A Framework for Virtual Channel onto Virtual Path Multiplexing in ATM-ABR

Sonia Fahmy, Raj Jain, Bobby Vandalore, and Rohit Goyal
Department of Computer Sciences, Purdue University, West Lafayette, IN 47907-1398, USA

Abstract— **This paper proposes an algorithm for aggregating virtual channel connections (VCCs) onto virtual path connections (VPCs) in asynchronous transfer mode (ATM) networks. We focus on the interesting problem of multiplexing onto an available bit rate (ABR) VPC. ABR VPCs are particularly useful for connecting enterprise sites over the Internet, providing a virtual private network (VPN). The VPC/VCC hierarchy is also important for supporting Internet differentiated services over ATM. The coupling between the flow control mechanisms for VCCs and VPCs is not standardized. We propose fairness definitions for VPC bandwidth allocation, and describe an algorithm for allocating the VPC capacity to the multiplexed VCCs. Preliminary simulation results indicate that the algorithm achieves the required fair allocations, while controlling queue sizes.**

Keywords— **Asynchronous Transfer Mode (ATM), Available Bit Rate (ABR), multiplexing, flow control, congestion control, virtual paths**

I. Introduction

VIRTUAL private networks (VPNs) are rapidly gaining popularity. A VPN uses the public Internet to transparently connect private networks or even users, as if they are on the same network. VPNs are attractive because of their reduced costs (over leased lines), reduced administration overhead, and support for remote access and collaboration with partners. In [1], we have shown that ATM backbones can provide a good VPN service to enterprise sites. Aggregation of the site traffic onto one or two ATM virtual path connections (VPCs) is necessary for scalability, overhead reduction, fast re-routing and simplified billing.

ATM is proposed to transport a wide variety of services in a seamless manner. End systems must set up virtual channel connections (VCCs) of appropriate service categories prior to transmitting information. Service categories distinguish a small number of general ways to provide quality of service (QoS), which are appropriate for different classes of applications. A representative list of current and future applications includes video, voice, image and data in conversational, messaging, distribution and retrieval modes [2]. ATM networks currently provide five service categories [3]: constant bit rate (CBR), real-time variable bit rate (rt-VBR), non-real-time variable bit rate (nrt-VBR), unspecified bit rate (UBR), and available bit rate (ABR). The CBR and rt-VBR services are intended to transport real-time traffic, while the nrt-VBR, UBR and ABR services are designed for non-real-time traffic. In addition to these categories, the guaranteed frame rate (GFR) service has recently been standardized at the ATM forum traffic management working group [4]. The ITU-T I.371 also defines similar (but not the same) categories called ATM transfer capabilities.

One of the key distinguishing design aspects of ATM networks is the use of labels for switching. Use of labels speeds up the switching functions, and improves scalability since the labels need not be globally unique. This technique has now been adopted into the Internet in the form of multiprotocol label switching (MPLS). An interesting feature of label usage in ATM is the *aggregation* mechanism defined by the two level hierarchy of virtual path connections (VPCs) and virtual channel connections (VCCs). VPCs provide an elegant method for combining several VCCs between two end points. This technique is essential for scalability in backbone networks where there is a large number of flows. Using VPCs in the backbone reduces complexity and cost, and improves utilization. The mechanisms to perform traffic management for aggregate flows are currently being debated at forums like the differentiated services working group at the Internet Engineering Task Force (IETF). The scalability of the future Internet that combines real time and non real time traffic is affected by the outcome of this work.

This paper examines the traffic management issues in aggregating several VCCs onto a VPC, with a focus on the ABR service. We propose fairness definitions for VPC bandwidth allocation, and describe an algorithm for allocating the VPC capacity to the multiplexed VCCs. Preliminary simulation results indicate that the algorithm achieves the required fair allocations, while controlling queue sizes. The remainder of the paper is organized as follows. We give some background on the use of ABR VPCs, and on the ABR flow control mechanism. Then, we propose a framework for the coupling of the VPC and VCC ABR control loops, and use the ERICA+ algorithm as an example mechanism. Preliminary simulation results of the algorithm are then given.

II. Available Bit Rate VPCs in VPNs

Enterprise networks can be connected with an ATM backbone using VPCs, as shown in figure 1. Real-time and data traffic of the enterprise can be integrated on a single backbone VPC between sites. The advantages of separating edge device functionality from backbone functionality include simplification and scalability of the network design and bandwidth management, as well as scalability of the number of connections [5]. Enterprise *voice, video and data* integration within a *single* carrier VPC decreases the costs the enterprise pays (one VPC is used instead of two or more between any two points), and also allows dynamic sharing of voice, video and data bandwidth.

The network we propose is thus a two-tiered network, with an outer (access) tier and an inner (backbone) tier [1]. The access tier performs flow identification and QoS management at the flow level. Each switching node manages a relatively small number of flows. It may use ATM, frame relay, integrated services, or differentiated services for quality of service, or classes

The authors can be reached through e-mails: fahmy@cs.purdue.edu, raj@nayna.com, vandalor@cis.ohio-state.edu, rgoyal@nexabit.com

of service (COS). Traffic is aggregated at the edge into an ATM backbone (forming the inner tier). The backbone works with aggregate flows, mapped to ATM VPCs. The backbone traffic management is simple because of the large number of flows within each connection, and the high speed between the nodes. Backbone traffic management is at the granularity of aggregates, not for traffic within a flow.

