Generalized Max-Min Rate Allocation: Theory and a Simple Implementation

Yiwei Thomas Hou, Shivendra S. Panwar, and Henry Tzeng

Abstract: An important concept in the available bit rate (ABR) service model is the minimum cell rate (MCR) guarantee as well as the peak cell rate (PCR) constraint for each flow. Due to the MCR and PCR requirements, the well-known max-min rate allocation policy no longer suffices to determine the rate allocation for each flow since it does not support either MCR or PCR. In this paper, we present a generalized max-min (GMM) rate allocation policy, which supports both the MCR and PCR requirements for each flow. Furthermore, a simple distributed algorithm using the ABR flow control protocol is developed to achieve the GMM rate allocation in a distributed network environment. The effectiveness of this distributed algorithm is demonstrated by simulation results.

Index Terms: Max-min rate allocation, available bit rate, minimum rate, peak rate, flow control.

I. INTRODUCTION

One key performance objective for flow-oriented packet-switched networks is to optimally share network bandwidth among all traffic flows. Optimality here typically includes two components: 1) each flow is entitled to as much network bandwidth as any other flow (i.e., fairness), and 2) the network bandwidth is utilized efficiently.

The classical max-min rate allocation has been widely regarded as an optimal bandwidth sharing policy among traffic flows in the network [1]. In ATM networks, the max-min rate allocation has been used for the available bit rate (ABR) service [2]. By the specifications in [2], on the establishment of an ABR flow, the user shall specify to the network both a minimum rate and a maximum rate, designated as minimum cell rate (MCR) and peak cell rate (PCR), respectively, for the requested flow. The source starts to transmit at an initial cell rate (ICR), which is greater than or equal to MCR, and may increase its rate up to PCR depending upon congestion and bandwidth information from the network.

Since the classical max-min does not address how to determine rate allocation for each flow when there are MCR and PCR constraints, an MCR-offsetted and an MCR-weighted max-min policies were proposed in [3], [4].

In this paper, we generalize the classical max-min rate allocation. We present a generalized max-min (GMM) rate allocation policy with both the minimum rate guarantee and the peak rate constraint for each flow. We also present a centralized bandwidth assignment algorithm to achieve the GMM policy and prove its correctness.

Based on the centralized theory for the GMM policy, we develop a distributed algorithm to achieve GMM rate allocation. Our distributed algorithm employs the ABR flow control protocol and is based on the Intelligent Marking technique by Siu and Tzeng [5], [6] which was developed for the classical max-min. We use simulation results to demonstrate the effectiveness of our distributed algorithm to achieve GMM rate allocation.

The remainder of this paper is organized as follows. In Section II, we present the theory of GMM rate allocation to generalize the classical max-min with both the minimum rate guarantee and the peak rate constraint for each flow. In Section III, we develop a distributed algorithm using ABR flow control protocol to achieve the GMM policy. In Section IV, we present simulation results to demonstrate the effectiveness of the distributed algorithm. Section V concludes this paper.

II. GENERALIZED MAX-MIN RATE ALLOCATION

We organize this section as follows. In Section II-A, we briefly summarize key results for the classical max-min rate allocation [1]. In Section II-B, we present the GMM rate allocation, which generalizes the classical max-min. A centralized algorithm to determine rate allocation for the GMM policy is then presented in Section II-C.

A. Preliminaries

In our model, a network \(\mathcal{N}\) is characterized by a set of links \(\mathcal{L}\) and flows \(\mathcal{S}\). Each flow \(s \in \mathcal{S}\) traverses one or more links in \(\mathcal{L}\) and is allocated a specific rate \(r^s\). Denote \(\mathcal{S}_\ell\) the set of flows traversing link \(\ell\). Then the (aggregate) allocated rate \(F_\ell\) on link \(\ell \in \mathcal{L}\) of the network is \(F_\ell = \sum_{s \in \mathcal{S}_\ell} r^s\).

Let \(C_\ell\) be the capacity of link \(\ell\). A link \(\ell\) is saturated if \(F_\ell = C_\ell\). A rate vector \(r = \{r^s | s \in \mathcal{S}\}\) is feasible if the following two constraints are satisfied: 1) \(r^s \geq 0\) for all \(s \in \mathcal{S}\); and 2) \(F_\ell \leq C_\ell\) for all \(\ell \in \mathcal{L}\).

A rate vector \(r\) is \(max-min\) if it is feasible, and for each flow \(s\), one cannot generate a new feasible rate vector by increasing the allocated rate \(r^s\) without decreasing the allocated rate of some other flow \(t\) with a rate \(r^t\) already less than or equal to \(r^s\) in the rate vector \(r\). More formally, we have the following.
Definition 1: A rate vector \( r \) is max-min if it is feasible, and for each \( s \in S \) and every feasible rate vector \( \hat{r} \) in which \( r^s > r^s \), there exists some flow \( t \in S \) such that \( r^t \geq r^t \) and \( r^t > r^t \).

Definition 2: Given a feasible rate vector \( r \), a link \( \ell \in L \) is a bottleneck link with respect to \( r \) for a flow \( s \) traversing \( \ell \) if \( F_\ell = C_\ell \) and \( r^s \geq r^\ell \) for all flows \( t \) traversing link \( \ell \).

