BOTTLENECK BRANCH MARKING FOR NOISE CONSOLIDATION IN MULTICAST NETWORKS

Jordi Ros, Wei K. Tsai and Mahadeven Iyer
Department of Electrical and Computer Engineering
University of California, Irvine, CA 92697
{jros, wtsai, miyer}@ece.uci.edu

Abstract

The noisy feedback consolidation problem in point-to-multipoint ATM multicast networks is studied. A new algorithm, which keeps track of the M smallest available rates (AR) from the branches at each branching point, is proposed. This algorithm has zero response delay, noise stability (defined in this paper), and small probability of noise. The probability model assumes no knowledge of the distribution of the available rate from the branches. Both analytical and simulation results demonstrate the superiority of the new algorithm.

1 Introduction

Multipoint communication has rapidly become the mode of choice for sharing data delivery across a network. An important problem here is the noisy feedback consolidation problem in point-to-multipoint communication across connection oriented networks such as ATM or MPLS networks. In point to multipoint connections, a source has to transmit at the available rate (AR) of the most bottlenecked link in the whole multipoint tree. This means that the source has to collect the state of all the branches and feedback consolidation at the branching points becomes necessary.

In all signaling protocols, backward resource management (BRM) cells sent from leaf nodes periodically arriving at the branching point bring (feedback) the state of the branches in the form of available rates, stored in the *explicit rate* (ER) field in the BRM cell. BRM cells are periodically sent back to the root bringing the newest bottleneck rate to the source. The bottleneck rate is obtained by minimizing over all leafs the available rates. If a costly per-branch accounting is to be avoided, this minimization is carried out at a branching point by keeping track of only one variable called MER (minimum explicit rate). Since this minimization has to be carried out in a distributed asynchronous way, a signaling protocol has to decide when to re-start the minimization. The re-start can be done in two ways: one is to reset MER to a default large value from time to time, the other is to reset it after at least one feedback value has been received from each branch. In the first approach, the MER values sent back to the source might not be the true bottleneck rate in the multipoint tree, in which case a *noise* is said to be generated. In the second approach, one has to wait for a long time, especially when there are non-responsive leafs, defeating the very purpose of timely feedback. Such are the key issues in the commonly known *consolidation noise* problem [fah97].

This paper presents a study of the consolidation feedback issue in multipoint connections for ATM networks. We first present a review of the current solutions of the problem. Then, we introduce a new approach: the Bottleneck Branch Marking (BBM) algorithm. By introducing the idea of storing the ID of the branch, we prove that some of the weaknesses of the previous approaches can be overcome. The trade-off between performance and complexity is reflected in the parameter *M*, the number of branches the switch keeps track of. This technique happens to have two nice properties. First, we prove that even if the number of branches *N* tends to infinity, the probability of having noise does not tend to 1. In this paper, a "worst-case" probability model of the consolidation noise is constructed. Second, the noise probability decreases exponentially with the number of branches *M* the switch keeps track of. For example, we prove that for *M=20* (only the 20 most bottleneck branches are stored) and $N = \infty$ (there are infinite number of branches), the noise probability is about less than 10^{-8}. This result says that we don't need infinite storage to consolidate infinite

[1] This research is supported by the National Science Foundation, award ANI-9979469, under the CISE ANIR program.
This work is also supported in part by the University of California, Irvine, and the Generalitat de Catalunya through a Balsells Fellowship to Jordi Ros.

number of branch ER values but just a small storage.

Even though the BBM scheme has been presented to solve the particular noise consolidation problem for multicast ATM connections, it should be considered as a general tool that can solve any general consolidation feedback problem. In these problems, a branching point receives feedback from many different sources and has to make a decision about which feedback is passed back to the root. We prove that the BBM approach can dramatically reduce the complexity while still achieve excellent performance.

This paper is organized as follows. Section 2 introduces the feedback consolidation issues and section 3 reviews the existing consolidation algorithms. The BBM algorithm is introduced in section 4 with its performance analysis in sections 5 and 6. Simulations and concluding remarks follow in sections 7 and 8.

2 Feedback Consolidation Issues

The following are the most important issues that need to be addressed when implementing a consolidation algorithm.

Consolidation noise. The available rate received from all the branches has to be minimized and sent back to the source. This can be done by using a per-branch accounting of this available rate. However, because this may be too expensive, current algorithms use only one field that is updated every time a BRM cell is received and is reset to infinity every time a BRM cell is sent. Because of this simplification, some algorithms tend to send feedback to the source with values that do not correspond to the most bottlenecked branch but to some other branch. We call this effect *consolidation noise*.

Noise stability. We will show that some of the current algorithms tend to be unstable in terms of noise. In other words, noisy feedback (feedback which does not bring a correct value) tends to produce more noisy feedback.

Transient response and level sensitivity. In order to consolidate the bottleneck rate, some algorithms wait for the feedback from all the branches to be received before sending a BRM cell. This incurs in a higher *transient response delay*. Because this effect is produced at each branching point, the consolidation algorithm may turn out to suffer *level sensitivity*: the transient response is sensitive to the number of levels in the multipoint tree.

Ratio BRM/FRM. Because in multipoint connections FRM (forward RM cell) cells are duplicated, the number of BRM cells may increase with the number of leafs of the tree. We define the *ratio BRM/FRM* as the number of BRM cells received at the source for each cell sent by the same source. In order to avoid implosion of BRM cells at the source, the consolidation algorithm has to control the ratio BRM/FRM so that it converges to one.

Branch responsiveness. Some algorithms need to implement additional code in order to overcome the case in which one of the branches is not responsive. This additional coding incurs additional complexity in the complete algorithm.

Complexity. The complexity of the algorithm needs also to be studied. Based on the previous papers ([Ren98], [Fah98]), the following will be assumed in this paper to judge the algorithm complexity:

-Turning around the cells is more complex than passing through the cells.

-Per-branch accounting is an expensive solution and should be avoided.