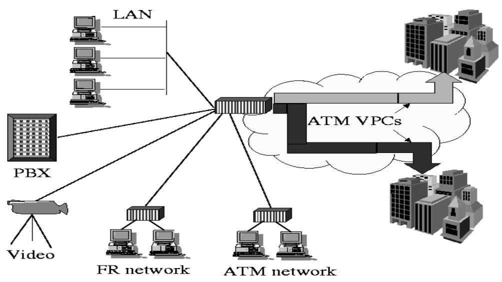


Fig. 1. The proposed architecture uses a single VPC to connect enterprise sites. Voice, video and data traffic can be multiplexed on this VPC.

The site implements the enterprise policy for managing the traffic. It performs flow identification and classification, QoS assignment, QoS management, and flow mapping within the local area network (the campus or the branch). QoS can be managed through: (1) tagging/marking, (2) dropping, or (3) assigning scheduling priorities. At the edge of the campus enterprise network, traffic is aggregated into the ATM VPCs for transport through the carrier network connecting the sites. The edge device uses a weighted fair queuing (WFQ) scheduler for scheduling traffic to the VPC(s), as shown in figure 2. The weights used by the WFQ scheduler for different traffic streams are assigned based upon: (1) the enterprise policy rules for users or applications, (2) the ATM parameters negotiated during connection admission, and the ATM service category, in case of ATM networks at an enterprise site, (3) the integrated services requests signaled by the application (if integrated services and the reservation protocol (RSVP) are used at the enterprise site), or (4) the service requested by the hosts and set in the packet headers using the differentiated services framework.

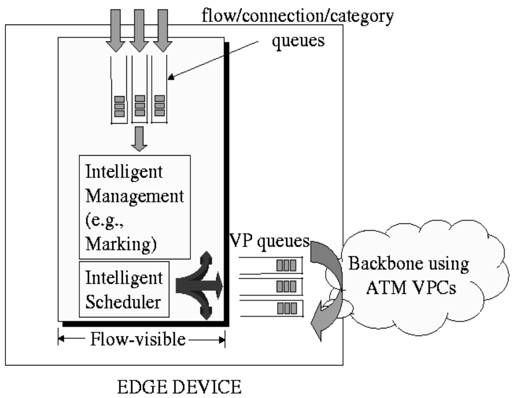


Fig. 2. The edge device performs traffic management based on the flows, and then intelligently schedules traffic to the backbone VPCs.

The choice of service category to use in the ATM backbone is critical to the quality of service experienced by applications sending traffic to another site of the enterprise. Each site is likely to have abundant bandwidth. Congestion most likely occurs on the relatively low-capacity WAN access link (for example, a Fast or Gigabit Ethernet feeding into a low capacity T1/E1 or T3/E3 link). Depending on the carrier ATM service category, congestion may occur in the carrier network leading to performance degradation.

A good ABR implementation performs well in the backbones connecting enterprise networks [1]. The ABR service pushes congestion to the edge devices, where adequate buffering can be provided, and, more importantly, the flows are visible and the enterprise policy can be applied. The ABR VPCs perform flow control for the pipes between enterprise networks. With ABR, there is very little loss in the backbone, and hence higher priority traffic can be transported without loss. On the other hand, the application takes advantage of all the bandwidth given by the network and efficiently utilizes the buffer at the edge device. This is not the case with other services, such as VBR, where either (1) the traffic is shaped according to the SCR to avoid loss in the network, which is clearly inefficient and increases delay, or (2) the traffic is shaped according to the PCR, which risks random losses inside the backbone, unless intelligent cell marking according to SCR is used.

III. ABR Flow Control

A good ABR implementation provides possibly non-zero minimum cell rate (MCR) guarantees, achieves fairness, and minimizes cell loss and queuing delay by periodically indicating to ABR source end systems the rates at which they should be transmitting. The switches monitor their load and compute the available bandwidth, dividing it fairly among active flows. The feedback from the switches to the sources is indicated in resource management (RM) cells which are generated by the sources and turned around by the destinations (refer to figure 3). RM cells flowing from the source to the destination are called forward RM (FRM) cells, while those flowing in the reverse direction are called backward RM (BRM) cells. The sources adjust their allowed cell rates (ACR) based upon the feedback received in BRMs. The ACR need not decrease beyond the guaranteed MCR, and cannot increase beyond the agreed upon peak cell rate (PCR).

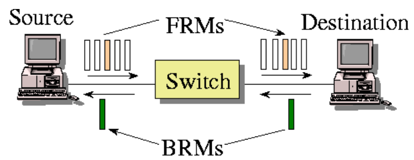


Fig. 3. Forward and backward RM cells flow in ABR connections

There are three ways for switches to give feedback to the sources. First, each cell header contains a bit called Explicit Forward Congestion Indication (EFCI), which can be set by a congested switch. Second, RM cells have two bits in their payload, called the Congestion Indication (CI) bit and the No Increase (NI) bit, that can be set by congested switches. Third, the RM cells have a field in their payload called explicit rate (ER) that can be reduced by congested switches to any desired value. The ACR cannot increase beyond the ER value. Such switches are called explicit rate switches [3]. We examine an explicit rate switch, the ERICA+ switch scheme, next.

