A Rate Allocation Policy with MCR/PCR Support and Distributed ABR Implementation Using Explicit Rate Feedback *

Yiwei Thomas Hou† Henry H.-Y. Tzeng‡ Shivendra S. Panwar§

Abstract

An important concept in the available bit rate (ABR) service model as defined by the ATM Forum is the minimum cell rate (MCR) guarantee as well as the peak cell rate (PCR) constraint for each ABR virtual connection (VC). Because of the MCR and PCR requirements, the well-known max-min fairness policy no longer suffices to determine rate allocation in the ABR service model. We introduce a network bandwidth assignment policy, MCRadd, which supports both the MCR and PCR requirements for each ABR virtual connection. A centralized algorithm is presented to compute network-wide bandwidth allocation to achieve this policy. Furthermore, an explicit-rate (ER) based ABR switch algorithm is developed to achieve the MCRadd policy in the distributed ABR environment and its convergence proof is also given. The performance of our ABR algorithm is demonstrated by simulation results based on the benchmark network configurations suggested by the ATM Forum.

Key Words: Rate Allocation Policy, Max-Min Fairness, Minimum Cell Rate, Peak Cell Rate, ABR Service, Centralized and Distributed Algorithms, Traffic Management, Congestion/Flow Control, ATM Networks.

1 Introduction

The ABR service defined by the ATM Forum [1] supports applications that allow the ATM source end system to adjust the information transfer rate based on the bandwidth availability in the network. By the specifications in [1], on the establishment of an ABR connection, the user shall specify to the network both a maximum bandwidth and a minimum required bandwidth, designated as peak cell rate (PCR) and minimum cell rate (MCR), respectively, for the requested connection. The source starts to transmit at an initial cell rate (ICR), which is greater than or equal to MCR, and may adjust its rate up to PCR based on congestion and bandwidth information from the network.

A key performance issue associated with ABR service is fair allocation of network bandwidth for each virtual connection. In particular, the ATM Forum has adopted the max-min fairness criterion to allocate network bandwidth for ABR connections [2]. Prior efforts to design ABR algorithms to achieve the max-min fair rate allocation, such as [4, 9, 10, 11, 12] did not address the fairness issue in the context of each individual connection's MCR requirement and PCR constraint. For connections with MCR requirements (a bandwidth QoS feature offered to ABR traffic by ATM networks) and PCR constraints (usually imposed by the host application or terminal equipment), a new definition of rate allocation policy is required.

In this paper, we present a rate allocation policy, called *MCRadd*, to allocate network bandwidth for each virtual connection with both MCR guarantee and PCR constraint. This policy was first informally described in [8, 13] for the simple single node case without PCR constraint. In this paper, we formally define this policy with MCR/PCR constraints. We also present a centralized bandwidth assignment algorithm to achieve the MCRadd policy.

To achieve the MCRadd policy for ABR service, we move on to develop a distributed ABR algorithm consistent with the ATM Forum ABR traffic management specifications. Our ABR algorithm is motivated by the work by Charny *et al.* [4], which achieves max-min fair rate allocation policy with no MCR and PCR constraints. We extend this technique and design an ABR algorithm to achieve the MCRadd policy with MCR/PCR constraints. An outline of a proof that the rate calculated by our ABR algorithm converges to the MCRadd policy through distributed and asynchronous iterations is also given.

Finally, we implement our ABR algorithm on a few benchmark network configurations suggested by the ATM Forum and use simulation results to further investigate its convergence properties.

The remainder of this paper is organized as follows. In Section 2, we present the MCRadd fairness policy. In section 3, we develop a distributed algorithm to achieve the MCRadd policy for ABR service and gives a correctness proof of its convergence. In Section 4, we present the simulation results of our ABR algorithm on a few network configurations. Section 5 concludes this paper.

*Part of this work was performed while Y. T. Hou spent the summer of 1996 at Bell Labs, Lucent Technologies, Holmdel, NJ. This work is supported in part by the New York State Center for Advanced Technology in Telecommunications (CATT), Polytechnic University, Brooklyn, NY.

†Y. T. Hou is a Ph.D. candidate under the National Science Foundation Graduate Research Traineeship Program at Polytechnic University, Brooklyn, NY.

‡H. H.-Y. Tzeng is with Bell Labs, Lucent Technologies, Holmdel, NJ.

§S. S. Panwar is with the Dept. of Electrical Engineering, Polytechnic University, Brooklyn, NY.

0-8186-7819-4/97 $10.00 © 1997 IEEE

2 The MCRadd Rate Allocation Policy

In our model, a network $\mathcal{N}$ is characterized by a set of links $\mathcal{L}$ and sessions $\mathcal{S}$.[1] 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* if $F_\ell = C_\ell$.