3 Consolidation Algorithms

In this section, we present some of the more relevant consolidation algorithms currently defined. Before this, we define the concept of consolidation algorithm.

Definition 1. *Consolidation Algorithm*. We define a *consolidation algorithm* at a branching switch as an algorithm that fulfills the following actions:

1) Periodically receives the available bandwidth from each branch.
2) Processes the set of available bandwidth in each branch to obtain the most bottlenecked branch and stores it in a field called *Minimal Explicit Rate (MER).*
3) Periodically sends the value of *MER* to the root.

For the rest of this section, we show a quick review based on [Fah98], which presents 7 consolidation algorithms. Because of space limitation, here we only examine algorithms 1, 3, 4 and 6 (using the same notation in [Fah98]). For a further review, refer to [Fah98].

3.1 Algorithm 1

The main idea for this algorithm is that BRM cells are returned from the branching point when FRM cells are received, and contain the minimum of the values indicated by the BRM cells received from the branches after the last BRM cell was sent. FRM cells are duplicated and multicast to all branches at the branching point.

3.2 Algorithm 3

In this approach, the branching point does not turn around the FRM cells, but the BRM cell that is received from a branch immediately after an FRM cell has been received by the branching point is passed back to the source, carrying the minimum ER value since the last BRM cell was sent. Again, FRM cells are duplicated and multicast to all branches at the branching point.

3.3 Algorithm 4

The main idea in this algorithm is that a BRM cell is passed to the source only when BRM cells have been received from all branches after the last time a BRM was

sent. FRM cells are duplicated and multicast to all branches at the branching point.

3.4 Algorithm 6

This approach reflects the idea that there is no need to wait for feedback from all the branches when an overload situation has been detected. In this case, the overload is immediately indicated to the source by sending a BRM cell and, hence, the response time is reduced. Because this may cause extra BRM cells to be sent, the ratio BRM over FRM may turn out to be bigger than one. To avoid this problem, algorithm 6 keeps track of the number of extra cells sent to the source. When feedback from all leaves indicates under-load and the value of extra BRM cells is more than zero, this particular feedback is ignored.

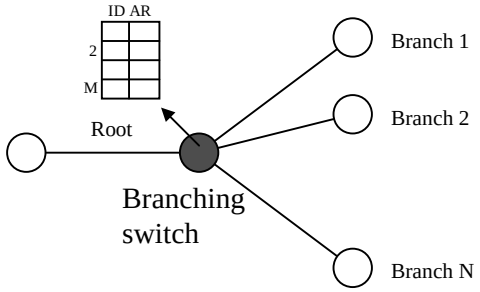

Figure 1. Branch switch model

4 The BBM Algorithm

There are two fundamental reasons for which the previous algorithms suffer from several consolidation issues

(1) The MER value has to be reset to infinity (or to the peak cell rate) every time a BRM cell is sent back to the root. Because of this, if a BRM cell is sent before all the feedback has been consolidated, noise may be generated.

(2) The only way to increase MER is to wait for it to be reset. In other words, suppose that the most bottlenecked branch increases its available rate and sends its feedback to the branching switch. Because there is no way for the switch to know that this particular feedback comes from the most bottlenecked branch, it cannot increase MER. Instead, it has to wait for a reset of MER and for a new minimization iteration to determine the new bottleneck available rate.

We suggest that both problems can be solved if the switch stores the identifier of the branch for which it has stored the available rate. Indeed, note that (2) can be solved since now the switch can identify that the most bottlenecked branch has increased its available rate. Moreover, this provides the means to increase MER without having to reset its value and, hence, (1) is also solved.

```
Upon the receipt of an FRM cell
   Multicast FRM cell to all branches;
Upon the receipt of a BRM cell
   NumberOfBRMsReceived ++;
   IF BRM.ER is one of the Mth most bottleneck
      Store BRM.ER and the branch ID in BBM;
   IF BRM.ER is the most bottleneck
      Send to the root this BRM cell;  / so that response time = 0 /
      IF NumberOfBRMsReceived == N
         NumberOfBRMsReceived ⟵ 0;
      ELSE
         ExtraBRM ++;
   ELSE
      IF NumberOfBRMsReceived ==N
         NumberOfBRMsReceived ⟵ 0;
         IF ExtraBRM == 0;
            BRM.ER ⟵ Smallest AR in BBM cell;
            Send to the root this BRM cell;
         ELSE
            ExtraBRM --;
            Discard this BRM cell;       / to ensure BRM/FRM ⟶ 1 /
      ELSE
         Discard this BRM cell;
```

Figure 2. BBM Algorithm

The previous reasoning is the motivation for the *Bottleneck Branch Marking* (BBM) algorithm. The BBM algorithm keeps track of the M most bottlenecked branches. For this, the switch stores a matrix with M entries including both the identifier of the branch (ID) and the last available rate received at the branching node, as shown in fig. 1. We call this matrix the *BBM matrix* and its size M can be adjusted depending on the scalability and performance requirements. For example, if N is the number of branches, then the case $M=N$ is equivalent to a per-branch approach.

Fig. 2 presents the pseudocode of the BBM algorithm. The BBM matrix has M entries, and each entry consist of both the ID and the available rate. A new feedback is stored in this structure only if its available rate is smaller than any of the available rates in the BBM matrix. There are two ways for which a branching switch may send a BRM cell back to the root:

(1) Whenever a new bottleneck appears: Note that this way the algorithm has zero response delay. However, that may cause the ratio BRM/FRM be bigger than one. To control this, the algorithm stores the number of extra BRM cells sent because of a new bottleneck in the *ExtraBRM* field.

(2) Whenever the number of feedback values received since the last time a BRM cell was sent is equal to the number of branches: This one is implemented with the *NumberOfBRMsReceived* field. However, if $ExtraBRM > 1$ then no BRM cell is sent. This way the

algorithm ensures convergence of the ratio BRM/FRM to 1.

5 Performance analysis

In this section, the issues defined in section 2 are examined one by one for each of the consolidation algorithms presented so far.