A. Example Explicit Rate Algorithm: ERICA+

In ERICA+, the switch periodically monitors the load on each link and determines a load factor, z, the available capacity, and the number of currently active virtual connections. The load factor is calculated as the ratio of the measured input rate at the port to the target ABR capacity of the output link:
$z \leftarrow \frac{\text{ABR Input Rate}}{\text{Target ABR Capacity}}$ where:
Target ABR Capacity$\leftarrow Fraction \times$ Total ABR Capacity

Input rate is measured over a time interval called the switch averaging interval. The above steps are executed at the end of the switch averaging interval. $Fraction$ can be a constant set to a fraction (say 80%) of the available capacity, or it can be a function of the queuing delay at that port $f(Q_{port})$. We use an inverse hyperbolic function as explained in [6]. The optimal operating point is at an overload (z) value equal to one.

The fair share of each VC, $FairShare$, is also computed as follows:
FairShare$\leftarrow \frac{\text{Target ABR Capacity}}{\text{Number of Active ABR VCs}}$
If the source does not use all of its $FairShare$, then the switch fairly allocates the remaining capacity to the sources that can use it. Thus the switch scales the current cell rate (CCR) of the connection (as indicated in the RM cells) by the overload factor:
VCShare$\leftarrow \frac{CCR[VC]}{z}$

To achieve max-min fairness, ERICA+ maintains the highest allocation given to any VC on this output port during each averaging interval and ensures that all eligible VCs can also get this high allocation. The variable $MaxAllocPrevious$ stores the maximum allocation given in the previous interval, and $MaxAllocCurrent$ accumulates the maximum allocation given during the current switch averaging interval. For $z > 1 + \delta$, where δ is a small fraction (e.g., 0.1):
ER Calculated $\leftarrow$Max (FairShare, VCShare)
But for $z \leq 1 + \delta$, all the rate allocations are equal:
ER Calculated $\leftarrow$Max (FairShare, VCShare, MacAllocPrevious)

Thus, VCs are given equal allocations during underload and the (equal) CCRs are divided by the same z during the subsequent overload to bring the sources to their max-min fair shares. The system is considered to be in a state of overload when its load factor, z, is greater than $1 + \delta$. The aim of introducing the quantity δ is to force the allocation of equal rates when the overload is fluctuating around unity, thus avoiding unnecessary rate oscillations.

IV. Fair Multiplexing of ABR VCCs on ABR VPCs

The relationship between the service category of the VPC and the VCCs within it is implementation specific. In [5], the authors suggest using a rt-VBR VPC to aggregate CBR and rt-VBR VCCs, and using an ABR VPC to aggregate nrt-VBR, UBR and ABR VCCs. As ABR VPCs provide the more interesting case, we focus on ABR in the remainder of this paper. The ABR service can apply to both VPCs and VCCs. End points of ABR VPCs and those of ABR VCCs comply with the ABR source and destination behavior as given in the specifications [3]. The method used to divide the VPC bandwidth among the VCCs it contains is implementation specific. In the case when link capacity must be shared between both ABR VPCs and ABR VCCs, the method used to allocate the bandwidth is also implementation specific. In this section, we will focus on the fair allocation of bandwidth in these situations.

A. Weighted Max-Min Fairness

The optimal operation of a distributed shared resource is usually given by a criterion called the *max-min allocation* [7]. This fairness definition is the most commonly accepted one, though other definitions are also possible.

Definition: Given a configuration (sources, destinations, switches, links, connections) with n contending sources, suppose the i^{th} source is allocated a bandwidth x_i. The allocation vector $\{x_1, x_2, \ldots, x_n\}$ is *feasible* if all link load levels are less than or equal to 100%. □

Operational Definition: Max-min allocation: Given an allocation vector $\{x_1, x_2, \ldots, x_n\}$, the source that is getting the least allocation is, in some sense, the "unhappiest source". Find the feasible vectors that give the maximum allocation to this unhappiest source. Now remove this "unhappiest source" and reduce the problem to that of the remaining $n-1$ sources operating on a network with reduced link capacities. Again, find the unhappiest source among these $n-1$ sources, give that source the maximum allocation, and reduce the problem by one source. Repeat this process until all sources have been allocated the maximum that they can get. □

Intuitively, this means that all sources bottlenecked on the same link get equal rates, and if a source cannot utilize its fair share, the left over capacity is shared fairly among those who can use it. An extension of this definition guarantees a minimum cell rate (MCR) for each source, and shares the left-over capacity in a weighted manner. This is called the general weighted fair allocation [8].

Definition: General weighted fair allocation: Given a weight vector $\{w_1, w_2, \ldots, w_n\}$ that denotes the weight to be given to each source switched to a certain output port, and an MCR vector $\{MCR_1, MCR_2, \ldots, MCR_n\}$ denoting the minimum cell rate for each source switched to that port, the allocation for each source is denoted by:

$$x_i = MCR_i + \frac{w_i \times (\text{Capacity} - \sum_{i=1}^{n} MCR_i)}{\sum_{j=1}^{n} w_j}$$

□

We use general weighted fairness throughout the remainder of this paper.