Theorem 1: A feasible rate vector \( r \) is max-min if and only if each flow has a bottleneck link with respect to \( r \).

Algorithm 1: The following iterative steps describe the centralized algorithm to determine rate allocation for each flow for the classical max-min policy.

1. Start the rate allocation of each flow with zero.
2. Increase the rate of all flows with the smallest rate such that some link becomes saturated.
3. Remove those flows that traverse saturated links and their associated bandwidth from the network.
4. If there is no flow left, the algorithm terminates; otherwise, go back to Step 2 for the remaining flows and remaining network capacity.

Theorem 2: There exists a unique rate vector that satisfies the max-min rate allocation.

A proof of Theorem 2 is given in the Appendix.

B. Generalized Max-Min Rate Allocation

We are now ready to formally define the generalized max-min rate allocation policy with minimum rate and peak rate constraints for each flow.

Let \( \text{MCR}^s \) and \( \text{PCR}^s \) be the minimum rate requirement and the peak rate constraint for each flow \( s \in S \). For the sake of feasibility, we make the following assumption.

Assumption 1: The sum of all the flows' MCR requirements traversing any link does not exceed the link's capacity, i.e., \( \sum_{s \in S} \text{MCR}^s \leq C_\ell \) for every \( \ell \in L \).

This assumption is enforced by admission control at call setup time to determine whether or not to accept a new flow.

Definition 3: A rate vector \( r = \{r^s | s \in S\} \) is ABR-feasible if the following two constraints are satisfied:

\[
\text{MCR}^s \leq r^s \leq \text{PCR}^s, \quad \text{for all } s \in S,
\]

\[
F_\ell \leq C_\ell, \quad \text{for all } \ell \in L.
\]

Before we give a definition for the GMM rate allocation, we use the following simple example to illustrate its basic concept.

In the peer-to-peer network configuration (Fig. 1), the output port link of SW1 (Link12) is the only bottleneck link for all flows. Assume that all links have the same normalized one unit of capacity. When there is no MCR and PCR requirements for each flow, using the centralized max-min algorithm described in Algorithm 1, we allocate to each flow with rate of \( \frac{1}{3} \). Now let the MCR requirement and PCR constraint for each flow be as listed in Table 1. It is clear that the classical max-min is no longer applicable here because it does not support either the MCR or the PCR.

In the following, we describe the iterative steps of the centralized algorithm to determine rate allocation for each flow under the generalized max-min policy, which we will formally define (Definition 4) shortly.

Algorithm 2: This algorithm describes how to determine the rate allocation for each flow under GMM.

1. Start the rate of each flow with its MCR.
2. Identify the flow with the smallest rate among all the remaining flows and increase the rate of such flow(s) until one of the following events first takes place:
   - the rate of such flow(s) reaches the second smallest rate among all the remaining flows;
   - some link saturates;
   - the rate of such flow(s) reaches its PCR.
3. If some link saturates or such flow reaches its PCR in Step 3, remove such flows that either traverse this saturated link or reach their PCRs, respectively, as well as the network capacity associated with such flows from the network.
4. If there is no flow left, the algorithm terminates; otherwise, go back to Step 2 for the remaining flows and network capacity.

With the above centralized algorithm for the GMM rate allocation, we are able to complete the rate allocation problem for the peer-to-peer network configuration (Fig. 1) with the MCR and PCR requirements listed in Table 1.

Example 1: (A Peer-to-Peer Network) Using Algorithm 2, we list the rate allocation for each flow at each iteration in Table 2, which are explained briefly as follows. A graphical display of the iterative steps in Table 2 is also shown in Fig. 2.

Step 1 – As shown in Fig. 2 and Table 2, we start the rate allocation for each flow with its MCR requirement (shown in the darkest shaded areas in Fig. 2).

Step 2 – Since the rate of \( s3 \) (0.05) is the smallest among all flows, we increase it until it reaches the second smallest rate, which is 0.1 (\( s2 \)).

Step 3 – The rates of both \( s2 \) and \( s3 \) being 0.1, we increase them together until \( s2 \) reaches its PCR constraint of 0.25.

\footnote{A formal mathematical description of this algorithm will be given in Section II-C.}
Table 2. Iterations of centralized rate allocation algorithm for the peer-to-peer network configuration.

<table>
<thead>
<tr>
<th>Iterations</th>
<th>Flow(MCR, PCR)</th>
<th>Remaining Link Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>s1(0.40, 1.00)</td>
<td>s2(0.10, 0.25)</td>
</tr>
<tr>
<td>Initialization</td>
<td>0.40</td>
<td>0.10</td>
</tr>
<tr>
<td>1st</td>
<td>0.40</td>
<td>0.10</td>
</tr>
<tr>
<td>2nd</td>
<td>0.40</td>
<td>0.25</td>
</tr>
<tr>
<td>3rd</td>
<td>0.40</td>
<td>0.35</td>
</tr>
</tbody>
</table>

Fig. 2. Graphical display of rate allocation for each flow at each iteration in the peer-to-peer example.