Noise Consolidation. Algorithms 1 and 3 suffer from the noise consolidation problem, since the MER is reset every time a BRM cell is sent and a BRM cell may be sent before the feedback from the most bottlenecked branch has arrived. Algorithms 4 and 6 do not suffer from this problem since a BRM cell waits for all the feedback before being sent.

We now show that the BBM scheme may also induce some erroneous feedback. Suppose *M=1* and that the two most bottlenecked branches have an available rate of AR_1 and AR_2, respectively, so that $MER = AR_1$. Assume now that a new feedback rate bigger than AR_2 arrives from the most bottlenecked branch. The new MER should be now AR_2. However, because the switch does not keep track of the second most bottlenecked rate, it cannot tell the correct MER. Note that this situation can be solved by making $M = 2$, but still we can find some other more complex situations in which this case also happens to fail.

The previous property proves that if *M* is smaller than the total number of branches, then some situations may induce noise. However, in the following sections we will prove that these situations happen to be very unlikely. Moreover, even if some noise is produced, we will prove that the BBM scheme is stable and that it quickly converges to the correct value without oscillations.

Noise Stability. It can be proven that if the network is in steady state (we assume a network is in steady state if no parameter changes in time) then algorithms 1 and 3 do not have noise. Intuitively, this is because in this particular case the frequency in which BRM cells arrive at the branching switch is equal to the frequency at which they are sent to the root. Then they can synchronize in such a way that by the time a BRM cell is sent, all the feedback has been consolidated. The BBM algorithm also does not suffer from noise in steady state. In this case, however, the reason is that as long as the available rates in the network do not change, the rates in the BBM matrix end up converging to the M most bottlenecked rates.

Suppose now that while being in steady state the source makes a mistake and transmits at a higher rate than the allowed for a short period of time. Then the inter-arrival time of FRM cells at the branching switch will decrease. In algorithms 1 and 3 this implies a decrease of the inter-departure time of a BRM cell at the same switch and, then, the chances of sending this BRM cell before all the feedback has been consolidated are bigger. This may produce a new noisy BRM cell that when arriving at the source will make the source transmit at a higher rate than the allowed. Now we are under the same condition than we were at the beginning and oscillation is produced.

This shows that algorithms 1 and 3 are unstable protocols. In other words, a single noisy BRM cell may induce the network to oscillate forever. Note that this is not the case of algorithms 4, 6 and the BBM algorithm. This is because for these cases, the inter-departure time of the BRM cells does not depend on the inter-arrival time of the FRM cells. In the particular case of the BBM algorithm, even though a casual noisy BRM cell may be generated (we proved this in the previous subsection), this one will not induce more noisy BRM cells and stability is immediately reached again.

Transient Response and Level Sensitivity. Algorithms 1, 3, 4, and 6 suffer transient delay since in any of them a BRM cell arrived at the switch with the most bottleneck value is not ensured to be sent immediately back to the root. Only algorithm 6 can send this BRM cell immediately in an overload situation. However, it cannot do the same for an under-load situation, this is when the most bottleneck branch increases its rate. For this case, in algorithm 6 the BRM cell will wait for all feedback to arrive incurring under-utilization of the network.

The BBM scheme achieves zero transient delay, in both overload and under-load situation. This is because it keeps track of the branch ID and hence can tell when the branch increases or decreases its available rate.

BRM/FRM. All presented algorithms solve the BRM/FRM issue so that it tends to one.

Responsiveness. Algorithms 4 and 6 wait for all the branches to respond before sending a BRM cell. Then, if any of the branches happens to become non-responsive, BRM cells will be never sent back. To solve this, the algorithm has to add more complexity so that this kind of situations can be detected.

Note that the rest of the algorithms do not suffer from non-responsive situations, since in these other approaches BRM cells never wait for all the feedback.

Complexity. Algorithm 1 turns around cells so it can be considered more complex, since most studies argue that turning around RM cells has a high implementation cost. The rest of the algorithms don't turn around cells and in this sense, they are simpler. The cost of BBM can be controlled by increasing or decreasing the number of entries (*M*) in the BBM matrix. Big values for *M* means high performance with more complexity, and vice versa.

One may wonder what is the minimal number of entries in the BBM matrix necessary to achieve a good enough performance for a fixed number of branches *N*. The following section provides the answer to this question. It will be shown that the performance increases exponentially as a function of *M*. Moreover, we will show

that even if the number of branches is very high (even infinity), small values of *M* can achieve very good performance results.

6 Noise Probability Analysis

In this section, we present a theoretical analysis of the noise probability for the BBM algorithm. This analysis turns out to be the key to understand the benefits of this approach. We will prove that a BBM scheme can dramatically reduce the complexity while achieving still good performance.

6.1 The Model

Fig. 2 showed the branch switch model used for the mathematical analysis of the noise probability. We focus on a single branching point connection with N branches and 1 root. The branching switch implements BBM and only keeps track of the M most bottlenecked branches, where $1 \le M \le N$. For the purpose of our mathematical analysis we will redefine *BBM* as an $N \times 2$ matrix such that for $i = 1,...,N$, $BBM(i,1)$ and $BBM(i,2)$ store the ID of the *i*th most bottlenecked branch and its last available bandwidth received, respectively. Note that even though we have now extended this matrix to include all *N* branches available in the switch, still the actual implementation of the BBM scheme consists of the sub-matrix including only the *M* most bottlenecked branches.

We define the time dimension in terms of iterations. An iteration is a period of time in which a complete set of consecutive feedbacks from all the branches has arrived. For the purpose of our mathematical analysis, we will assume that within an iteration, one and only one feedback from each branch arrives. Even though this may seem unrealistic, in section 7.3 we will prove through simulations that this assumption does not affect the results in this paper.