B. Fairness for the VPC/VCC Hierarchy

Computation of the ideal allocations for the two level hierarchy (VCCs multiplexed on VPCs) is not straightforward. This is because scenarios are conceivable where a VPC with a larger number of VCCs multiplexed on it should be given more bandwidth than a VPC with a small number of VCCs. The question of how bandwidth is allocated among the VPCs (inter-VPC), and among the VCCs multiplexed on the same VPC (intra-VPC), becomes an important one. This is similar to the intra-group fairness and inter-group fairness for multicast groups discussed in [9], [10].

Example 1: Intra-VPC Fairness:
Consider the simple example in figure 4. A VPC has its own

flow control loop between the VPC end points (Switch 1 and Switch 3). Assume that the VPC MCR is zero. Suppose that three VCCs are multiplexed on this VPC: a VCC from user A to B, another from user C to D, and a third from user E to F. Assume the 3 VCC MCRs are zero. All available capacities on the links are 150 Mbps, except for the link from user A to Switch 1, which is only 10 Mbps. In this case, the flow control for the VPC will detect that 150 Mbps is available for the VPC, and will allocate it the entire available capacity. The VPC source end system (Switch 1) and the VPC destination end system (Switch 3) will cooperate with the network to regulate the VPC at this rate. The flow control for the VCCs within the VPC will divide the VPC capacity among the active VCCs multiplexed on the VPC. The connection from user A to B will be allocated its bottleneck rate of 10 Mbps. The available capacity of 150 Mbps − 10 Mbps = 140 Mbps will be equally divided upon the other two connections (C to D and E to F) and each will be allocated $\frac{140}{2} = 70$ Mbps. □

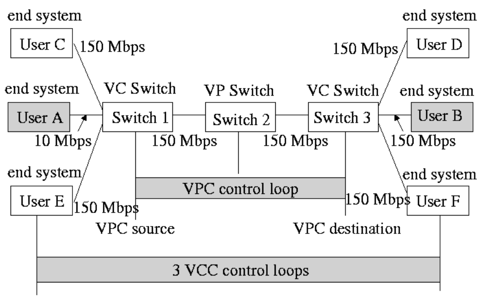


Fig. 4. Example 1: A single VPC and multiple VCCs

Example 2: Inter-VPC Fairness:
Now consider the example shown in figure 5. This is the same as the previous example, except that there is a second ABR VPC between Switch 1 and Switch 3. Suppose that the three VCCs (A to B, C to D, and E to F) are multiplexed on one of the VPCs, while there are 10 VCCs multiplexed on the second VPC (the 10 VCCs are assumed to be bottlenecked on the Switch 1 to Switch 3 path). The weights assigned to the two VPCs at a switch may be equal or different as follows.

Case 1: Equal Weights:

$$\forall i, j : i, j \in VPC : i \neq j : w_i = w_j$$

Assuming zero MCRs, each VPC is allocated $\frac{150}{2} = 75$ Mbps. The 75 Mbps is allocated to the 3 VCCs A to B, C to D, and E to F as follows. A to B is allocated 10 Mbps. The remaining bandwidth $75 - 10 = 65$ Mbps is divided equally among the 2 remaining connections so each is allocated $\frac{65}{2} = 32.5$ Mbps.

Case 2: Unequal Weights:

$$\forall i, j : i, j \in VPC : i \neq j : w_i \neq w_j$$

For example, suppose the VPC with 10 VCCs is assigned 5 times the bandwidth of the other VPC. In this case, the VPC with 10 VCCs gets $\frac{5}{6} \times 150 = 125$ Mbps, while the other VPC

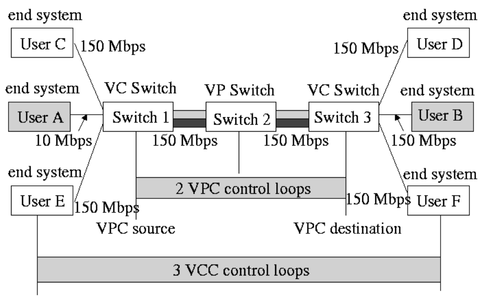


Fig. 5. Example 2: Two VPCs and multiple VCCs

is allocated 25 Mbps. The 25 Mbps is equally divided upon the three connections, such that each is allocated $\frac{25}{3} = 8.33$ Mbps.

Two interesting special cases arise:

Case 2.1: Weights according to Number of VCCs:
Suppose the VPC weights are assigned according to the number of connections multiplexed on them:

$$\forall i, j : i, j \in VPC : i \neq j : \frac{w_i}{NumOfVCC_i} = \frac{w_j}{NumOfVCC_j}$$

In this case, the weight for the VPC with 10 connections is $\frac{10}{3}$ times the weight of the VPC with 3 connections. The weights may need to be updated if new connections join the VPC.