Step 4 — Remove s2 (with a rate of 0.25) out of future iterations and we now have the rates of 0.40 and 0.25 for s1 and s3, respectively, with a remaining capacity of 0.10 on Link 12.

Step 5 — Since s3 has a smaller rate (0.25) than s1 (0.4), we increase the rate of s3 to 0.35 and Link 12 saturates. The final rate assignments are 0.40, 0.25, and 0.35 for s1, s2, and s3, respectively.

Informally, a rate vector r is GMM if it is ABR-feasible, and for each flow s, one cannot generate a new ABR-feasible rate vector \( \hat{r} \) by increasing the allocated rate \( r_s \) without decreasing the allocated rate of some other flow t with a rate \( r_t \) already less than or equal to \( r_t^i \) in the rate vector \( r \). Formally, the GMM rate allocation is defined as follows.

**Definition 4:** A rate vector \( r \) is Generalized Max-Min (GMM) if it is ABR-feasible, and for every \( s \in S \) and every ABR-feasible rate vector \( \hat{r} \) in which \( r_s^i > r_s^\hat{r} \), there exists some flow \( t \in S \) such that \( r_t^s \geq r_t^\hat{r} \), and \( r_t^s > r_t^\hat{r} \).

Since there are MCR and PCR requirements for each flow, we define a new notion of bottleneck link as follows.

**Definition 5:** Given an ABR-feasible rate vector \( r \), a link \( \ell \in L \) is an GMM-bottleneck link with respect to \( r \) for a flow \( s \) traversing \( \ell \) if \( F_{\ell} = C_{\ell} \) and \( r_s^\ell \geq r_t^\ell \) for every flow \( t \) traversing link \( \ell \) for which \( r_t^s > MCR^i \).

**Theorem 3:** An ABR-feasible rate vector \( r \) is GMM if and only if each flow has either an GMM-bottleneck link with respect to \( r \) or a rate assignment equal to its PCR.

For a proof of Theorem 3, see the Appendix.

In Example 1, Link 12 is an GMM-bottleneck link for both s1 and s3 (see Definition 5). On the other hand, s1 and s3 have different rate allocations (0.4 for s1 and 0.35 for s3). Thus, it is necessary to have a more precise definition for GMM-bottleneck link rate, which is presented as follows.

Let \( I\{\text{event A}\} \) be the indicator function with the following definition.

\[
I\{\text{event A}\} = \begin{cases} 
1, & \text{if event A is true;} \\
0, & \text{otherwise.}
\end{cases}
\]

**Definition 6:** Given an GMM rate vector \( r \), suppose that link \( \ell \in L \) is an GMM-bottleneck link with respect to \( r \) and let \( \tau_\ell \) denote the GMM-bottleneck link rate at \( \ell \). Then \( \tau_\ell \) satisfies

\[
\tau_\ell \cdot \sum_{r_i \in \mathcal{U}_\ell} I\{MCR^i \leq \tau_\ell\} + \sum_{r_i \in \mathcal{U}_\ell} MCR^i \cdot I\{MCR^i > \tau_\ell\} = C_\ell - \sum_{r_i \in \mathcal{Y}_\ell} r_i^\ell,
\]

where

- \( \mathcal{U}_\ell \) denotes the set of flows that are GMM-bottlenecked at link \( \ell \);
- \( \mathcal{Y}_\ell \) denotes the set of flows that are not GMM-bottlenecked at link \( \ell \).

It is worth pointing out that in the special case when \( MCR^s = 0 \) for every \( s \in S \), the GMM-bottleneck link rate \( \tau_\ell \) in Definition 6 becomes:

\[
\tau_\ell \cdot |\mathcal{U}_\ell| = C_\ell - \sum_{r_i \in \mathcal{Y}_\ell} r_i^\ell,
\]

or

\[
\tau_\ell = \frac{C_\ell - \sum_{r_i \in \mathcal{Y}_\ell} r_i^\ell}{|\mathcal{U}_\ell|},
\]

where \( |\mathcal{U}_\ell| \) denotes the number of flows in \( \mathcal{U}_\ell \). This is exactly the expression for the classical max-min rate allocation at link \( \ell \).

With Definition 6, we are now ready to go back to Example 1. It is now straightforward to see that the GMM-bottleneck link rate at Link 12 is 0.35.

**C. GMM Centralized Algorithm and More Examples**

An intuitive description of the centralized algorithm to determine rate allocation for the GMM was given in Algorithm 2. In the following, we present the formal mathematical description of such a centralized algorithm.

**Algorithm 3:** The following is a formal mathematical description of the centralized algorithm for GMM rate allocation.

Initialization:
Table 3. A second example of GMM rate allocation for the peer-to-peer network.