We use $n \in \mathrm{N}$ to denote a particular iteration in such a way that iteration *n+1* occurs just after iteration *n*. Using this notation, $f^n{}_i$ denotes the available bandwidth feedback received from branch *i* at iteration *n* and $BBM^n(i,j)$ is the final state of the BBM matrix at the end of iteration *n*. We also define the subsets of identifiers $BBM^n{}_h$ and $BBM^n{}_t$ as

$BBM_h^n = \{BBM^n(1,1), BBM^n(2,1),..., BBM^n(M,1)\}$

$BBM_t^n = \{BBM^n(M+1,1), BBM^n(M+2,1),..., BBM^n(N,1)\}$

, respectively, where the subindex *h* and *t* stand for the head and the tail of the BBM matrix.

We will assume that within an iteration, any order of arrivals of the feedback values is equally likely and that the available rates in the branches are independent and identically distributed (iid). Even though unrealistic, this model proves to be a worst-case assumption. Note that in a realistic scenario, branches that belong to a congested area in the network will likely remain congested, whereas branches that are not congested, e.g. because they have larger capacities, will likely remain not congested. This realistic scenario would make the BBM matrix more static reducing the probability of having noise.

6.2 Theory of the Noise Probability

Definition 2. *Most Bottlenecked Branch.* Let f_i be the last available rate feedback that has arrived from branch *i* at a branching switch with *N* branches. We define the *most bottlenecked branch* at this switch as

$$f^* = \min\{f_i,\ i = 1...N\}. \qquad (6.1)$$

Definition 3. *Noise Probability.* We define the *noise probability of a consolidation algorithm* as the probability that *MER* is not equal to f^*.

Note that the above definition does not depend on the source, which means that the noise probability is not necessary the same as the probability that the source receives a wrong feedback value. In fact, this last probability is lower than the noise probability, because a switch may have $MER \neq f^*$ but never send this value back to the root. Then, the noise probability in definition 3 is an upper bound of the probability that the source receives a wrong feedback value.

Property 1. *Consistency.* We say that the BBM matrix is consistent if $BBM^n(1,2) = f^*$. Then the following is true,

(1.a) If BBM^{n-1} is consistent and there exists at least one entry in the BBM matrix which has not received the feedback from its corresponding branch after iteration n-1, then the BBM matrix is still consistent.

(1.b) If a new feedback arrives while in iteration *n* with an available rate smaller than $BBM^{n-1}(M,2)$, then the BBM matrix is consistent for the rest of the *nth* iteration.

The proofs of the previous properties are straightforward and we omit them.

Lemma 1. *Set Partition.* Let $A^n{}_i$ be the set of events that satisfy the condition "*i* and only *i* feedback values from branches with identifier in $BBM^{n-1}{}_t$ arrive at iteration *n* earlier than one or more feedback values from branches with identifiers in $BBM^{n-1}{}_h$, with $i = 0,1,...,N-M$ ". Then the following is true,

1) $A^n{}_i$ is a partition. In other words, $\bigcup_{i=0}^{N-M} A^n{}_i = U$, where U is the set that includes all possible events at iteration *n*, and $A^n{}_i \cap A^n{}_j = \{\varnothing\}$, for $i \neq j$.

2) $\left|A^n{}_i\right| = \dfrac{M(N-M)!(M+i-1)!}{i!}$.

Proof. Let's begin proving (1). $\bigcup_{i=0}^{N-M} A^n{}_i = U$ is obvious if we consider that $i = 0,1,...,N-M$ covers all cases. $A^n{}_i \cap A^n{}_j = \{\varnothing\}$, for $i \neq j$, is also true because of the "i and only i feedback values …" statement in the definition of event $A^n{}_i$.

To prove (2), we consider a permutation in the set of feedback values $\{f^n{}_i, i = 1,...,N\}$ as a chronological arrival ordering of these feedback values. To count the number of elements in $A^n{}_i$, we now consider the steps to be done in order to build this set. At every step, we count the number of added events.

Step 1. Get i and only i feedback values from branches with identifier in $BBM^{n-1}{}_t$. These are a total number of events of,

$$\binom{N-M}{i} \qquad (6.2)$$

Step 2. Add the branch ID's of these feedback values to the set $BBM^{n-1}{}_h$ and remove them from the set $BBM^{n-1}{}_t$. Build all possible permutations considering this two-set partition of the whole set of branch ID's. This makes a total number of events of,

$$(M+i)!(N-M-i)! \qquad (6.3)$$

In addition, considering step 1 we have a total number of events of,

$$\binom{N-M}{i}(M+i)!(N-M-i)! \qquad (6.4)$$

Note that the previous expression includes all events in the set $\bigcup_{j=0}^{i} A^n{}_j$, since when considering all permutations we are also including those events in $A^n{}_j$ for $j = 1,...,i-1$. This fact is used in the last step.

Step 3. We subtract the overlapping events in the total number of events obtained in step 2. Mathematically,

$$A^n{}_i = \bigcup_{j=0}^{i} A^n{}_j - \bigcup_{j=0}^{i-1} A^n{}_j = \binom{N-M}{i}(M+i)!(N-M-i)! - \binom{N-M}{i-1}(M+i-1)!(N-M-i+1)! \qquad (6.5)$$

Finally, simplifying the previous expression we obtain the total number of events in $A^n{}_i$,

$$\left|A^n{}_i\right| = \frac{M(N-M)!(M+i-1)!}{i!} \qquad (6.6)$$ ■

Lemma 2. *Noise necessary condition*. Let j be the identifier of a branch such that either

1) j belongs to the set $BBM^{n-1}{}_h$, or
2) j belongs to the set $BBM^{n-1}{}_t$ and the feedback from j at iteration n arrives earlier than one or more feedback values from branches with identifiers in $BBM^{n-1}{}_h$.

If such a branch exists and the available rate brought by it is smaller than $BBM^{n-1}(M,2)$, then for the duration of the iteration n we have that $BBM^n(1,2) = f^*$. In other words, there is no noise at iteration n.