For the above example, the VPC with 10 VCCs is allocated $\frac{10}{13} \times 150 = 115.38$ Mbps and each of the 10 VCCs is allocated $\frac{1}{10} \times 115.38 = 11.54$ Mbps. The VPC with 3 VCCs is allocated $\frac{3}{13} \times 150 = 34.62$ Mbps. User A is bottlenecked at 10 Mbps. Users C and E are allocated $\frac{34.62-10}{2} = 12.31$ Mbps each.

Case 2.2: Weights according to VCC ERs:
Suppose the VPC weights are assigned according to the explicit rates of connections multiplexed on them:

$$\forall i, j : i, j \in VPC : i \neq j : \frac{w_i}{\sum_i ER} = \frac{w_j}{\sum_j ER}$$

In this case, the available capacity on each link is divided fairly among the active connections, regardless of which VPC each connection belongs to. The ER (and hence ACR) for the VPC is simply the sum of the ERs for the VCCs it contains. This is a constantly varying quantity, and hence the weights must constantly be updated.

For the above example, user A is allocated 10 Mbps while all the other users are allocated $\frac{150-10}{12} = 11.67$ Mbps. □

From the above examples, it is clear that flow control for the ABR VPCs requires a weighted ABR flow control scheme, such as our weighted ABR with MCR scheme described in [8], in order to support giving different weights to different VPCs. Further modifications are necessary as explained next.

V. A Framework for Flow Control of ABR VCCs on an ABR VPC

ABR VCCs within a VPC share its capacity in the same way ABR connections share the capacity of a physical link. Figure 6 shows the use of ABR VPCs. A separate queue is used for each VPC at the VPC source to control its rate to the allowed cell rate (ACR), according to the feedback from the VPC BRM cells. Virtual source/virtual destination (VS/VD) can be used in the framework as discussed next.

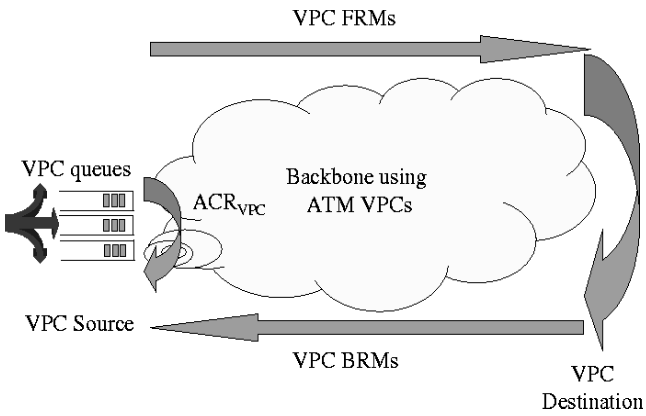


Fig. 6. ABR VPCs can be used in the network backbones to minimize delay and loss.

A. Using VS/VD

One option is to use a virtual destination (VD) for the VCC, and a virtual source (VS) for the VPC at the VPC *source*, and a virtual destination for the VPC, and a virtual source for the VCC at the VPC *destination*. This option is illustrated in figure 7.

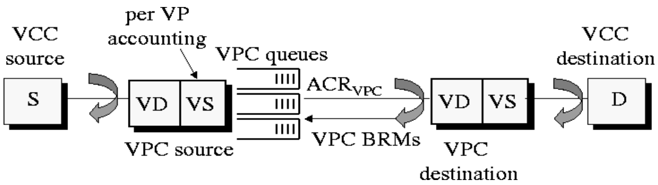


Fig. 7. Virtual source/virtual destination at the VPC end points

At the VS of the VPC, a separate VPC queue is used to control the VPC rate. The VDs of the corresponding VCCs in the same switch need: (1) per VP accounting information performed at the VPC VS, and (2) the ACR of the VPC, in order to compute the ER values for the VCC. Terminating/starting the VCC loop at the VPC end points is not required, but it eliminates the per-VC RM cell overhead and VCC RM cell processing inside the VPC loop. Separation of the flow control loops of the VCCs and the VPCs is also useful. VS/VD does incur additional overhead, however, since the end systems and switch functionality must all be provided at the VPC end points.

B. Without VS/VD

An alternative architecture without VS/VD is shown in figure 8. As in the VS/VD case, each VPC has a separate queue at the VPC source. Again, per VP accounting information and the VPC ACR are used to compute the rate indicated in the VCC RM cells at the VPC source. The two architectures and the rate computation operations are quite similar in both cases (with and without VS/VD). In the remainder of this section, we explain the operation of the VCC rate allocation algorithm in more detail.

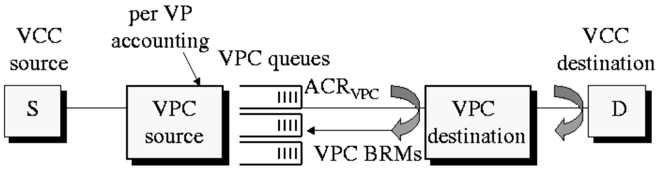


Fig. 8. VPC/VCC flow control coupling without VS/VD

C. Flow Control Framework

The framework has two main aspects: capacity estimation, and accounting, as discussed next.