<table>
<thead>
<tr>
<th>Flow</th>
<th>MCR</th>
<th>PCR</th>
<th>GMM Rate Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>0.40</td>
<td>1.00</td>
<td>0.45</td>
</tr>
<tr>
<td>s2</td>
<td>0.05</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>s3</td>
<td>0.05</td>
<td>0.50</td>
<td>0.45</td>
</tr>
</tbody>
</table>

\[ r^{s,(0)} = \text{MCR}^s, \quad \text{for every } s \in S, \]
\[ F^{(0)} = \sum_{s \in S} \text{MCR}^s, \quad \text{for every } \ell \in L, \]
\[ k = 1, S^{(1)} = S, L^{(1)} = L. \]

1. Sort all the flows in \( S^{(k)} \) into \( m \) sets \((1 \leq m \leq |S^{(k)}|)\):

\[ u_1, u_2, \ldots, u_m, \]

such that 1) each flow in the same set has the same rate; and 2) rate values in these sets are in increasing order, i.e.,
\[ r^s < r^t < \ldots < r^p, \quad \text{where } s \in u_1, t \in u_2, \ldots, p \in u_m. \]

2. \( n^{(k)}_{\ell} := \text{number of flows } s \in u_1 \text{ traversing link } \ell, \text{ for every } \ell \in L^{(k)}. \)

3. \( a^{(k)} := \begin{cases} 
\min_{\ell \text{ traversed by } s \in u_1} \frac{(C_\ell - F^{(k-1)}_{\ell})}{n^{(k)}_{\ell}}, & \text{if } m > 1, \\
\min_{\ell \text{ traversed by } s \in u_1} \frac{(C_\ell - F^{(k-1)}_{\ell})}{n^{(k)}_{\ell}}, & \text{if } m = 1.
\end{cases} \)

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. \( F^{(k)}_{\ell} := \sum_{s \in S} r^{s,(k)}, \quad \text{for every } \ell \in L^{(k)}. \)

6. \( L^{(k+1)} := \{ \ell \mid C_\ell - F^{(k)}_{\ell} > 0, \ell \in L^{(k)} \}. \)

7. \( S^{(k+1)} := \{ s \mid s \text{ does not traverse any link } \ell \in (L - L^{(k+1)}) \text{ and } r^{s,(k)} \neq \text{PCR}^s \}. \)

8. \( k := k + 1. \)

9. If \( S^{(k)} \) is empty, then \( r^{(k-1)} = \{ r^{s,(k-1)} \mid s \in S \} \) is the rate vector satisfying the GMM policy and the algorithm terminates; otherwise, go back to Step 3.

The correctness proof of Algorithm 3 is given in the Appendix. Note that in Step 1 of Algorithm 3, if \( m > 1 \), then the rate values in \( u_2, \ldots, u_m \) are the MCR values of the flows in these sets. Also, starting from the second iteration \((k = 2)\), the sorting procedure in Step 1 only requires minor updates based on the sorted sets in the previous iteration.

It is clear that by Definition 6 and Algorithm 3, the GMM rate assignment for a flow \( s \in S \) can only be one of the following:
- a rate assignment equal to its MCR;
- a rate assignment equal to its GMM-bottleneck link rate;
- a rate assignment equal to its PCR.

Using arguments similar to those in the proofs of Theorems 2 and 3, it can be shown that the following theorem is true.

**Theorem 4:** There exists a unique rate vector that satisfies the GMM rate allocation.

We use the following examples to illustrate how Algorithm 3 allocates network bandwidth such that the GMM rate allocation is satisfied.

**Example 2:** (A Second Example for the Peer-to-Peer Network) This is the same network configuration (Fig. 1) as for Example 1. The MCR and PCR requirements for each flow are listed in Table 3. The rate allocation for each flow at each iteration is listed in Table 4, with a graphical display in Fig. 3. The GMM-bottleneck link rate at Link 12 is 0.45. Since \( s_2 \) is constrained by its PCR, it is equivalent to treating such a flow as being bottlenecked at some other link, e.g. its access link to the network.

We use the following three-node example to further illustrate the concept of GMM-bottleneck link rate. As we shall see, the GMM-bottleneck link rate of each GMM-bottleneck link being reached during the iterations of Algorithm 3 has the property of ascending order.

**Example 3:** (A Three-Node Network) In this network configuration (Fig. 4), the output port links of SW1 (Link 12) and

Table 4. Iterations of the centralized GMM rate allocation algorithm for the second example of the peer-to-peer network.

<table>
<thead>
<tr>
<th>Iterations</th>
<th>Flow(MCR, PCR)</th>
<th>Remaining Link Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1(0.40, 1.00)</td>
<td>s2(0.05, 0.10)</td>
<td>s3(0.05, 0.50)</td>
</tr>
<tr>
<td>Initialization</td>
<td>0.40</td>
<td>0.05</td>
</tr>
<tr>
<td>1st</td>
<td>0.40</td>
<td>0.10</td>
</tr>
<tr>
<td>2nd</td>
<td>0.40</td>
<td>0.40</td>
</tr>
<tr>
<td>3rd</td>
<td>0.45</td>
<td>0.45</td>
</tr>
</tbody>
</table>
Table 5. MCR requirement, PCR constraint, and GMM rate allocation for each flow under the three-node network.