Proof. If a feedback from a branch j brings an available rate smaller than $BBM^{n-1}(M,2)$, then by second part of lemma 1 the BBM matrix will be consistent after the arrival of j's feedback and for the rest of the iteration. Now suppose that j satisfies (1). Then, by the first part of lemma 1 we have that the BBM matrix was consistent before the arrival of j's feedback. Suppose that j satisfies (2). Then by the time that j's feedback arrives, there exists at least one entry in the BBM matrix that has still not received feedback. Because of this, we can apply first part of lemma 1 and this implies that the BBM matrix was consistent before the arrival of j's feedback.

Now we have proved that for the entire iteration (after and before j's arrival), the BBM matrix is consistent and, hence, noise is not produced within this iteration. ■

Note that the previous lemma gives a sufficient (not necessary) condition of not having noise. This is equivalent to a necessary condition for having noise. We will use this lemma to provide and upper bound to the noise probability.

Lemma 3. Assume that the set of available rates arrived at the branching switch are independent and identically distributed. Then, the probability that a new available rate that has arrived from an arbitrary branch j is bigger than $BBM(M,2)$ is equal to,

$$P(AR_j \geq BBM(M,2)) = \frac{\text{N!H([-N+M-1, M],[1+M],1)}}{\text{(M-1)!(N-M)!M}} \qquad (6.7)$$

where the function $H(n,d,z)$ is the generalized hypergeometric function, also known as Barnes's extended hypergeometric function.

Proof. To simplify the notation, we denote $BBM(M,2)$ by $AR^*{}_M$. Note that AR_j refers to the available rate at an arbitrary branch with identifier j and that $AR^*{}_M$ refers to the *Mth* smallest available rate from all the branches. We begin by expressing $P(AR_j \geq AR^*{}_M)$ in terms of conditioned probability by using the continuous version of the Theorem on Total Probability [PAP79, page 177]. Therefore, we have that,

$$P(AR_j \geq AR^*_M) = \int_0^1 P(AR_j \geq AR^*_M / AR^*_M = x) f_{AR^*_M}(x)dx \quad (6.8)$$

where $f_{AR^*_M}$ is the probability density function (pdf) of AR^*_M. Note that we have normalized the available rate from a branch without a lost of generality so that the limits of integration are from 0 to 1.

In the previous expression, we have that $P(AR_j \geq AR^*_M / AR^*_M = x) = 1 - F_{AR_j}(x) = 1 - F_{AR}(x)$, where $F_{AR}(x)$ is the cumulative distribution function (cdf) of the available rate at any branch. We now concern with the term $f_{AR^*_M}$. This term is the marginal pdf of the *Mth* smallest value among a set of *N* independent and identically distributed random variables. To solve this, we apply *Order Statistics Theory* [Kar81]. Suppose $\overline{AR}^*_M$ is the *Mth* smallest value among a set of *N* independent and uniformly distributed random variables. Then [Kar81, pages 100-137] proves that the pdf of $\overline{AR}^*_M$ is the following:

$$f_{\overline{AR}^*_M}(x) = \begin{cases} \dfrac{N!x^{M-1}(1-x)^{N-M}}{(M-1)!(N-M)!}, & \text{for } 0 \leq x \leq 1 \\ 0, & \text{elsewhere} \end{cases} \quad (6.9)$$

In order to find an expression for any arbitrary distribution function (recall that (6.9) corresponds to the uniformly distributed case), we use the transformation method in [GAR94, page 155]. This method proves that a uniform distributed random variable is obtained when the cdf of a random variable is applied to this same random variable. Mathematically, we have that

$$\overline{AR}^*_M = F_{AR}(AR^*_M) \quad (6.10)$$

Now (6.9) shows the pdf for the uniform distribution case and (6.10) shows the relation between the general distribution, AR^*_M, and the uniform distribution, $\overline{AR}^*_M$. With both results and knowing that F_{AR} is monotonically increasing, we can use the Fundamental Theorem in [PAP79, page 126] to derive $f_{AR^*_M}$:

$$f_{AR^*_M}(x) = f_{\overline{AR}^*_M}(F_{AR}(x)) \cdot \frac{dF_{AR}(x)}{dx} \quad (6.11)$$

Substituting $f_{\overline{AR}^*_M}(x)$ we have,

$$f_{AR^*_M}(x) = \begin{cases} \dfrac{N![F_{AR}(x)]^{M-1}(1-[F_{AR}(x)])^{N-M}}{(M-1)!(N-M)!} \dfrac{dF_{AR}(x)}{dx}, & \text{for } 0 \leq x \leq 1 \\ 0, & \text{elsewhere} \end{cases}$$

Now we know both $P(AR_j \geq AR^*_M / AR^*_M = x)$ and $f_{AR^*_M}$. Substituting both in (6.8) and integrating, we obtain,

$$P(AR_j \geq AR^*_M) = \frac{\text{N!H([-N+M-1, M],[1+M],1)}}{\text{(M-1)!(N-M)!M}}$$

which proves the lemma. ■

The surprising result in lemma 3 is the fact that $P(AR_j \geq AR^*_M)$ does not depend on the distribution function of the available rate at a branch. Though it may seem controversial, there is an intuitive reasoning to understand this property, which is not presented in this paper because of space constraints. Thanks to this property, we will show that there exists an upper bound for the noise probability independently of the distribution function of the available rate.