C.1 Capacity Estimation

In most ABR rate allocation algorithms, the available capacity for ABR is estimated as:

Total ABR Capacity $\leftarrow$Link Capacity$-$CBR/VBR Capacity

This means that higher priority bandwidth is estimated by computing the sum of the number of CBR, rt-VBR and nrt-VBR cells scheduled during a certain interval of time. This sum is then subtracted from the link capacity, and a fraction of that is divided upon the VPCs (other than the CBR/VBR ones) according to the preassigned weights. This ABR capacity estimation operation must be performed by the *VPC* flow control mechanism if a VPC-VCC hierarchy exists. The total ABR capacity for multiplexed VCCs is simply the VPC allowed cell rate (ACR).

Once the total ABR capacity is estimated, the target ABR capacity is computed. For example, ERICA+ [11] (section III-A) computes the target ABR capacity as:

Target ABR Capacity $\leftarrow$*Fraction* $\times$ Total ABR Capacity

where the *Fraction* can be a constant, or a function of the queuing delay, $f(Q_{VPC})$, of the queue for this VPC at this port of the switch. *If the VPC contains VCCs of higher classes (e.g., CBR/VBR) their capacity must first be subtracted from the total ABR capacity.*

It is essential to take a *fraction* of the capacity allocated to the VPC. This is because we must allow the VPC queues to drain. These queues are caused by the delay between the instant when the ABR VPC allowed cell rate is controlled to the new value, and the instant the ACRs of all the multiplexed ABR VCCs are controlled. Since there are propagation and queuing delays between the VPC source end system, and the source end systems of the VCCs (refer to figure 8), the VPC queue can grow and must be controlled in the same way any ABR queue (whether a port queue, a VPC queue, or a VCC queue) must be controlled.

C.2 Accounting

In addition to the target ABR capacity, other estimates are required to be able to divide the capacity fairly among the active virtual connections. Examples of such metrics used in the ERICA+ scheme (section III-A) are: (1) the ABR input rate, (2) the number of active ABR connections, and (3) the maximum allocation given to any ABR VCC during the previous and current intervals.

In case of a VPC/VCC hierarchy, such computations and estimates must be separately performed for the VCCs on each VPC, and the VCCs on other VPCs should not interfere with this. In other words, estimating the input rate becomes estimating the input rate of the VCCs *on this VPC*, estimating the number of active connections becomes estimating the number of active connections *on this VPC*, and keeping track of the maximum allocation given during a certain interval only considers the allocations given to VCCs *on this VPC*.

C.3 Framework Model and Summary

We use the following notation:

$\lambda_{port,VPC}$	input rate of queue for VPC at *port*
$\mu_{port,VPC}$	service rate of queue for VPC at *port*
$Q_{port,VPC}$	queue length of queue for VPC at *port*
ER_{VCC}	explicit rate indicated to the VCC source by the VPC end point
ER_{VPC}	explicit rate indicated to the VPC source
ACR_{VPC}	allowed cell rate computed by the VPC source
$N_{port,VPC}$	number of VCCs multiplexed on VPC at *port*

We need to compute ER_{VCC} such that:

$$\lambda_{port,VPC} \leq f(Q_{port,VPC}) \times \mu_{port,VPC}$$

Or:

$$\sum_{VCC=1}^{VCC=N_{port,VPC}} ER_{VCC} \leq f(Q_{port,VPC}) \times ACR_{VPC}$$

This is performed as follows. Assume that the VPC flow control mechanism assigns an explicit rate value, ER_{VPC} to the VPC (this mechanism must handle the estimation of VBR and CBR bandwidth of other VPCs/VCCs, and the target ABR capacity). The VPC source sets the allowed cell rate of the VPC, ACR_{VPC} to the minimum of ER_{VPC} and $RIF_{VPC} \times PCR_{VPC}$, assuming the CI and NI bits are zero (or decreases the rate by RDF_{VPC} if CI is set) according to the source end system rules in [3].

As the VPC source rate must be controlled to ACR_{VPC}, per VP queues are required. The value of ACR_{VPC} must be communicated to the rate allocation algorithm for the VCCs at the VPC end point. The rate allocation algorithm will use this value as the estimated capacity and take a fraction of that (minus the CBR/VBR VCCs on the same VPC) as the target capacity. In addition, the algorithm must perform its accounting, e.g., the accounting of the input rate, active connections and maximum allocation, separately for the VCCs of each VPC.

VI. VPC/VCC ERICA+

We apply the general framework proposed above to the ERICA+ algorithm as described in section III-A. The only modifications required for ERICA+ at the VPC source end system are as follows: (1) The allowed cell rate of each VPC is controlled to ACR_{VPC}; (2) The Target ABR Capacity for the VCCs multiplexed on the VPC is computed as a *fraction* of the ACR of the VPC, ACR_{VPC} (minus the capacity of any CBR/VBR VCCs on this VPC). The fraction may depend on the queuing delay of the VPC queue $f(Q_{VPC})$ (or the VCC queues for the VCCs multiplexed on this VPC); and (3) The ABR Input Rate, Number of Active ABR VCCs, MaxAllocPrevious, and MaxAllocCurrent variables only apply for this VPC. Therefore, per-VP accounting must be performed at each output port.