Let MCR_s and PCR_s be the MCR and PCR constraints for session $s \in \mathcal{S}$. For the sake of feasibility, we assume that the sum of VCs' MCR requirements traversing any link does not exceed that link's capacity. That is, $\sum_{\text{all } s \in \mathcal{S} \text{ traversing } \ell} \text{MCR}_s \leq C_\ell$ for every $\ell \in \mathcal{L}$. This condition is used by admission control at call setup time to determine whether or not to accept a new ABR virtual connection.

We say that a rate vector $r = (\cdots, r_s, \cdots)$ is *ABR-feasible* if the following two constraints are satisfied:

$$\begin{aligned} \text{MCR}_s \leq r_s \leq \text{PCR}_s \quad & \text{for all } s \in \mathcal{S}, \\ F_\ell \leq C_\ell \quad & \text{for all } \ell \in \mathcal{L}. \end{aligned}$$

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 (after removing MCR for each session from network capacity) while satisfying each session's PCR constraint. The rate allocation of each session $s \in \mathcal{S}$ is its MCR_s plus a max-min fair share from the network with the remaining capacity. Formally, this policy is defined as follows.

Definition 1 A rate vector r is *MCRadd fair* if it is ABR-feasible, and for each $s \in \mathcal{S}$ and every ABR-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 2 Given an ABR-feasible rate vector r, a link $\ell \in \mathcal{L}$ is a *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 two theorems hold [6].

Theorem 1 An ABR-feasible rate vector r is MCRadd fair if and only if each session has either an MCRadd-bottleneck link with respect to r or a rate assignment equal to its PCR. □

Theorem 2 There exists a unique rate vector that satisfies the MCRadd rate allocation policy. □

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

Based on Theorem 1, we construct the following centralized algorithm to compute the rate allocation for each session in any network $\mathcal{N}$ such that the MCRadd fairness policy is satisfied. Informally, the MCRadd centralized algorithm works as following:

1. Start the rate allocation of each session with its MCR.
2. Increase the rate of each session with the smallest rate increment such that either some link becomes saturated or some session reaches its PCR, whichever comes first.
3. Remove those sessions that either traverse saturated links or have reached their PCRs and the capacities associated with such sessions from the network.
4. If there is no session left, the algorithm terminates; otherwise, go back to Step 2 for the remaining sessions and remaining network capacity. □

Formally, the MCRadd centralized algorithm is stated as following.

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, $\ell \in \mathcal{L}^k$.
2. $a^k := \min\{\min_{\ell \in \mathcal{L}^k} \frac{(C_\ell - F_\ell^{k-1})}{n_\ell^k}, \ \min_{s \in \mathcal{S}^k}(\text{PCR}_s - r_s^{k-1})\}$.
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. $\mathcal{S}^{k+1} := \{s \mid s \text{ does not traverse any link in } (\mathcal{L} - \mathcal{L}^{k+1}) \text{ and } r_s^k \neq \text{PCR}_s\}$.
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. □

The following example illustrates how Algorithm 1 allocates network bandwidth such that the MCRadd fairness policy is satisfied.

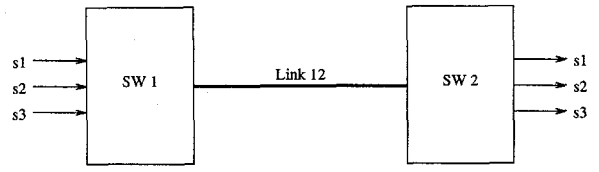

Figure 1: The peer-to-peer network configuration.

Session	MCR	PCR	MCRadd Rate Allocation
$s1$	0.15	0.30	0.30
$s2$	0.10	0.35	0.35
$s3$	0.05	0.50	0.35

Table 1: MCR requirement, PCR constraint, and MCRadd rate allocation for each session in the peer-to-peer network configuration.

Example 1 Peer-to-Peer Configuration

In this network configuration (Fig. 1), the output port link of SW1 (Link 12) is the only bottleneck link for all VC sessions. Assume that all links are of unit capacity. The MCR requirement and PCR constraint for each session are listed in Table 1. Here, we give the major steps in using Algorithm 1 to allocate network bandwidth for each session.

