RENEGOTIABLE VBR SERVICE

THÈSE N^{O} 1948 (1999)

PRÉSENTÉE À LA SECTION DE SYSTÈMES DE COMMUNICATION

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES TECHNIQUES

PAR

Silvia GIORDANO CREMONESE

Laurea in Scienze dell'Informazione, Università degli Studi di Pisa, Italie
originaire de Schaffhouse (SH)

acceptée sur proposition du jury:

Prof. J.-Y. Le Boudec, directeur de thèse
Prof. C. Petitpierre, rapporteur
Dr M. Potts, rapporteur
Dr F. Scaroni, rapporteur

Lausanne, EPFL
1999

Renegotiable VBR Service

A Piergi e Filippo.

Ama e ridi se amor risponde,
piangi forte se non ti sente
Dai diamanti non nasce niente,
dal letame nascono i fior.
Fabrizio De André

Acknowledgements

I am very grateful to Prof. Jean-Yves Le Boudec who is an excellent PhD director and who spent much time guiding me and contributing to this work. I would like to thank all my colleagues (past and present) and professors at the ICA institute at EPFL for many stimulating discussions: Werner Almesberger, Ljubiza Blasevic, Catherine Boutremans, Olivier Crochat, Leena Chandran-Wadia, Falk Dietrich, Bruno Dufresne, Jean-Pierre Dupertuis, Hans Einsiedler, Eric Gauthier, Constant Gbaguidi, Guy Genilloud, Maher Hamdi, Jean-Pierre Hubaux, Paul Hurley, Shawn Koppenhoefer, Thorsten Kurz, Francis Lapique, Xavier Logean, Monika Lundell, Sam Manthorpe, Jean-Philippe Martin-Flatin, Carmen Mas, Andrey Naumenko, Txomin Nieva, Raffaele Noro, Philippe Oechslin, Gil Regev, Albin Schorer, Patrick Thiran, Olivier Verscheure, Alain Wegmann, Arnis Ziedins. I also owe special thanks to Holly Cogliati not only for her editing support and to Danielle Alvarez and Yvette Dubuis, secretaries of ICA, for many relaxing discussions.

I spent a particularly enjoyable three years working in the ACTS projects. I would like to acknowledge the help of many members of ACTS community and projects, in particular Samuli Aalto, Andrea Costoloni, Paulo De Sousa, Hannu Flinck, Martin Lorang, Sean Murphy, Martin Potts, Sathya Rao, Peter Sorensen, Rolf Schmidt and all the ASPA, ASCOM and TELSCOM teams. I also learned a lot by working in the ACTS programme and am grateful to all the members of the various projects for stimulating technical discussions in various cities around Europe.

I thank all my friends (especially G.E. and L.K.) for being always with me, even during my blue periods.

Finally I want to thank my husband, Piergi, for his encouragement and endurance during this work and for being the perfect mate for me. And my wonderful family and especially my parents, who never force me to do anything but always encourage me in whatever I do.

Abstract

In this work we address the problem of supporting the QoS requirements for applications while efficiently allocating the network resources. We analyse this problem at the source node where the traffic profile is negotiated with the network and the traffic is shaped according to the contract.

We advocate VBR renegotiation as an efficient mechanism to accommodate traffic fluctuations over the burst time-scale. This is in line with the Integrated Service of the IETF with the Resource reSerVation Protocol (RSVP), where the negotiated contract may be modified periodically.

In this thesis, we analyse the fundamental elements needed for solving the VBR renegotiation.

A source periodically estimates the needs based on: (1) its future traffic, (2) cost objective, (3) information from the past. The issues of this estimation are twofold:

1. future traffic prediction

2. given a prediction, the optimal change.

In the case of a CBR specification the optimisation problem is trivial. But with a VBR specification this problem is complex because of the multidimensionality of the VBR traffic descriptor and the non zero condition of the system at the times where the parameter set is changed. We, therefore, focus on the problem of finding the optimal change for sources with pre-recorded or classified traffic. The prediction of the future traffic is out of the scope of this thesis.

Traditional existing models are not suitable for modelling this dynamic situation because they do not take into account the non-zero conditions at the transient moments.

To address the shortfalls of the traditional approaches, a new class of shapers, the *time varying leaky bucket shaper* class, has been introduced and characterised by network calculus. To our knowledge, this is the first model that takes into account non-zero conditions at the transient time. This innovative result forms the basis of *Renegotiable VBR Service (RVBR).*

The application of our RVBR mathematical model to the initial problem of supporting the applications' QoS requirements while efficiently allocating the network resources results in simple, efficient algorithms.

Through simulation, we first compare RVBR service versus VBR service and versus renegotiable CBR service. We show that RVBR service provides significant advantages in terms of resource costs and resource utilisation. Then, we illustrate that when the service assumes zero conditions at the transient time, the source could potentially experience losses in the case of policing because of the mismatch between the assumed bucket and buffer level and the policed bucket and buffer level.

As an example of RVBR service usage, we describe the simulation of RVBR service in a scenario where a sender transmits a MPEG2 video over a network using RSVP reservation protocol with Controlled-Load service. We also describe the implementation design of a Video on Demand application, which is the first example of an RVBR-enabled application.