The pseudocode of the algorithm is given next. A brief description of ERICA+ operation was given in section III-A (Refer to [11] for a more complete description of the ERICA+ algorithm and its performance.):

Initialization:

MaxAllocPrevious$_{VPC}$ $\leftarrow$ FairShare$_{VPC}$
MaxAllocCurrent$_{VPC}$ $\leftarrow$ FairShare$_{VPC}$

End of averaging interval:

Target ABR Capacity$_{VPC}$ $\leftarrow$ $Fraction_{VPC}$ $\times$ Allowed Cell Rate$_{VPC}$ $- CBR/VBR_{VPC}$

z_{VPC} $\leftarrow$ $\dfrac{\text{ABR Input Rate}_{VPC}}{\text{Target ABR Capacity}_{VPC}}$

FairShare$_{VPC}$ $\leftarrow$ $\dfrac{\text{Target ABR Capacity}_{VPC}}{\text{Number of Active VCs}_{VPC}}$

MaxAllocPrevious$_{VPC}$ $\leftarrow$ MaxAllocCurrent$_{VPC}$

MaxAllocCurrent$_{VPC}$ $\leftarrow$ FairShare$_{VPC}$

When an FRM is received:

CCR[VC] $\leftarrow$ CCR_in_RM_Cell

When a BRM is received:

VCShare $\leftarrow$ $\frac{CCR[VC]}{z_{VPC}}$

IF ($z_{VPC} > 1 + \delta$)
THEN ER $\leftarrow$ Max (FairShare$_{VPC}$, VCShare)
ELSE ER $\leftarrow$ Max (MaxAllocPrevious$_{VPC}$, FairShare$_{VPC}$, VCShare)

MaxAllocCurrent$_{VPC}$ $\leftarrow$ Max (MaxAllocCurrent$_{VPC}$, ER)

IF (ER $>$ FairShare$_{VPC}$ AND CCR[VC] $<$ FairShare$_{VPC}$)
THEN ER $\leftarrow$ FairShare$_{VPC}$

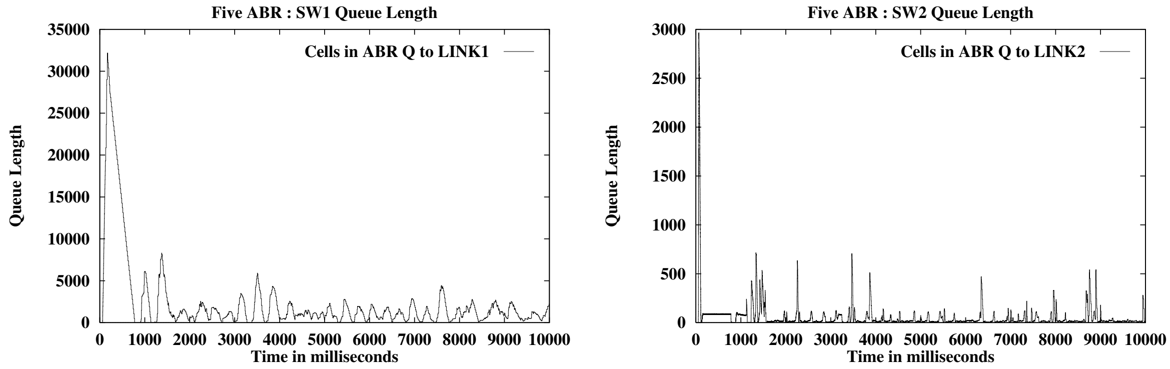
ER_in_RM_Cell $\leftarrow$ Min (ER_in_RM_Cell, ER, Target ABR Capacity$_{VPC}$)

VII. Simulation Results

Figure 9 shows the configuration used in our preliminary simulations. We have also used multiple bottleneck scenarios which are not shown here. The configuration consists of three switches separated by 1000 km links. The one way delay between the switches is 5 ms. Five sources send data as shown in the figure. The first hop from the sources to switch 1 is a long delay satellite hop. We simulated a one way delay of 50 ms (LEO satellite delay). The link capacity of link 2 is 45 Mbps, while all other links are 155 Mbps links. Our simulations use persistent ABR sources. ABR initial cell rates are set to 30 Mbps in all experiments. Link 2 is the bottleneck link for all connections.

The simulations demonstrate the basic idea of the algorithm, although they do not show the exact implementation discussed above. (The implementation used is a VS/VD scheme similar to the one explained in section V-A.)Thus the control loops for VCCs are terminated/started at the switches. (This is the reason for the presence of 4 and not 3 control loops.) All sources are multiplexed on a single VPC which is allocated a fraction of the link capacity. The resulting VPC ACR becomes the total capacity for all VCCs on this VPC.