<table>
<thead>
<tr>
<th>Flow</th>
<th>MCR</th>
<th>PCR</th>
<th>GMM Rate Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>0.20</td>
<td>0.50</td>
<td>0.425</td>
</tr>
<tr>
<td>s2</td>
<td>0.05</td>
<td>0.15</td>
<td>0.150</td>
</tr>
<tr>
<td>s3</td>
<td>0.10</td>
<td>0.50</td>
<td>0.425</td>
</tr>
<tr>
<td>s4</td>
<td>0.50</td>
<td>1.00</td>
<td>0.575</td>
</tr>
</tbody>
</table>

Fig. 5. Graphical display of GMM rate allocation for each flow for the GMM at each iteration in the three-node example.

SW2 (Link 23) are the links shared by flows. The MCR requirement and PCR constraint for each flow are listed in Table 5. The rate allocation for each flow at each iteration of Algorithm 3 is listed in Table 6, with a graphical display in Fig. 5.

The GMM-bottleneck link rate at Link 12 is 0.425, which was reached at the end of the 4th iteration, and the GMM-bottleneck link rate at Link 23 is 0.575, which was reached at the end of the 5th iteration, and 0.425 < 0.575. In general, by the operation of Algorithm 3, an GMM-bottleneck link rate obtained at a later iteration for some link is greater than an GMM-bottleneck link rate obtained at an earlier iteration for some other link.

Thus far we have completed the centralized theory of the GMM rate allocation. To demonstrate its practical merit for a distributed network, we design a distributed algorithm using the ABR flow control protocol [2] in the next section.

III. A DISTRIBUTED ALGORITHM FOR GMM RATE ALLOCATION

A schematic for ABR flow control protocol is shown in Fig. 6. 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 current cell rate (CCR) in the RM cell’s field), minimum cell rate (MCR) requirement, explicit rate (ER), congestion indication (CI) bit and no increase (NI) bit. All RM cells of an ABR flow are turned back towards its source after arriving at the destination. A transit node (either along the forward direction or along the backward direction) and destination end system may set the ER field, CI and NI bits in the RM cells.

<table>
<thead>
<tr>
<th>Iterations</th>
<th>Flow(MCR, PCR)</th>
<th>Remaining Link Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>s1 (0.20, 0.50)</td>
<td>s2 (0.05, 0.15)</td>
</tr>
<tr>
<td>Initialization</td>
<td>0.20</td>
<td>0.05</td>
</tr>
<tr>
<td>1st</td>
<td>0.20</td>
<td>0.10</td>
</tr>
<tr>
<td>2nd</td>
<td>0.20</td>
<td>0.15</td>
</tr>
<tr>
<td>3rd</td>
<td>0.20</td>
<td>0.20</td>
</tr>
<tr>
<td>4th</td>
<td>0.425</td>
<td>0.425</td>
</tr>
<tr>
<td>5th</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Upon receiving backward RM cells, the source adjusts its cell generating rate accordingly.

There are two modes of switch algorithms, namely, binary mode and explicit rate (ER) mode. Binary schemes (e.g., explicit forward congestion indication (EFCI) [9]) rely on a single bit feedback to indicate congestion. Due to limited feedback information about network congestion status, the source only knows that either the congestion in the network is present or absent, but doesn’t know how much to increase or decrease its transmission rate. Therefore, the source’s cell rate experiences oscillations. On the other hand, ER schemes employ rate calculation at a switch to estimate available bandwidth and convey this information through the ER field in the returning RM cells. Hence, an ER scheme promises higher efficiency and stability than a binary scheme.

Our switch algorithm for the GMM policy employs the ER mode and is based on the Intelligent Marking technique, originally proposed in [8] and further refined in [5], [6]. The key idea of this technique is to employ several variables at each link of a switch to estimate the max-min bottleneck link rate with a small number of computations. Using the ABR flow control protocol, the ER field of a returning RM cell is set to the minimum of all the estimated bottleneck link rates on all its traversing links, resulting in approximate max-min rate allocation.

Fig. 7 illustrates the switch behavior of the Intelligent Marking technique [5], [6]. Four variables, MCCR (Mean Current Cell Rate), UCR (Upper Cell Rate), EBR (Estimated Bottleneck Rate), and LOAD, are defined for the following purpose: 1) MCCR contains an estimated average cell rate of all flows traversing this link; 2) UCR contains an estimated upper limit of the cell rates of all flows traversing this link; 3) EBR contains an estimated bottleneck link rate; 4) LOAD corresponds to the aggregated cell rate entering the queue normalized with respect to the link capacity and is measured over a period of time. Furthermore, two parameters TLR and $\alpha$ are defined for each link, where the value of TLR is the target load ratio, and $0 < \alpha < 1$.

**Algorithm 4: (Intelligent Marking)**

Upon the receipt of RM(CCR, ER) from the source of a flow if (CCR > MCCR), then

- UCR := UCR + $\alpha$(CCR-UCR);
- MCCR := MCCR + $\alpha$(CCR-MCCR);
- Forward RM(CCR, ER) to its destination.

Upon the receipt of RM(CCR, ER) from the destination of a flow

- EBR := UCR * TLR / LOAD;
- if (QS > QT), then
  - EBR := (QT / QS) * EBR;
- if (ER > EBR), then
  - ER := EBR;
- Forward RM(CCR, ER) to its source.