- Step 1: We start the rate allocation for each session with its MCR requirement. That is, we start the rate for $s1$, $s2$, and $s3$ with 0.15, 0.10, and 0.05, respectively. The remaining capacity of Link 12 is now 0.7.
- Step 2: The minimum increment for each session before Link 12 saturates or a session first reaches its PCR constraint is $a^1 := \min\{\frac{0.7}{3}, \min\{0.15, 0.25, 0.45\}\} = 0.15$. Here session $s1$ first reaches its PCR constraint of 0.3.
- Step 3: Remove $s1$ (with a rate of 0.3) out of future iterations and we now have the rate of 0.25 and 0.20 for $s2$ and $s3$, respectively, with a remaining capacity of 0.25 on Link 12.
- Step 4: Now the minimum increment for $s2$ and $s3$ before Link 12 saturates or a session first reaches its PCR constraint is $a^2 := \min\{\frac{0.25}{2}, \min\{0.10, 0.30\}\} = 0.10$. Here, session $s2$ reaches its PCR constraint first.
- Step 5: Remove $s2$ (with a rate of 0.35) out of future iteration and we now have the rate of 0.3 for $s3$ with remaining capacity of 0.05 on Link 12.
- Step 6: Increase the rate of $s3$ up to 0.35 and Link 12 saturates before $s3$ reaches its PCR (0.5). The final rate assignments are 0.30, 0.35, and 0.35 for $s1$, $s2$ and $s3$, respectively and satisfy the MCRadd fair rate allocation policy. □

Although the definitions and centralized algorithms presented in this section are essential for our understanding on how MCRadd policy works to perform network-wide bandwidth assignment, they cannot be applied directly to a distributed traffic management environment for ABR service. To show the practical merit of implementing the MCRadd rate allocation policy for ABR service, we will develop an explicit rate (ER)-based ABR algorithm conforming to the ATM Forum traffic management specifications [1] in the next section.

3 A Distributed ABR Implementation

A generic closed-loop rate-based flow control for an ABR virtual connection is shown in Fig. 2. Resource Management (RM) cells are inserted periodically among ATM data cells to convey network congestion and available bandwidth information to the source. RM cells contain important information such as the source's allowed cell rate (ACR) (called the current cell rate (CCR) in the RM cell's field), MCR requirement, explicit rate (ER), congestion indication (CI) bit and no increase (NI) bit. A transit node and destination end system (DES) may set the ER field, CI and NI bits in RM cells. All RM cells of an ABR virtual connection are turned back towards its source after arriving at the destination. Upon receiving backward RM cells, the source adjusts its ACR accordingly.

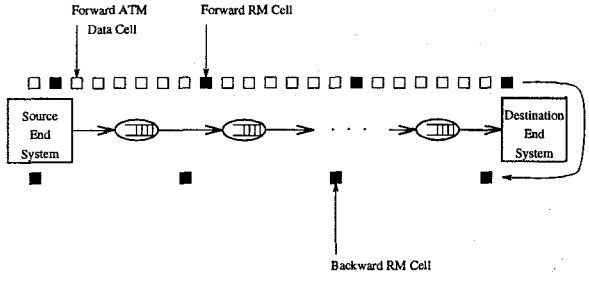

Figure 2: Closed-loop rate-based flow control for an ABR virtual connection.

We first specify the source behavior of our ABR algorithm [1].

Algorithm 2 ABR Source Behavior

- The source starts to transmit at ACR := ICR, which is greater than or equal to its MCR;
- For every N_{rm} transmitted ATM data cells, the source sends a forward RM(CCR, MCR, ER) cell with CCR := ACR, MCR := MCR, and ER := PCR;
- Upon the receipt a backward RM(CCR, MCR, ER) cell from the destination, the ACR at source is adjusted to: ACR := ER. □

The destination end system simply returns every RM cell back towards the source upon receiving it.

The ATM Forum has not specified the switch behavior for ABR service and intends to leave its implementation to vendors. In the following, we present our ABR switch algorithm to achieve the MCRadd rate allocation policy.

3.1 The ABR Switch Algorithm

Our ABR implementation for MCRadd rate allocation policy is motivated by the work by Charny *et al.* [4]. The *Consistent Marking* technique in [4] emulates the centralized rate allocation algorithm for max-min fairness policy through distributed and asynchronous iterations and is general enough for a broad class of distributed algorithms for rate allocation policies. We will incorporate the Consistent Marking technique to design our ER-based ABR switch algorithm and achieve the MCRadd rate allocation policy with MCR/PCR support.

The switch keeps track of each VC's state information (so-called per-VC accounting). Specifically, for each RM cell traversing this link, the switch records the CCR and MCR for each VC and performs the switch algorithm (Algorithm 4) at this link. Each link $\ell \in \mathcal{L}$ maintains a variable called advertised rate, μ_ℓ, which is used to estimate the MCRadd-bottleneck rate at this link.

The following are the link parameters and variables used in our switch algorithm.

C_ℓ: Capacity of link ℓ, $\ell \in \mathcal{L}$.

$\mathcal{G}_\ell$: Set of sessions traversing link ℓ, $\ell \in \mathcal{L}$.

n_ℓ: Number of sessions in $\mathcal{G}_\ell$, $\ell \in \mathcal{L}$, i.e., $n_\ell = |\mathcal{G}_\ell|$.

r_ℓ^i: CCR value of session $i \in \mathcal{G}_\ell$ at link ℓ.

