. The cell rates shown in these plots are normalized with respect to the link capacity (150 Mbps) for easy comparison with those values obtained with our centralized algorithms (Table 1). We find that after initial transient period, the rate allocation through heuristic algorithms are quite accurate to achieve each policy. Figs. 6, 8, and 10 show the link utilization (Link12 and Link23) and buffer occupancy (SW1 and SW2) from the same simulation run under each ABR algorithm. We find that the bottleneck links are efficiently utilized with small buffer requirements.

Table 2
Simulation parameters.

End System	PCR	150Mbps
	ICR	MCR
	Nrm	32
	AIR	3.39 Mbps
Link	Speed	150 Mbps
Switch	Cell Switching Delay	4 μsec
	Output Buffer Size	2000 cells

5. CONCLUSION

We have defined three fair rate allocation policies to support MCR requirement for ABR service in ATM networks. Centralized algorithms to compute rate allocation under each policy are also presented. Furthermore, simple distributed implementations of three rate allocation policies are developed in the context of ATM Forum ABR traffic management specifications and their effectiveness are demonstrated with simulation results.

REFERENCES

1. ATM Forum Technical Committee, "Traffic Management Specification - Version 4.0," *ATM Forum/95-0013R13,* February 1996.
2. D. Bertsekas and R. Gallager, *Data Networks,* Prentice Hall, 1992.
3. A. Charny, D. Clark, and R. Jain, "Congestion Control with Explicit Rate Indication," *Proc. IEEE ICC'95,* pp. 1954-1963.
4. D. Hughes, "Fair Share in the Context of MCR," *ATM Forum Contribution 94-0977,* October 1994.
5. R. Jain, *et al.*, "ERICA Switch Algorithm: A Complete Description," *ATM Forum Contribution, 96-1172,* August 1996.
6. K.-Y. Siu and H.-Y. Tzeng, "Intelligent Congestion Control for ABR Service in ATM Networks," *ACM SIGCOMM Computer Communication Review*, 24(5):81–106, October 1994.
7. K.-Y. Siu and H.-Y. Tzeng, "Limits of Performance in Rate-based Control Schemes," *ATM Forum Contribution 94-1077,* November 1994.
8. H.-Y. Tzeng and K.-Y. Siu, "Comparison of Performance Among Existing Rate Control Schemes," *ATM Forum Contribution 94-1078,* November 1994.
9. N. Yin, "Max-Min Fairness vs. MCR Guarantee on Bandwidth Allocation for ABR," *Proc. IEEE ATM'96 Workshop,* San Francisco, CA, August 25-27, 1996.

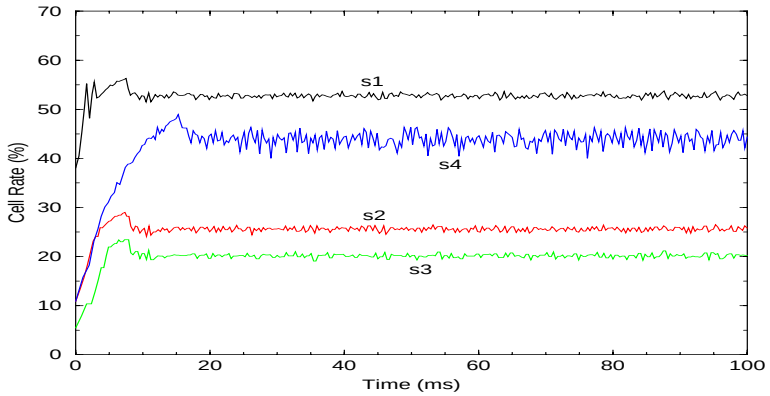

Figure 5. The cell rates of all connections with the MCRadd policy in the three-node configuration.

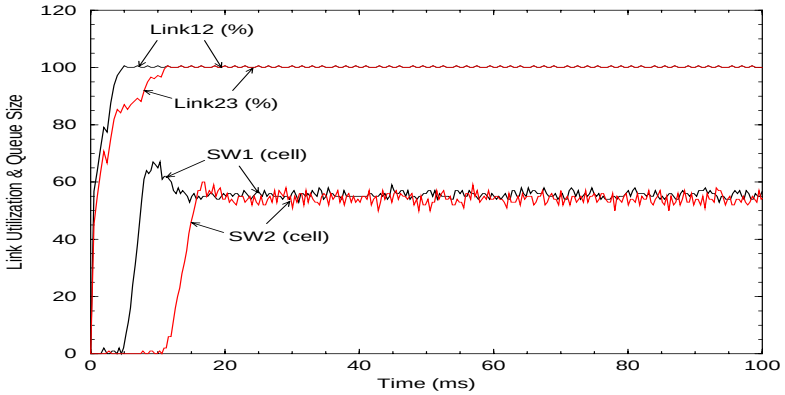

Figure 6. The link utilization and the queue size of the congested switches with the MCRadd policy in the three-node network configuration.