The simulation and experimentation results lead us to believe that RVBR service provides an adequate service (in terms of QoS guaranteed and of efficient resource allocation) to sources with pre-recorded or classified traffic.

Sommario

In questa tesi affrontiamo il problema del supporto della qualità del servizio richiesta dalle applicazioni a fronte di un'efficiente allocazione delle risorse di rete. Questo probleme viene analizzato al nodo sorgente, dove il profilo del traffico (traffic profile) viene negoziato con la rete ed il traffico viene reso conforme al contratto.

Sosteniamo che la rinegoziazione VBR è un meccanismo efficiente per la gestione delle fluttuazioni del traffico a livello burst. Questa visione è in linea con le specifiche di Integrated Service di IETF con Resource reSerVation Protocol (RSVP), in cui il contratto negoziato può essere periodicamente modificato.

In questa tesi analizziamo gli elementi fondamentali necessari alla risoluzione della rinegoziazione VBR.

Un nodo sorgente stima le risorse di cui ha bisogno basandosi su: (1) il proprio traffico futuro, (2) i costi, (3) le informazioni sul passato. Questa stima presenta due problemi:

1. la predizione del traffico futuro

2. data una predizione, trovare il nuovo valore ottimo da negoziare.

Il problema dell'ottimizzazione è banale nel caso di specifica CBR, ma risulta complesso nel caso di specifica VBR a causa della multidimensionalità del descrittore del traffico VBR e delle condizioni iniziali non nulle al momento di una transizione. Per questo motivo ci concentriamo sul problema di trovare il nuovo profilo ottimo di traffico per sorgenti che lavorano con traffico pre-recorded o classificato. La predizione del traffico futuro non fa parte degli obiettivi di questa tesi.

I modelli tradizionali non sono adatti alla modellizzazione di questa situazione dinamica, perché non considerano il fatto che, al momento della transizione, le condizioni del sistema possono essere diverse da zero.

Per colmare la lacuna degli approcci tradizionali, introduciamo una nuova classe di shapers che caratteriziamo con network calculus: la classe dei *time varying leaky bucket shapers*. Per quanto ci è dato sapere, questo è il primo modello che tiene conto di condizioni iniziali non nulle al momento di una transizione. Questo risultato innovativo è alla base del *Renegotiable VBR Service (RVBR)*.

L'applicare il modello matematico del servizio RVBR al problema del supporto della qualità del servizio richiesta dalle applicazioni nel rispetto dell'allocazione efficiente le risorse di rete porta, come risultato, ad algoritmi semplici ed efficienti.

Per mezzo di simulazioni, paragoniamo innanzitutto il servizio RVBR con il servizio VBR e con il servizio renegotiable CBR, dimostrando che il servizio RVBR fornisce vantaggi significativi in termini di costi e di utilizzazione delle risorse. In seguito, facciamo vedere che, qualora un servizio si basi sulla supposizione che le condizioni del sistema siano uguali a zero al momento della transizione, il nodo sorgente può subire delle perdite di traffico a causa del policing, in quanto, in questo caso,il nodo sorgente presuppone livelli di buffer e buckets disponibili che non collimano con i livelli che viengono controllati.

Come esempio di utilizzo del servizio RVBR, presentiamo la simulazione del servizio RVBR in uno scenario dove un sorgente trasmette un flusso MPEG2 su di una rete che utilizza il protocollo di riservazione RSVP con il servizio Controlled-Load. In seguito descriviamo il primo esempio di applicazione di Video on Demand, che supporta il servizio RVBR.

I risultati della simulazione e della sperimentazione ci portano ad asserire che il servizio RVBR offre un servizio adeguato (in termini di qualità del servizio e di allocazione efficiente delle risorse) a sorgenti che lavorano con traffico pre-recorded o classificato.

Contents

Chapter 1

Introduction

Integrated services networks introduce support to current and future applications that make use of different technologies as voice, data, and video. These multimedia applications require, in many cases, better service than a best effort service. This service is generally expressed in terms of Quality of Service (QoS), whereas network efficiency depends crucially on the degree of resources sharing inside the network.

To achieve both the applications' QoS requirements and network resources efficiency is extremely important for several reasons, for instance, network dimensioning or traffic charging.

We analyse how to achieve these goals at the source node where a traffic profile is negotiated with the network and the traffic is shaped according to the contract. The reference configuration is shown in Figure 1.1. A shaper, fed with a bursty traffic described by $R(t)$[1], shapes the traffic to respect the traffic profile established with the network, using a buffer of size X. A shaper is a system that stores incoming bits in a buffer and delivers them as early as possible while forcing the output $R^*(t)$ to be constrained with a given curve.