MCR^i: MCR requirement of session i.

b_ℓ^i: Bit used to mark session $i \in \mathcal{G}_\ell$ at link ℓ.

$$b_\ell^i = \begin{cases} 1 & \text{if session } i \in \mathcal{G}_\ell \text{ is marked at link } \ell; \\ 0 & \text{otherwise.} \end{cases}$$

$\mathcal{Y}_\ell$: Set of sessions marked at link ℓ, i.e. $\mathcal{Y}_\ell = \{i \mid i \in \mathcal{G}_\ell \text{ and } b_\ell^i = 1\}$.

$\mathcal{U}_\ell$: Set of sessions unmarked at link ℓ, i.e. $\mathcal{U}_\ell = \{i \mid i \in \mathcal{G}_\ell \text{ and } b_\ell^i = 0\}$ and $\mathcal{Y}_\ell \cup \mathcal{U}_\ell = \mathcal{G}_\ell$.

μ_ℓ: Advertised MCRadd-bottleneck link rate at link ℓ, calculated as follows:

Algorithm 3 μ_ℓ **Calculation**

$$\mu_\ell := \begin{cases} C_\ell & \text{if } n_\ell = 0; \\ C_\ell - \sum_{i \in \mathcal{G}_\ell} r_\ell^i + \max_{i \in \mathcal{G}_\ell}(r_\ell^i - \text{MCR}^i) & \text{if } n_\ell = |\mathcal{Y}_\ell|; \\ \dfrac{(C_\ell - \sum_{i \in \mathcal{G}_\ell} \text{MCR}^i) - \sum_{i \in \mathcal{Y}_\ell}(r_\ell^i - \text{MCR}^i)}{|\mathcal{U}_\ell|} & \text{otherwise.} \end{cases}$$

□

Link initialization: $\mathcal{G}_\ell = \emptyset$; $n_\ell = 0$; $\mu_\ell = C_\ell$.

Algorithm 4 ABR Switch Behavior

Upon the receipt of a forward RM(CCR, MCR, ER) cell from the source of session i {
  if RM cell signals session termination[2]{
    $\mathcal{G}_\ell := \mathcal{G}_\ell - \{i\}$;
    $n_\ell := n_\ell - 1$;
    table_update();
    }
  if RM cell signals session initiation {
    $\mathcal{G}_\ell := \mathcal{G}_\ell \cup \{i\}$;
    $n_\ell := n_\ell + 1$;
    $b_\ell^i := 0$; $r_\ell^i :=$ CCR; $\text{MCR}^i :=$ MCR;
    table_update();
    }
  else /* i.e. RM cell belongs to a session already in $\mathcal{G}_\ell$. */ {
    $r_\ell^i :=$ CCR;
    if $((r_\ell^i - \text{MCR}^i) \le \mu_\ell)$ then $b_\ell^i := 1$;
    table_update();
    }

  Forward RM(CCR, MCR, ER) towards its destination;
}

Upon the receipt of a backward RM(CCR, MCR, ER) cell from the destination of session i {
  ER := min{ER, $\mu_\ell + \text{MCR}^i$};
  Forward RM(CCR, MCR, ER) towards its source;
}

table_update()
{
  rate_calculation_1: use Algorithm 3 to calculate μ_ℓ^1;

  Unmark any session $i \in \mathcal{G}_\ell$ at link ℓ with $r_\ell^i - \text{MCR}^i > \mu_\ell^1$;

  rate_calculation_2: use Algorithm 3 to calculate μ_ℓ again;[3]
}

□

In the following, we give an outline of the proof that the rate allocation for each session calculated by the above distributed ABR algorithm converges to the MCRadd rate allocation policy through distributed and asynchronous iterations.

3.2 Convergence Theorem

We first give the following definition for *marking-consistent.*

Definition 3 Let $\mathcal{Y}_\ell$ be the set of sessions that are marked at link $\ell \in \mathcal{L}$ and μ_ℓ be calculated according to

[2] This information is conveyed through some unspecified bits in the RM cell, which can be set either at the source or the UNI.

[3] Both μ_ℓ^1 and μ_ℓ follow the same rate calculation in Algorithm 3.

Algorithm 3. The marking of sessions at link $\ell \in \mathcal{L}$ is *marking-consistent* if $r_\ell^i - \text{MCR}^i \leq \mu_\ell$ for every session $i \in \mathcal{Y}_\ell$. □

The proof of convergence is based on the following three key lemmas. Due to the space limitation, we will only present the statements of these lemmas with the intent of giving a sketch of the overall proof of our convergence theorem. Interested readers should refer to [6] for detailed proofs of these lemmas.