The Intelligent Marking algorithm is a heuristic algorithm. We can only give an intuitive explanation on how it works. The RM cells from all flows participate in exponential averaging for MCCR with MCCR := MCCR + $\alpha$(CCR - MCCR) while only some flows with greater than MCCR (potentially flows bottlenecked at this link) participate in UCR averaging. EBR is used to estimate max-min bottleneck link rate and is based on UCR and LOAD variables. Since 1) there can be only one bottleneck rate at a link and it is greater than or equal to any of the flow’s rate traversing this link; and 2) the returning RM cell’s ER field is set to the minimum of all the bottleneck link rates along its path, the final rate allocation through Intelligent Marking achieves the max-min rate allocation for each flow (see the “if” part of Theorem 1).

Another interesting fact is that MCCR is larger than the algebraic average of each flow’s CCR traversing this link. This is because MCCR is updated more frequently by those flows with relatively larger CCR than those with relatively smaller CCR traversing the same link.

The most attractive feature of the Intelligent Marking technique is its scalability and low implementation cost. It does not require each link of a switch to maintain the state information of each flow and has $O(1)$ storage requirements and computational complexity.

So far we have given a detailed description of the Intelligent Marking technique, which was designed to achieve the classical max-min without MCR/PCM support. Let’s see how to extend the Intelligent Marking technique for the GMM rate allocation.

Comparing the definitions for the classical max-min (Definition 1) and the GMM (Definition 4), we observe that they are similar—except the additional requirement under GMM that a rate vector must be ABR-feasible (see Definition 3). This motivates us to take the following steps to design a distributed algorithm for the GMM policy.

1. Continue to use Intelligent Marking (Algorithm 4) as

   \footnote{This step is a finer adjustment of the EBR calculation using buffer occupancy information and is not shown in Fig. 7 due to space limitation in the figure. QS is the Queue Size of the output link and QT is a predefined Queue Threshold.}
switch algorithm for the GMM policy. This will also satisfy the second requirement for the ABR-feasibility (\( F_\ell \leq C_\ell \) for all \( \ell \in \mathcal{L} \)) due to the nature of Intelligent Marking, i.e., queue size is always kept finite.

2. Let each ABR source enforce the first requirement of ABR-feasibility, i.e., \( MCR^s \leq r^s \leq PCR^s \) for all \( s \in S \).

The following algorithm specifies the source behavior of our distributed algorithm.

**Algorithm 5**: (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; ER := PCR; Upon the receipt a backward RM(CCR, MCR, ER) from the destination, the ACR at source is adjusted to:

\[
ACR := \max \{ \min \{ (ACR + AIR), ER, PCR \}, MCR \}.
\]

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

### IV. SIMULATION RESULTS

In this section, we implement our distributed algorithm on our network simulator and perform simulations to demonstrate its effectiveness in achieving the GMM policy. The switches in all the simulations are assumed to have output buffers with a speedup equal to the number of their ports. The buffer of each output port of a switch employs the simple FIFO queuing discipline and is shared by all flows going through that port.

In all of our simulations, we assume persistent sources. The network configurations that we use are the peer-to-peer and three-node network configurations shown in Figs. 1 and 4, respectively, and the parking-lot (Fig. 11) network configuration.

Table 7 lists the parameters used in our simulation. The distance from source/destination to the switch is 100 m and the link distance between switches is 10 km (within the scope of a regional enterprise network).

#### A. The Peer-to-Peer Network

In this network configuration (Fig. 1), the output port link of SW1 (Link 12) is the only bottleneck link for all flows.

#### Table 7. Simulation parameters.

<table>
<thead>
<tr>
<th>End system</th>
<th>PCR</th>
<th>PCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCR</td>
<td>MCR</td>
<td>MCR</td>
</tr>
<tr>
<td>ICR</td>
<td>MCR</td>
<td>MCR</td>
</tr>
<tr>
<td>Nrm</td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>AIR</td>
<td>3.39 Mbps</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Switch</th>
<th>Cell switching delay</th>
<th>( 4 \mu s )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \alpha )</td>
<td>Load/ utilization measurement interval</td>
<td>0.125</td>
</tr>
<tr>
<td>Queue threshold for ER adjustment</td>
<td>500 ( \mu s )</td>
<td></td>
</tr>
<tr>
<td>Output buffer size</td>
<td>50 cells</td>
<td></td>
</tr>
<tr>
<td>Output buffer size</td>
<td>2000 cells</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 8(a) shows the ACR at source for flows \( s_1, s_2, \) and \( s_3 \), respectively with the MCR/PCR requirements for each flow listed in Table 1. The cell rates shown in the plot are normalized with respect to the link capacity (150 Mbps) for easy comparison with those values obtained with our centralized algorithm under unit link capacity (Table 1). After the initial transient period, we see that the cell rate of each flow matches its respective rate in Table 1. To study the network utilization of our distributed algorithm, we also show the inter-switch link utilization (Link 12) and the queue size of congested switch (SW1) in Fig. 8(b). We find that the link is 100% utilized with reasonably small buffer requirements.