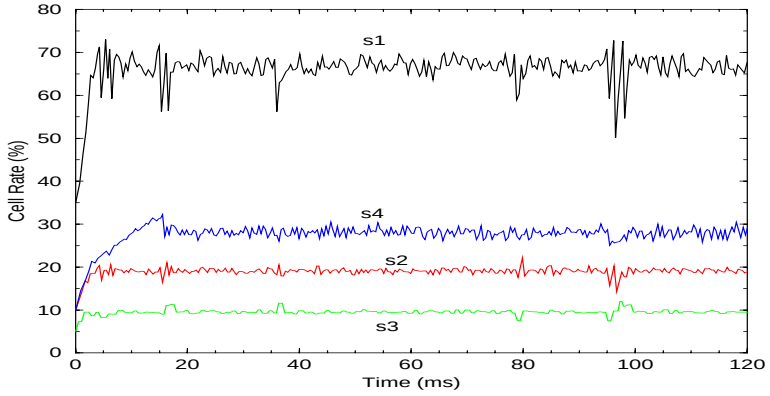

Figure 7. The cell rates of all connections with the MCRprop policy in the three-node network configuration.

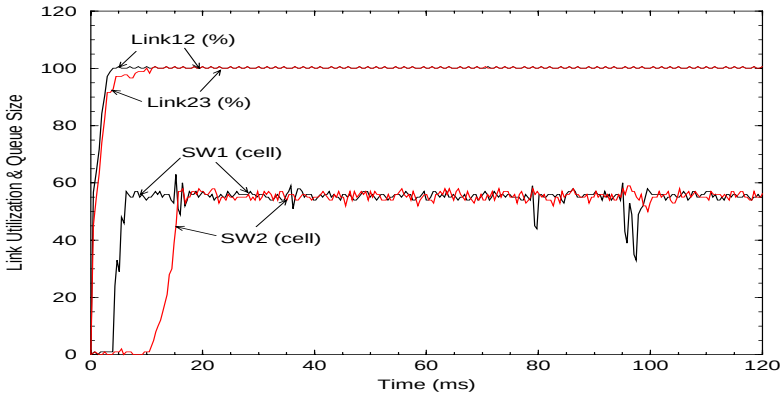

Figure 8. The link utilization and the queue size of the congested switches with the MCRprop policy in the three-node network configuration.

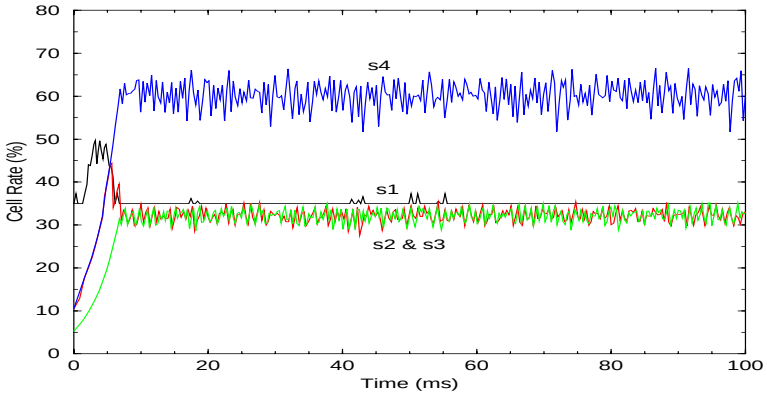

Figure 9. The cell rates of all connections with the MCRmin policy in the three-node network configuration.

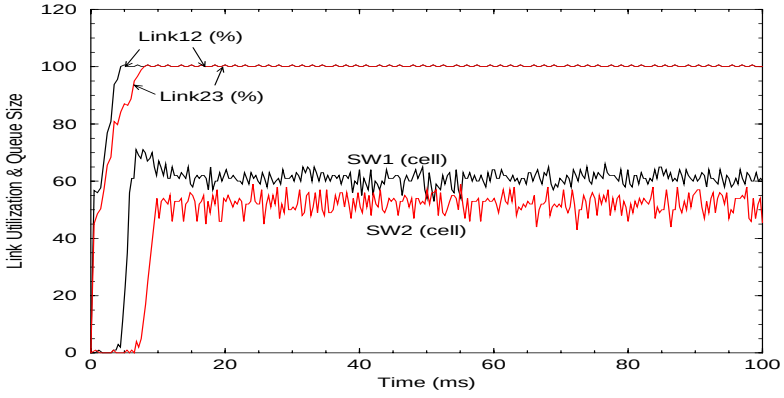

Figure 10. The link utilization and the queue size of the congested switches with the MCRmin policy in the three-node network configuration.