Lemma 1 After the switch algorithm is performed for each RM cell traversing a link, the marking of sessions at this link is marking-consistent. □

Let M be the total number of iterations needed to execute Algorithm 1. Let $\mathcal{S}_i$, $1 \leq i \leq M$ be the set of sessions being removed at the end of the ith iteration, i.e. sessions in $\mathcal{S}_i$ have either reached the MCRadd-bottleneck link rate or their PCRs during the ith iteration of Algorithm 1. Let $\mathcal{L}_i$, $1 \leq i \leq M$ be the set of links traversed by sessions in $\mathcal{S}_i$. Note that $\mathcal{S}_1$, $\mathcal{S}_2$, ..., $\mathcal{S}_M$ are mutually exclusive and the sum of $\mathcal{S}_1$, $\mathcal{S}_2$, ..., $\mathcal{S}_M$ is $\mathcal{S}$ while $\mathcal{L}_1$, $\mathcal{L}_2$, ..., $\mathcal{L}_M$ may be mutually inclusive. That is, there may be links belonging to both $\mathcal{L}_i$ and $\mathcal{L}_{i+1}$. This happens when sessions in $\mathcal{S}_i$ reach their PCRs before saturating link $\ell \in \mathcal{L}_i$ and link $\ell \in \mathcal{L}_i$ becomes part of $\mathcal{L}_{i+1}$.

Let τ_i, $1 \leq i \leq M$ be defined as following:

$$\tau_i = r^s - \text{MCR}^s \quad \text{for every } s \in \mathcal{S}_i, \quad 1 \leq i \leq M.$$

where r^s is the final MCRadd rate allocation for session s by Algorithm 1. By the operation of Algorithm 1, for a session which has not yet gone through a saturated link or reached its PCR, its rate allocation minus its MCR increases at each iteration. Therefore, we have the following property for τ_i, $1 \leq i \leq M$.

$$\tau_1 < \tau_2 < \ldots < \tau_M.$$

Lemma 2 Base Case

There exists a $T_1 \geq 0$ such that:

i) If $\tau_1 = \frac{(C_\ell - \sum_{i \in \mathcal{G}_\ell} \text{MCR}^i)}{n_\ell} \leq (\text{PCR}^s - \text{MCR}^s)$ for $s \in \mathcal{S}_1$, i.e., the MCRadd-bottleneck link rate is reached before some session $s \in \mathcal{S}_1$ reaches its PCR, then for $t \geq T_1$, the following statements hold.

1. $\mu_\ell = \tau_1$ for every link $\ell \in \mathcal{L}_1$.
2. The ER field of every returning RM cell of session $i \in \mathcal{S}_1$ satisfies $\text{ER} = \tau_1 + \text{MCR}^i$.
3. The ACR at source for every session $i \in \mathcal{S}_1$ satisfies $\text{ACR} = \tau_1 + \text{MCR}^i$.
4. $b_\ell^i = 1$, $r_\ell^i = \tau_1 + \text{MCR}^i$ for every session $i \in \mathcal{S}_1$ and every link ℓ traversed by session $i \in \mathcal{S}_1$.
5. The ER field of every returning RM cell of session $j \in (\mathcal{S} - \mathcal{S}_1)$ satisfies $\text{ER} > \tau_1 + \text{MCR}^j$.
6. The ACR at source for every session $j \in (\mathcal{S} - \mathcal{S}_1)$ satisfies $\text{ACR} > \tau_1 + \text{MCR}^j$.
7. The recorded CCR of session $j \in (\mathcal{S} - \mathcal{S}_1)$ satisfies $r_\ell^j > \tau_1 + \text{MCR}^j$ at every link ℓ traversed by session j.

ii) If $\tau_1 = (\text{PCR}^s - \text{MCR}^s) < \frac{(C_\ell - \sum_{i \in \mathcal{G}_\ell} \text{MCR}^i)}{n_\ell}$ for $s \in \mathcal{S}_1$, i.e., some session $s \in \mathcal{S}_1$ reaches its PCR before the MCRadd-bottleneck link rate is reached, then for $t \geq T_1$, the following statements hold.

1. $\mu_\ell > \tau_1$ for every link $\ell \in \mathcal{L}_1$.
2. The ER field of every returning RM cell of session $i \in \mathcal{S}_1$ satisfies $\text{ER} = \text{PCR}^i$.
3. The ACR at source for every session $i \in \mathcal{S}_1$ satisfies $\text{ACR} = \text{PCR}^i$.
4. $b_\ell^i = 1$, $r_\ell^i = \text{PCR}^i$ for every session $i \in \mathcal{S}_1$ and every link ℓ traversed by session $i \in \mathcal{S}_1$.
5. — 7. Same as statements i)–5 to i)–7, respectively. □

The result of Lemma 2 will now be used as the base case for induction on the index i of $\mathcal{S}_i$. Note that Lemma 2 states that not only session $p \in \mathcal{S}_1$ has reached its optimal rate of $\tau_1 + \text{MCR}^p$ (in case i) or PCR^p (in case ii), but that its rate will never change and that these sessions will remain marked at all links along their paths.