Figure 10 shows the queue length results. The queue accumulation during the initial open loop period (before the feedback mechanism is in effect) is moved from switch 2 to switch 1 by the VS/VD mechanism. Thus, there are very small queues at



(a) Switch 1 Queue

(b) Switch 2 Queue

Fig. 10. Switch Queue Lengths for a 5-source LEO configuration

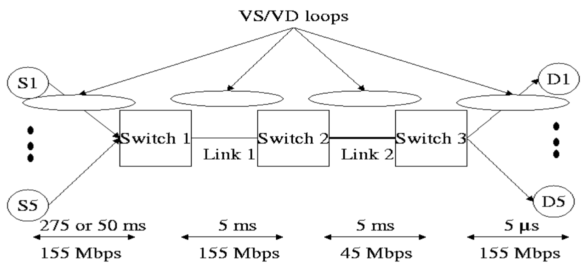


Fig. 9. Five source satellite configuration

switch 2. Pushing the queues to the edge is an important component of the architecture discussed in section II. Moreover, in case of satellite switches as in figure 9, it is important to minimize queue length in terrestrial switches (switch 2) which may not have sufficient buffering for an entire satellite round trip. The satellite switch (Switch 1) usually has larger buffers [12]. The 5 sources should each be allocated $\frac{45}{5} = 9$ Mbps. The ACR graphs (not shown here due to space constraints) illustrate that the scheme is fair in the steady state.

VIII. Summary and Conclusions

This paper has examined the flow control of the ABR virtual path/virtual channel hierarchy. The flow control at the VPC level needs to estimate the bandwidth available for ABR (accounting for CBR/VBR bandwidth), and assign the appropriate weights for different ABR VPCs. We have discussed the issues involved in the VPC/VCC coupling, and have given an example framework. The key aspect of this coupling is the use of the allowed cell rate value for the VPC source as the total capacity available for the VCCs multiplexed on this VPC. This capacity is scaled using the queuing delay of the VPC queue (or the appropriate VCC queues if per-VC queuing is used). In addition, all accounting performed at the output port is performed separately for each VPC. Other VCCs, and VCCs multiplexed on other VPCs, should not interfere with the flow control of VCCs multiplexed on a VPC.

This framework can be used for connecting enterprise sites on the Internet as a VPN. A single ABR VPC is used to connect two sites, and appropriate scheduling weights and drop policies are employed at the edge devices, as discussed in section II. This architecture can also be used for supporting differentiated services over ATM through a hierarchical scalable mechanism.

References

[1] S. Fahmy, R. Jain, S. Rabie, R. Goyal, and B. Vandalore, "Quality of service for internet traffic over ATM service categories," *Computer Communications*, vol. 22, no. 14, pp. 1307–1320, September 1999, http://www.cis.ohio-state.edu/˜jain/papers/enterprs.htm.

[2] Mark W. Garrett, "Service architecture for ATM: from applications to scheduling," *IEEE Network*, vol. 10, no. 3, pp. 6–14, May/June 1996.

[3] The ATM Forum, "The ATM forum traffic management specification version 4.0," ftp://ftp.atmforum.com/pub/approved-specs/af-tm-0056.000.ps, April 1996.

[4] John B. Kenney (Editor), "Traffic management working group baseline text," ATM Forum/BTD-TM-02.02, July 1999.

[5] Steve Rosenberg, Mustapha Aissaoui, Keith Galway, and Natalie Giroux, "Functionality at the edge: Designing scalable multiservice ATM networks," *IEEE Communications Magazine*, vol. 36, no. 5, pp. 88–99, May 1998.

[6] B. Vandalore, R. Jain, R. Goyal, and S. Fahmy, "Dynamic queue control functions for ATM ABR switch schemes: Design and analysis," *Journal of Computer Networks*, 1999, http://www.cis.ohio-state.edu/˜jain/papers/cnis_qctrl.htm.

[7] J. Jaffe, "Bottleneck flow control," *IEEE Transactions on Communications*, vol. COM-29, no. 7, pp. 954–962, July 1981.

[8] Bobby Vandalore, Sonia Fahmy, Raj Jain, Rohit Goyal, and Mukul Goyal, "Definition of general weighted fairness and its support in explicit rate switch algorithms," in *Proceedings of the IEEE International Conference on Network Protocols (ICNP)*, October 1998.

[9] S. Fahmy, R. Jain, R. Goyal, and B. Vandalore, "Fairness for ABR multipoint-to-point connections," in *Proceedings of SPIE '98 Conference on Performance and Control of network systems II*, November 1998, vol. 3530, pp. 131–142.

[10] S. Fahmy, R. Jain, R. Goyal, B. Vandalore, S. Kalyanaraman, S. Kota, and P. Samudra, "Feedback consolidation algorithms for ABR point-to-multipoint connections," in *Proceedings of the IEEE INFOCOM '98*, March 1998, vol. 3, pp. 1004–1013, http://www.cis.ohio-state.edu/˜jain/papers/cnsldt.htm.

[11] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and B. Vandalore, "The ERICA Switch Algorithm for ABR Traffic Management in ATM Networks," *IEEE/ACM Transactions on Networking*, vol. 8, no. 1, pp. 87–98, February 2000, http://www.cis.ohio-state.edu/˜jain/papers/erica.htm.

[12] R. Goyal, X. Cai, R. Jain, S. Fahmy, and B. Vandalore, "Per-VC rate allocation techniques for ATM-ABR virtual source virtual destination networks," in *Proceedings of the IEEE GLOBECOM*, November 1998.

[13] J. B. Kenney, "Solution for problem of resetting EFCI at ABR VP source," ATM Forum/97-0184, February 1997.