Theorem 1. *Noise Probability.* The noise probability of the BBM algorithm can be upper bounded by,

$$P(noise) \leq \sum_{i=0}^{N-M} (\rho_{M,N})^{M+i} \frac{M(N-M)!(M+i-1)!}{i!N!} \quad (6.12)$$

where $\rho_{M,N}$ is equal to

$$\rho_{M,N} = \frac{\text{N!H([-N+M-1, M],[1+M],1)}}{\text{(M-1)!(N-M)!M}} \quad (6.13)$$

Proof. Using the Theorem on Total Probability we have,

$$P(noise) = \sum_{i=0}^{N-M} P(noise / A^n_i) P(A^n_i) \quad (6.14)$$

Note that we can use this theorem because A^n_i is a partition of U, as proved in first part of lemma 1. Let's first calculate $P(A^n_i)$. Recall that in our model we assume that within an iteration, any order of arrivals is equally likely. Then, all events in A^n_i are equally likely and we can write $P(A^n_i)$ as,

$$P(A^n_i) = \frac{\text{number of events in } A^n_i}{\text{number of events in } U} \quad (6.15)$$

Since the number of events in U corresponds to the total number of permutations, this is $N!$, using the second part of lemma 1 we have that,

$$P(A^n_i) = \frac{M(N-M)!(M+i-1)!}{i!N!} \quad (6.16)$$

We now provide an upper bound for $P(noise / A^n_i)$. From lemma 2 we can bound the probability of not having noise conditioned to A^n_i,

$$P(\overline{noise} / A^n_i) \geq P(\text{conditions in lemma 3 are met} / A^n_i)$$

Note that the above is true since lemma 2 is a sufficient but not necessary condition for not having noise. Now the probability that the conditions in this lemma are met is equal to one minus the probability that these conditions are not met. The later can be obtained by ensuring that for any branch satisfying conditions (1) or (2) in lemma 2, its available rate brought at iteration *n* is bigger than

$BBM^{n-1}(M,2)$. Under condition $A^n{}_i$, there are $M+i$ such branches. Assuming independent and equally distributed branches we have,

$$P(\overline{noise}/A^n{}_i) \geq 1-\left[P(AR_j \geq BBM^{n-1}(M,2))\right]^{M+i}$$

where AR_j is the available rate of an arbitrary branch *j*. Equivalently, we have,

$$P(noise/A^n{}_i) \leq \left[P(AR_j \geq BBM^{n-1}(M,2))\right]^{M+i}$$

Now, using lemma 3 we have,

$$P(noise/A^n{}_i) \leq \left[\frac{\text{N!H([-N+M-1, M],[1+M],1)}}{\text{(M-1)!(N-M)!M}}\right]^{M+i}$$

which completes the prove. ■

Theorem 2. *Noise Probability Bound.* The noise probability for the particular case $M=1$ (i.e. the branching switch only keeps track of the most bottleneck branch) and with infinite number of branches is at most $1-e^{-1}$. Mathematically,

$$\lim_{N\to\infty} P(noise)\Big|_{M=1} \leq 1-e^{-1} \qquad (6.17)$$

Proof. Using theorem 1 we have that

$$P(noise)\big|_{M=1} \leq \sum_{i=0}^{N-1}\left(\text{NH([-N, 1],[2],1)}\right)^{1+i}\frac{(N-1)!}{N!}$$

The limit as $N\to\infty$ can be solved by expanding the hypergeometric function and using *Stirling's Formula* [GAR94, page 45] to obtain

$$\lim_{N\to\infty} P(noise)\Big|_{M=1} \leq 1-e^{-1} \ ■$$

This last theorem may seem a surprising result at first sight. Ideally, to achieve a noise probability of zero we need to keep track of all the branches. If the number of branches is infinity, then we need to keep track of infinite number of branches. However, if we only keep track of one branch (the most bottleneck) the noise probability does not tend to one. Indeed, the theorem proves that it can be bounded by $1-e^{-1} \cong 0.6321$. Though surprising, this result is actually intuitive if we carefully consider lemma 2. In this case, note that when *N tend*s to infinity, the probability that the second condition in this lemma becomes true goes to one, since now $BBM^{n-1}{}_t$ has infinite elements and $BBM^{n-1}{}_h$ has *M* finite elements. This implies that the conditions in the lemma are now more likely and, because these are sufficient conditions to avoid noise, the chances to avoid noise are better.

Actually, it is possible to find a bound for any arbitrary value of *M*. Because Stirling's Formula turns out to be only applicable to the case $M=1$, so far no symbolic expression has been derived for cases $M>1$. For this cases, we present a numeric solution. Table 1 shows the results of this analysis.

Table 1. Bound Noise Probability for infinite number of branches

M	1	2	3	4	5	6
P(noise)	0.6321	0.2969	0.1281	0.0531	0.0214	0.0085

Fig. 3 shows the noise probability bound for a branching switch with up to 100 branches using a BBM matrix with up to five entries. Note that as *N* becomes bigger, the noise probability bound tends to the values in table 1.

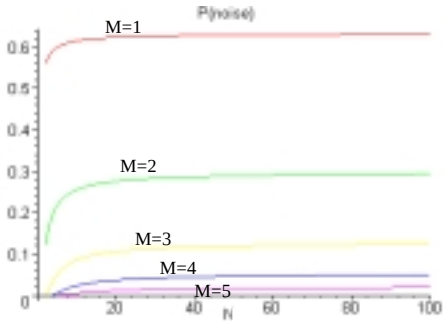

Figure 3. Noise Probability bound for M=1, 2, 3, 4 and 5.

In fig. 4 the noise probability bound has been numerically evaluated for the case 20,000 branches ($N = 20,000$) in order to approximate the case of infinite number of branches. Note that the *y*-axis is represented in a logarithmic scale. This means that the noise probability drops exponentially as the number of branches we keep track of (*M*) increases. In particular, the probability that noise occurs when we only keep track of 20 branches of a total number of 20,000 branches, is about less than 10^{-8}. Note that this dramatically reduces the implementation complexity, compared to a per-branch accounting approach, while still maintains high degree of performance.

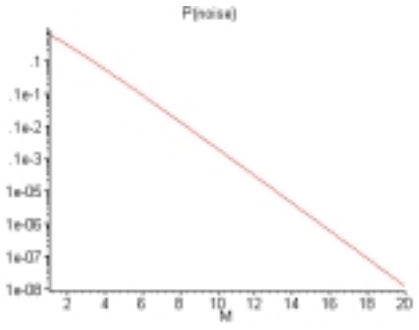

Figure 4. Noise probability bound for N=20,000 branches

This example shows the utility of the BBM scheme. Actually, this approach is totally general and can be seen as a tool to be used in any protocol when the per-branch accounting scheme is too expensive to be implemented.