Lemma 3 Induction

Let M be the total number of iterations to execute Algorithm 1. Suppose for some $1 \leq i \leq M-1$, there exists a $T_i \geq 0$ such that:

i) If $\tau_j < (\text{PCR}^s - \text{MCR}^s)$ for $s \in \mathcal{S}_j$, $1 \leq j \leq i$, i.e., the MCRadd-bottleneck link rate is reached before some session $s \in \mathcal{S}_j$ reaches its PCR, and for $t \geq T_i$, the following statements hold.

1. $\mu_\ell = \tau_j$ for every link $\ell \in \mathcal{L}_j$.
2. The ER field of returning RM cell of session $p \in \mathcal{S}_j$ satisfies $\text{ER} = \tau_j + \text{MCR}^p$.
3. The ACR at source for every session $p \in \mathcal{S}_j$ satisfies $\text{ACR} = \tau_j + \text{MCR}^p$.
4. $b_\ell^p = 1$, $r_\ell^p = \tau_j + \text{MCR}^p$ for every session $p \in \mathcal{S}_j$ and every link ℓ traversed by session $p \in \mathcal{S}_j$.
5. The ER field of every returning RM cell of session $p \in (\mathcal{S} - (\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_i))$ satisfies $\text{ER} > \tau_i + \text{MCR}^p$.
6. The ACR at source for every session $p \in (\mathcal{S} - (\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_i))$ satisfies $\text{ACR} > \tau_i + \text{MCR}^p$.
7. The recorded CCR of session $p \in (\mathcal{S} - (\mathcal{S}_1 \cup \cdots \cup \mathcal{S}_i))$ satisfies $r_\ell^p > \tau_i + \text{MCR}^p$ at every link ℓ traversed by session p.

ii) If $\tau_j = (\text{PCR}^s - \text{MCR}^s)$ for $s \in \mathcal{S}_j$, $1 \leq j \leq i$, i.e., some session $s \in \mathcal{S}_j$ reaches its PCR before the MCRadd-bottleneck link rate is reached, and for $t \geq T_i$, the following statements hold.

1. $\mu_\ell > \tau_j$ for every link $\ell \in \mathcal{L}_j$.
2. The ER field of every returning RM cell of session $p \in \mathcal{S}_j$ satisfies $\text{ER} = \text{PCR}^p$.
3. The ACR at source for every session $p \in \mathcal{S}_j$ satisfies $\text{ACR} = \text{PCR}^p$.
4. $b_\ell^p = 1$, $r_\ell^p = \text{PCR}^p$ for every session $p \in \mathcal{S}_j$ and every link ℓ traversed by session $p \in \mathcal{S}_j$.
5. — 7. Same as statements i)–5 to i)–7, respectively.

Then there exists a $T_{i+1} \geq 0$ such that for $t \geq T_{i+1}$, all statements in i) and ii) hold for $i+1$. □

The following main theorem follows from Lemma 2 and Lemma 3.

Theorem 3 After the number of active sessions in the network stabilizes, the rate allocation for each session by Algorithm 4 converges to the rate calculated by the MCRadd fair rate allocation policy. □

It can be shown that the following corollary holds [6].

Corollary 3.1 Let D be an upper bound on the round-trip delay of all sessions. Then an upper bound on the convergence time to the final MCRadd fair share rate by our ABR algorithm from the time when the number of active sessions in the network stabilizes is given by $2.5MD$. □

4 Simulation Results

Our work in Section 3.2 gives a correctness proof that the rate calculation by our ABR switch algorithm in Section 3.1 converges to the MCRadd fair rate allocation policy through distributed and asynchronous iterations. This gives us a theoretical guarantee that our ABR algorithm will converge to the MCRadd policy under *any* network configuration and *any* set of link distances. In this section, we implement our ABR switch algorithm on our network simulator [5] and perform simulations on a few benchmark network configurations suggested by the ATM Forum Traffic Management Group. The purpose of our work in this section is to have some quantitative insights on the convergence time of our ABR algorithm.

The network configurations that we use are the peer-to-peer network configuration in Fig. 1, the *three-node* configuration in Fig. 4 and the *generic fairness* configuration in Fig. 6. The ATM switches in all the simulations are assumed to have output port buffers with a speedup equal to the number of their ports. The buffer of each output port of a switch employs the simple FIFO queueing discipline and is shared by all VCs going through that port. At each output port of an ATM switch, we implement our ABR switch algorithm.

End System	PCR	PCR
	MCR	MCR
	ICR	MCR
	Nrm	32
Link	Speed	150 Mbps
Switch	Cell Switching Delay	4 μs

Table 2: Simulation parameters.