Similarly, Fig. 9(a) shows the ACR at source for each flow with the MCR/PCR requirements listed in Table 3 and Fig. 9(b) shows the link utilization and queue size of Link 12. Again, we find that the cell rates under our distributed algorithm match with Table 3 with 100% link utilization and small buffer requirements.

#### B. The Three-Node Network

For this configuration (Fig. 4), the output port links of SW1 and SW2 are the GMM-bottleneck links.

Fig. 10(a) shows the normalized cell rate of each flow under our distributed algorithm. Comparing with the rates obtained by our centralized algorithm in Table 5, we find that after the initial

---

**Fig. 8.** (a) The cell rates of all flows; (b) the link utilization and queue size of the congested switch for the first example of the peer-to-peer network.
The cell rates of all flows; (b) link utilization and queue size of the congested switch for the second example of the peer-to-peer network.

The cell rates of all flows; (b) the link utilization and queue size at the output ports of congested switches for the three-node network.

<table>
<thead>
<tr>
<th>Flow</th>
<th>MCR</th>
<th>PCR</th>
<th>GMM Rate Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1</td>
<td>0.05</td>
<td>0.50</td>
<td>0.225</td>
</tr>
<tr>
<td>s2</td>
<td>0.05</td>
<td>0.15</td>
<td>0.150</td>
</tr>
<tr>
<td>s3</td>
<td>0.10</td>
<td>0.50</td>
<td>0.225</td>
</tr>
<tr>
<td>s4</td>
<td>0.40</td>
<td>0.50</td>
<td>0.400</td>
</tr>
</tbody>
</table>

The GMM-bottleneck link rate at Link 34 is 0.225. Fig. 12(a) shows the normalized cell rates of each flow under our distributed algorithm. We see that they match quite well with the rates listed in Table 8 after initial transient period. Fig. 12(b) shows the link utilization and buffer occupancy of the congested link (Link 34). Again, the GMM-bottleneck link is 100% utilized with small buffer requirements.

In summary, based on the simulation results in this section, we have demonstrated that our distributed algorithm achieves the GMM rate allocation policy in a regional enterprise network environment.

For a wide area network, the effectiveness of our simple distributed algorithm depends on careful system parameter tuning to minimize oscillations. Here, a more sophisticated distributed algorithm using per-flow state information [7] might provide better performance. But in a regional enterprise network environment, where implementation cost is critical, our simple algorithm is a viable solution (O(1) storage requirements and computational complexity).

V. CONCLUSIONS

The main contributions of this work are the generalization of the theory of the classical max-min to include the minimum rate and peak rate constraints for each flow, and the development of a simple distributed algorithm to achieve the generalized max-min rate allocation policy. Simulation results based on several benchmark network configurations demonstrated the effective-
ness of our distributed algorithm.

APPENDIX

A proof of Theorem 2 is given as follows.

Proof: The existence part is proved by the construction of a rate vector through a centralized rate allocation algorithm (e.g., Algorithm 1) and show that the vector satisfies the max-min policy. Therefore, we only need to prove the uniqueness part here. Suppose that the rate vector \( r \) is max-min and we will show that there cannot exist another rate vector which is also max-min.

Assume that there is some other feasible rate vector \( \hat{r} \neq r \) which is also max-min. Then there must exist a flow \( s \in S \) such that \( \hat{r}^s > r^s \) (if \( \hat{r}^s \leq r^s \) for every \( s \in S \), then some flow in \( \hat{r} \) does not have a bottleneck link and the rate vector \( \hat{r} \) cannot be max-min). For the rate vector \( r \), at the bottleneck link \( \ell \) for flow \( s \), we have \( F_\ell = C_\ell \) and \( r^s \geq \hat{r}^s \) for all \( t \) traversing \( \ell \). Since \( \hat{r}^s > r^s \), for feasibility we must have some flow \( t \) with \( \hat{r}^t < r^t \) at the bottleneck link \( \ell \) for flow \( s \). Thus the quantity

\[
\delta = \min\left\{ (\hat{r}^s - r^s), \ (r^t - \hat{r}^t) \right\}
\]

is positive. Therefore, we can increase \( r^t \) by \( \delta \) while decreasing the same amount of rate from flow \( s \) with \( \hat{r}^s \) on the link \( \ell \) traversed by \( s \) and \( t \) with \( F_\ell = C_\ell \). We maintain feasibility without decreasing the rate of any flow \( p \) with \( \hat{r}^p \geq r^p \) and this contradicts the max-min definition of the rate vector \( \hat{r} \).

The following is a proof for Theorem 3.