7 Simulations

In this section we present some simulations to prove the theoretical results presented in this paper. The first simulation shows how the consolidation noise issue affects each of the examined algorithms. In the second simulation, the transient response in the branching switch is measured for each algorithm. Finally, the third part in this section proves the validity of the model assumption presented in section 6.1 for the case of real networks.

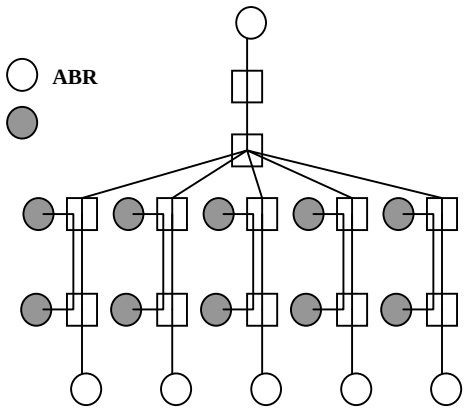

Figure 5. Network Configuration

7.1 Simulation I: Noise Consolidation

In this simulation we evaluate the effect of the noise consolidation issue. Fig. 5 shows the network configuration that has been used. It presents a multipoint switch with 5 branches. In order to simulate bottlenecks moving from branch to branch, VBR traffic is added at each branch. All link capacities are 155 Mbps. From left to right, the transmission rates for the VBR sources are [5, 35, 65, 90, 40] Mbps and their initial transmission times are [0, 3, 6, 9, 12] milliseconds, respectively. The results in terms of available cell rate (ACR) at the source for each algorithm are presented in fig. 6. Note that for algorithms 1 and 3 when a new bottleneck branch appears noise is produced, which makes the source oscillate among the available rates of different branches. Algorithms 4, 6 and the BBM algorithm, which has been simulated with *M=1*, do not suffer from noise.

7.2 Simulation II: Transient Response

This second simulation measures the transient response for each of the consolidating algorithms. Again, we use the network configuration in fig. 5. In order to simulate a dynamic situation in which the bottleneck link continuously changes from branch to branch, we configure the VBR sources to generate ON-OFF Poisson traffics with a mean burst length of 80 μsecs and a mean interval between burst of 2000 μsecs. The transmission rates in the ON interval are [40, 50, 60, 70, 80] Mbps, from the most left to the most right VBR source.

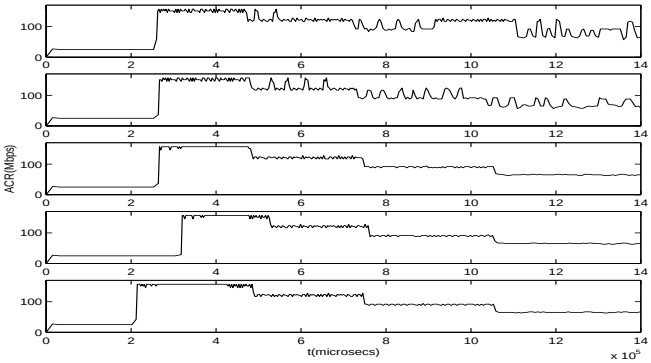

Figure 6. From up to down, ACR for algorithms1, 3, 4, 6, and BBM, respectively

Fig. 7 shows, for all the algorithms, the transient delay that a BRM cell with a new bottleneck rate suffers at the branching switch before it is passed back to the root. All cases have been simulated with the same seed in order to make a fair evaluation. While all the algorithms suffer transient delay, note that the BBM algorithm has a zero delay.

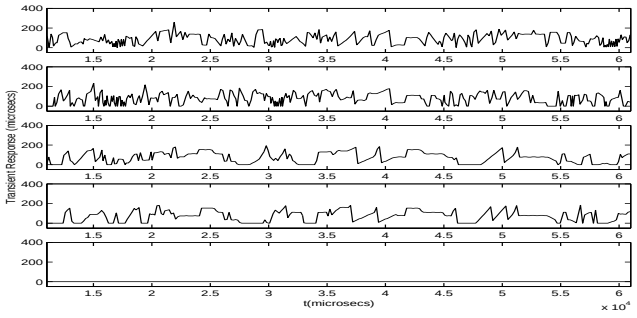

Figure 7. From up to down, transient delay for algorithms 1, 3, 4, 6 and BBM, respectively

7.3 Simulation III: Model Assumption

In this section we will provide some simulations to prove the validity of the mathematical model used to obtain the noise probability upper bound. In our model, we assumed that in one iteration (as defined in section

6.1) one and only one feedback arrives from each branch. We simulate three different traffic patterns:

- Traffic pattern 1: corresponds to the traffic pattern defined by our theoretical model, i.e. exactly one feedback from each branch arrives at each iteration. The distribution of the order in which each feedback arrives at each iteration corresponds to a uniform distribution.

- Traffic pattern 2: The interarrival time of the feedback in each branch corresponds to a uniform distribution.

- Traffic pattern 3: The interarrival time of the feedback in each branch corresponds to an exponential distribution.

Hence, for traffic pattern 2 and 3 no assumption regarding the number of feedbacks arriving from each branch in a iteration is done. Note also that we do not specify the time properties of the interarrival distributions (mean and variance) since the noise probability analysis is independent of them.

For all simulations, the value of the available bandwidth in the feedback cells is assumed to be uniformly distributed. There is no loss of generality since our upper bound does not depend on the type of distribution. The results below have been obtained by simulating 2000 iterations for each plotted point in the graphs.

Fig. 8a shows the results for traffic pattern 1. Comparing this graph to that of fig. 3, we show that expression *(6.12)* is an actual upper bound of the noise probability under our model assumption.

Fig. 8b and 8c show the same results applied to traffic patterns 2 and 3. This graphs show that the asymptotic behavior of the noise probability still applies even for the real case of any order of arrivals.

Finally, fig. 8d shows for all traffic patterns the scalability property of the BBM algorithm. In other words, the graph proves that the noise probability decreases exponentially with the number of branches we keep track of. This is consistent with the theoretical results presented in fig. 4.