Table 2 lists the parameters used in our simulation. The link capacity is 150 Mbps. For stability, we set the target link utilization to be 0.95. That is, we set $C_\ell = 0.95 \times 150$ Mbps $= 142.5$ Mbps at every link $\ell \in \mathcal{L}$ for the ER calculation. The distance from source/destination to the switch is 1 km and the link distance between ATM switches is 1000 km (corresponding to a wide area network) and we assume that the propagation delay is 5 μs per km.

The Peer-to-Peer Network Configuration

For this configuration (Fig. 1), the output port link of SW1 is the only bottleneck node for all sessions.

Fig. 3 shows the ACR at source for sessions s1, s2 and s3, respectively. The cell rates shown in the plot are normalized with respect to the capacity C_ℓ (142.5 Mbps) for easy comparison with those values obtained with our centralized algorithm under unit link capacity (Table 1). After initial iterations, we see that the cell rate of each session converges to the final rate listed in Table 1.

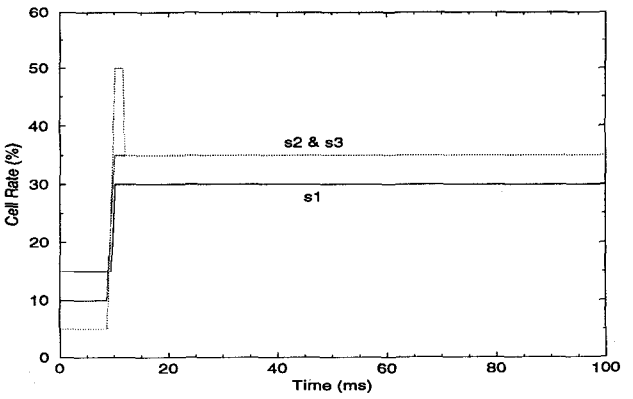

Figure 3: The cell rates of all connections for the MCRadd policy in the peer-to-peer network configuration.

The convergence time of our ABR algorithm is much faster than the upper bound given in Corollary 3.1. Here the round trip time (RTT) is 10 ms and it takes less than 2 RTT for our ABR algorithm to converge to the final rates.

The Three-Node Network Configuration

For this configuration (Fig. 4), the output port links of SW1 and SW2 are potential MCRadd-bottleneck links for VC sessions. The MCR requirement and PCR constraint for each session are listed in Table 3.

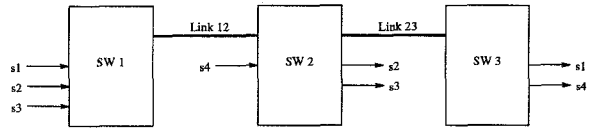

Figure 4: The three-node network configuration.

Session	MCR	PCR	MCRadd Rate Allocation
$s1$	0.05	0.20	0.20
$s2$	0.10	0.70	0.45
$s3$	0.15	0.35	0.35
$s4$	0.20	1.00	0.80

Table 3: MCR requirement, PCR constraint, and MCRadd rate allocation for each session in the three-node network configuration.

Fig. 5 shows the normalized cell rate of each session under our ABR algorithm. Comparing with the rates obtained by our centralized algorithm in Table 3, we find that after the initial transient period, the rate allocation through the distributed ABR implementation converges to the final rates listed in Table 3.

Here the maximum RTT is 20 ms for $s1$ and it takes less than 2 RTT for the rate allocation to converge to the optimal rates.

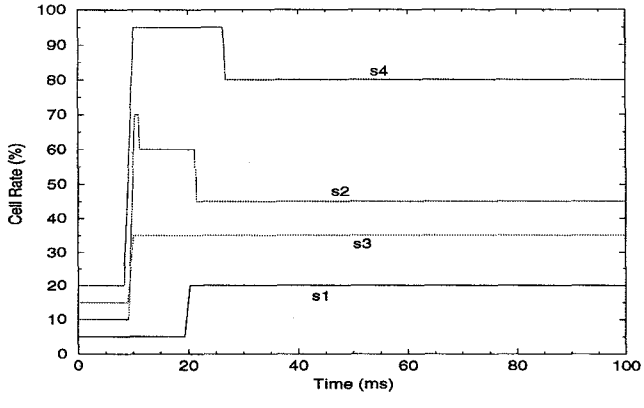

Figure 5: The cell rates of all connections for the MCRadd policy in the three-node network configuration.

The Generic Fairness Network Configuration

The generic fairness configuration that we use is shown in Fig. 6 where there are 5 ATM switches connected in a chain with 6 session paths traversing these ATM switches and sharing link capacities [3].

Table 4 lists the MCR requirement and PCR constraint for each session, as well as rate allocation for each session under the MCRadd policy.