Proof: To show the "only if" part, we suppose that the ABR-feasible rate vector \( r \) is GMM and assume that on the contrary that there exists some flow \( s \in S \) which has neither an GMM-bottleneck link with respect to \( r \) nor a rate assignment equal to its PCR. Then for every non-saturated link \( \ell \) (\( F_\ell < C_\ell \)) traversed by \( s \), we can increase \( r^s \) by an increment until it reaches the PCR of \( s \) or some link saturates, whichever is smaller. For every saturated link \( \ell \) (\( F_\ell = C_\ell \)) traversed by \( s \), if \( T = \{ t \ | \ r^t > MCR^t, \ t \text{ traversing } \ell \} \) is nonempty, there must exist a flow \( p \in T, p \neq s \), such that \( r^p > r^s \). Thus the quantity

\[
\delta_\ell = \min\left\{ (C_\ell - F_\ell), \ (PCR^s - r^s) \right\} \quad \text{if } F_\ell < C_\ell,
\]

\[
\min\left\{ (r^p - r^s), \ (PCR^s - r^s) \right\} \quad \text{if } F_\ell = C_\ell
\]

is positive. Now let \( \delta \) be the minimum of \( \delta_\ell \) over all links \( \ell \) traversed by \( s \). Therefore, we can increase \( r^s \) by \( \delta \) while decreasing the same amount of rate from flow \( r^p \) on the links \( \ell \) traversed by \( s \) with \( F_\ell = C_\ell \). We maintain ABR-feasibility without decreasing the rate of any flow \( t \) with \( r^t \leq r^s \). This contradicts the GMM definition of the rate vector \( r \).

For the proof of the "if" part of Theorem 3, we assume that each flow has either an GMM-bottleneck link with respect to the ABR-feasible rate vector \( r \) or a rate assignment equal to its PCR.

- **Case 1:** To increase the rate of any flow \( s \) with \( r^s < PCR^s \) while maintaining ABR-feasibility, we must decrease the rate of some flow \( p \) with \( r^p > MCR^p \) and \( p \) traverses the GMM-bottleneck link \( \ell \) of \( s \) (flow \( s \) must go through an GMM-bottleneck link since \( r^s < PCR^s \) and we have \( F_\ell = C_\ell \) by the definition of an GMM-bottleneck link). Since \( r^s \geq r^p \) for all \( p \) in \( T = \{ t \ | \ r^t > MCR^t, \ t \text{ traversing } \ell \} \) by the definition of GMM-bottleneck link, the rate assignment for any flow \( s \in S \) with \( r^s < PCR^s \) satisfies the definition for GMM rate allocation.

- **Case 2:** For any flow \( s \) with \( r^s = PCR^s \), we cannot further increase the rate of \( r^s \) while maintaining ABR-feasibility. That is, we cannot generate another ABR-feasible rate vector \( \hat{r} \) with \( \hat{r}^s > r^s \). Thus, the rate assignment for any flow \( s \) with \( r^s = PCR^s \) satisfies the requirement for GMM.
Combining Cases 1 and 2, we have proved the “if” part of the theorem. □

The correctness proof of Algorithm 3 is given as follows.

Proof: Since 1) At least one of the following three events happens during an iteration of Algorithm 3: i) The rate of flows in \( u_1 \) reaches the rate of flows in \( u_2 \), which is MCR; ii) Some link saturates, and iii) Some flow in \( u_1 \) reaches its PCR; and 2) the number of flows in the network \( N \) is a constant equal to \( |S| \), the algorithm terminates at most by \( 2(|S| - 1) \) iterations (at most \( |S| - 1 \) iterations for \( |S| - 1 \) flows to reach the maximum MCR value among all flows and another \( |S| \) iterations for each flow to reach its GMM-bottleneck link rate or its PCR).

The correctness of this algorithm is proved by showing that each flow will either have some GMM-bottleneck link with respect to the final rate vector \( r \) or a rate assignment equal to its PCR when the algorithm terminates. Initially, the rate allocation of each flow \( s \in S \) is \( r_s^{\ast}(0) = MCR^\delta \). During each iteration, an equal increment of rate is added to all flows in \( u_1 \) (each flow in \( u_1 \) has the same rate) and as a result one of the following three events listed above happens at the end of an iteration.

- Case 1: Suppose that the rate of flows in \( u_1 \) reaches the rate of flows in \( u_2 \) at the \( k \)th iteration. Then no flow is removed from \( S^{(k)} \) during this iteration. The flows in both \( u_1 \) and \( u_2 \) become the new \( u_1 \) at the \( (k + 1) \)th iteration. The new \( u_2 \) at the \( (k + 1) \)th iteration is \( u_3 \) from the \( k \)th iteration, etc. That is, the number of sorted sets \( m \) for the \( (k + 1) \)th iteration is 1 less than that in the \( k \)th iteration. Case 1 can happen at most \((|S| - 1)\) times in executing Algorithm 3 since the number of flows is a constant with \( |S| \).

- Case 2: Suppose that a link \( \ell \) is saturated at the \( k \)th iteration, and \( s \in S^{(k)} \) traverses \( \ell \). We have \( r_s^\ast \geq r^\ell \) for every \( t \) traversing \( \ell \) such that \( r^\ell > MCR^\delta \). That is, link \( \ell \) is a GMM-bottleneck link with respect to \( r \) for flow \( s \).

- Case 3: Suppose that flow \( s \) reaches its PCR at the \( k \)th iteration. Then \( r_s^\ast(k) = PCR^\delta \) and \( r_s^\ast(k) \) will not be increased further during future iterations.

As a result, upon termination of the algorithm, each flow either has some GMM-bottleneck link or a rate assignment equal to its PCR. By Theorem 3, the final rate vector \( r \) satisfies the GMM policy. □