8 Conclusions

Table 2 shows a comparison between the consolidation algorithms. The BBM scheme is the only approach that can achieve zero transient response delay. It is also stable in the sense that a noisy BRM cell does not generate more noisy BRM cells. It does not suffer from the non-responsive branches issue and ensures that the ratio BRM/FRM converges to 1. Finally, BBM defines a trade-off between the complexity and the noise consolidation issue that can be controlled with the parameter *M*. However, as we have proved, this trade-off turns out to be favorable, since very low noise can be achieved with small complexity (small *M)*.

References

[Fah97a] Sonia Fahmy, Raj Jain, Shivkumar Kalyanaraman, Rohit Goyal, Bobby Vandalore and Xiangrong Cai, "A Survey of Protocols and Open Issues in ATM Multipoint Communication," OSU Technical Report, August 21, 1997.

[Fah97b] Sonia Fahmy, Raj Jain, Rohit Goyal, Bobby Vandalore, "A switch algorithm for ABR multipoint-to-point connections," ATM Forum/97-1085, December 1997.

[Fah98] Sonia Fahmy, Raj Jain, Rohit Goyal, Bobby Vandalore, Shivkumar Kalyanaraman, Sastri Kota, and Pradeep Samudra, "Feedback Consolidation Algorithms for ABR Point-to-Multipoint Connections in ATM Networks," Infocom 98, San Francisco, March 1998, vol. 3 pp. 1004-1013 .

[Fah99] Sonia Fahmy, Raj Jain, Rohit Goyal and Bobby Vandalore, "Fairness for ABR Multipoint-to-point Connections," Submitted to IEEE Network Magazine, March 1999.

[Gar94] Alberto Leon-Garcia, "Probability and Random Processes for Electrical Engineering", 2nd Edition, Addison Wesley, Massachusetts, 1994.

[Kar81] Samuel Karlin, Howard M.Taylor, "A Second Course in Stochastic Processes", Academic Press, New York, 1981.

[Pap79] Athanasios Papoulis, "Probability, Random Variables, and Stochastic Processes", McGraw-Hill, Taiwan, 1979.

[Ren96] W. Ren, K-Y Siu, and H. Suzuki, "On the Performance of Congestion Control Algorithms for Multicast ABR Service in ATM," Proceedings of IEEE ATM'96 Workshop, San Francisco, August 1996.

[Ren97] W. Ren, K.-Y. Siu, and H. Suzuki, "Multipoint-to-Point ABR Service in ATM Networks," IEEE International Conference on Communications, Montreal, Canada, June 1997.

[Ren98] W. Ren, K-Y Siu, and H. Suzuki, and M. Shinohara, "Multipoint-to-multipoint ABR Service in ATM Networks," Computer Networks and ISDN Systems, October 1998, vol. 30, no.19, pp.1793-1810.

[Sim98] MANUAL: The NIST ATM/HFC Network Simulator, Operation and Programming Guide, Version 4.0, December 1998.

[Van99] Bobby Vandalore, Sonia Fahmy, Raj Jain, Rohit Goyal and Mukul Goyal, "QoS and Multipoint support for Multimedia Applications over ATM ABR service," IEEE Communications Magazine, January 1999, pp. 53-57.

Table 2. Comparison of consolidation algorithms

Algorithm	1	3	4	6	BBM
Consolidation noise	Yes	Yes	No	No	Exponentially decreases with M
Noise stability	Unstable	Unstable	Stable	Stable	Stable
Transient response delay	> 0	> 0	> 0	> 0	0
BRM/FRM	1	1	1	1	1
Responsiveness	OK	OK	KO	KO	OK
Complexity	High	Low	Low	Low	Scalable with M

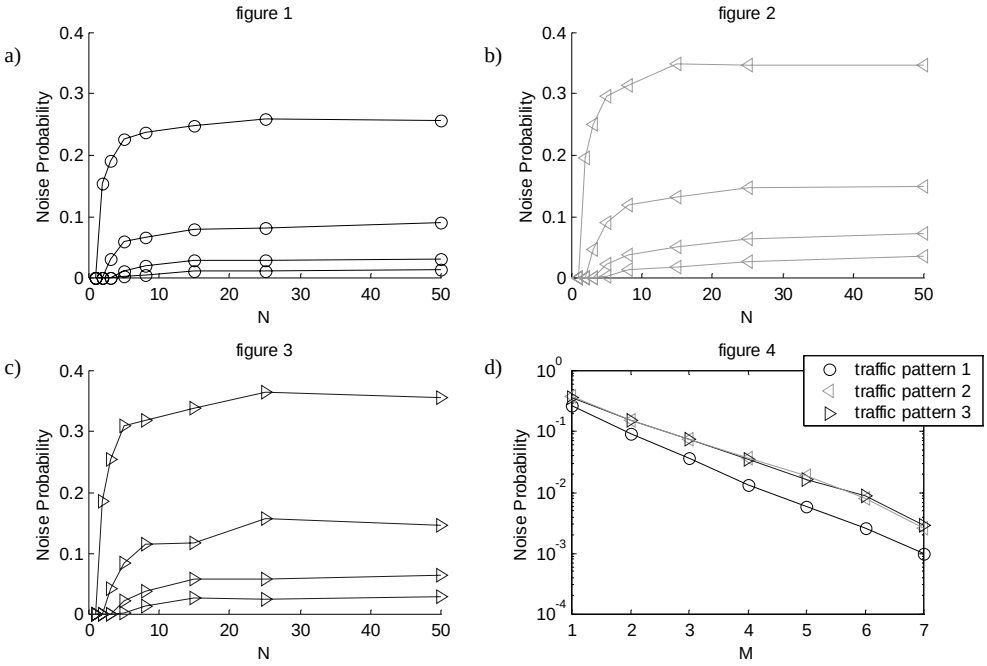

Figure 8. Noise probabilities graphs for traffic patterns 1, 2 and 3.