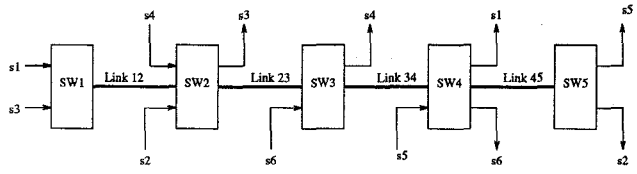

Figure 6: The generic fairness network configuration.

Session	MCR	PCR	MCRadd Rate Allocation
$s1$	0.05	0.50	0.30
$s2$	0.15	0.30	0.30
$s3$	0.10	1.00	0.70
$s4$	0.05	0.60	0.40
$s5$	0.20	0.60	0.60
$s6$	0.30	0.40	0.40

Table 4: MCR requirement. PCR constraint, and MCRadd rate allocation for each session in the generic fairness network configuration.

Fig. 7 shows the normalized cell rate of each VC session under our distributed ABR implementation. Again, they converge to the rates listed in Table 4. Here the maximum RTT among all sessions is 30 ms ($s1$ and $s2$) and its takes less than 2 RTT for our algorithm to converge to the optimal rate for each session.

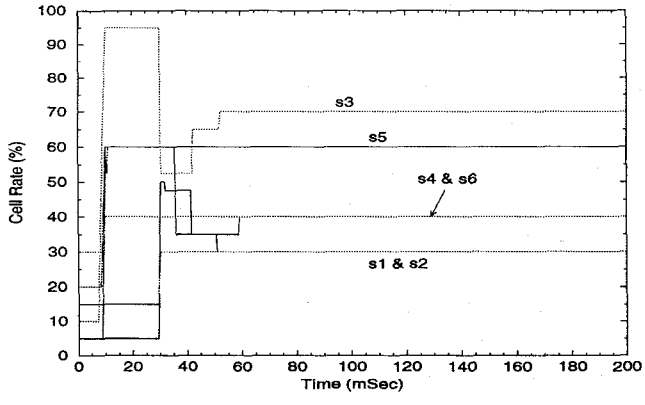

Figure 7: The cell rates of all connections for the MCRadd policy in the generic fairness network configuration.

In summary, based on the simulation results in this section, we have demonstrated that our distributed ABR algorithm achieves the MCRadd rate allocation policy with fast convergence time.

5 Concluding Remarks

We have defined a network bandwidth assignment policy to support both the MCR requirement and PCR constraint for ABR service in ATM networks. A centralized algorithm to compute network bandwidth assignment for MCRadd is also presented, as well as its correctness proof.

We have developed an ER-based ABR algorithm in the context of the ATM Forum ABR traffic management framework to achieve the MCRadd policy. We gave a proof that the final rate allocation converges to the MCRadd fair rate allocation through distributed and asynchronous iterations. Our proof gave a theoretical guarantee that our ABR algorithm converges to the MCRadd policy for any network configuration and any set of link distances. Simulation results based on benchmark network configurations used by the ATM Forum demonstrated the fast convergence property of our ABR algorithm.

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] F. Bonomi and K. W. Fendick, "The Rate-Based Flow Control Framework for the Available Bit Rate ATM Service," *IEEE Network,* vol. 9, no. 2, Mar./Apr. 1995, pp.25-39.

[4] A. Charny, D. Clark, and R. Jain, "Congestion Control with Explicit Rate Indication," *Proc. IEEE ICC'95,* pp. 1954-1963.

[5] A. Heybey, "The Network Simulator – Version 2.1," *Laboratory of Computer Science,* Massachusetts Institute of Technology, September 1990.

[6] Y. T. Hou, H. Tzeng, and S. S. Panwar, "The MCRadd Rate Allocation Policy for ABR Service," *Tech. Report CATT 97-104,* Center for Advanced Technology in Telecommunications, Polytechnic University, Brooklyn, NY, Jan. 20, 1997.

[7] Y. T. Hou, H. Tzeng, and V. P. Kumar, "On Rate Allocation Policies with Minimum Cell Rate Guarantee for ABR Service in ATM Networks," *Proc. 15th International Teletraffic Congress (ITC-15),* Washington, D.C., June 23-27, 1997.

[8] D. Hughes, "Fair Share in the Context of MCR," *ATM Forum Contribution 94-0977,* October 1994.

[9] R. Jain, *et al.,* "ERICA Switch Algorithm: A Complete Description," *ATM Forum Contribution, 96-1172,* August 1996.

[10] L. Kalampoukas, A. Varma, and K. K. Ramakrishnan, "Dynamics of an Explicit Rate Allocation Algorithm for Available Bit Rate (ABR) Service in ATM Networks," *Proc. 6th IFIP Int. Conf. High Performance Networking (HPN '95),* Sept. 1995, pp.143-154.

[11] L. Roberts, "Enhanced PRCA (Proportional Rate Control Algorithm," *ATM Forum Contribution 94-0735R1,* August 1994.

[12] 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.

[13] 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.