In order to allocate resources to satisfy the applications' QoS requirements, the sources use a resource reservation mechanism. Networks as ATM [1] or IP with RSVP [2] offer a limited set of way for describing the reservation, namely:

- Constant Bit Rate (CBR), primary specified by a peak rate p

[1] $R(t)$ represents the number of bits arrived at time t

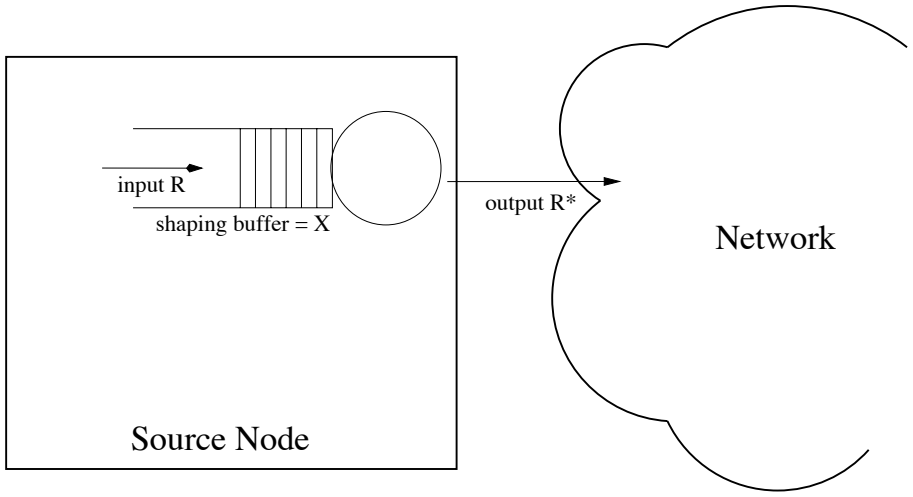


Figure 1.1: Shaping at a source node: using the shaping buffer of size X, the input traffic R is shaped in order to respect the traffic profile established with the network. The output of the shaping is indicated by R^*.

- Variable Bit Rate (VBR), mainly characterised by a peak rate p and a sustainable rate r and a burst size b.

- Renegotiated CBR (RCBR)[3] or VBR (RVBR)[4], where the traffic parameters of already active connections can be modified

A simple example is shown in Figure 1.2. A source S, as described in Figure 1.1, generates some bursty traffic represented by the curve R. We illustrate the output for CBR, VBR, RCBR and RVBR services.

When we attempt to satisfy the QoS requirements of S with a CBR service, the output is limited by the peak p. This implies that, in order to guarantee the QoS, S must request a very large p. This results, for bursty traffic, in a very unsatisfactory network utilisation: as illustrated in Figure 1.2(a), at time v_0, the source could have sent up to $p \cdot v_0$, but it has sent only $R^*(v_0) = R(v_0)$. The difference represents the unutilised resources.

VBR service allows the burst to go through for a limited period. The situation of unutilised resources is less frequent. However, even if VBR is more sophisticated, it is still unable to adapt to many traffic changes and, for long periods, the resources can be used inefficiently.

The situation improves with RCBR. S renegotiates the peak p to match its traffic changes and in the second interval it reduces the peak from p_1 to p_2. However, the choice is still limited to a single parameter.

The performance of the RVBR service is the best: S can renegotiate the set of parameters to adapt it to its traffic and the multidimensional parameters set assures flexibility on the interval time-scale. The gap between the resources requested and the resources used is substantially reduced.

Therefore, we advocate VBR renegotiation as an efficient mechanism to accommodate traffic fluctuations over the burst time-scale. This is in line with the Integrated Service of the IETF with the Resource reSerVation Protocol (RSVP), where the negotiated contract may be modified periodically [5].

In this thesis, we examine the system components that we need to put in place in order to solve the VBR renegotiation.

A source periodically estimates the needs based on: (1) its future traffic, (2) cost objective, (3) information from the past. The issues of this estimation are twofold:

1. future traffic prediction

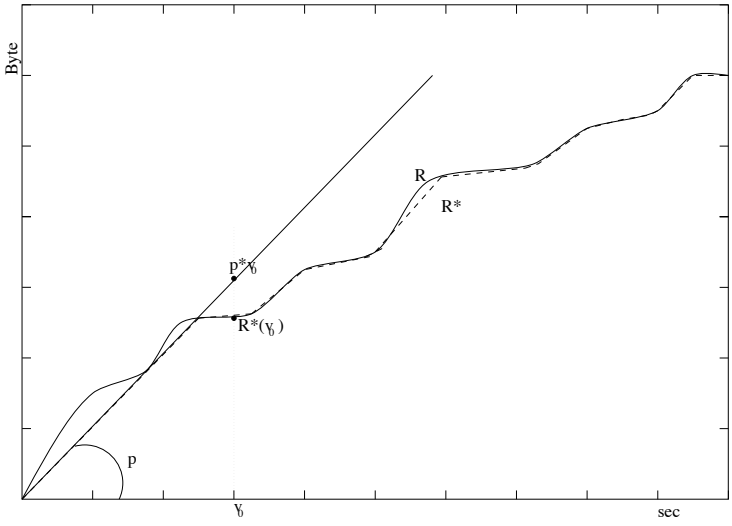
2. given a prediction, the optimal change.

If we use a CBR specification, the second issue is straightforward. But with a VBR specification the problem of finding the optimal change is complex because:

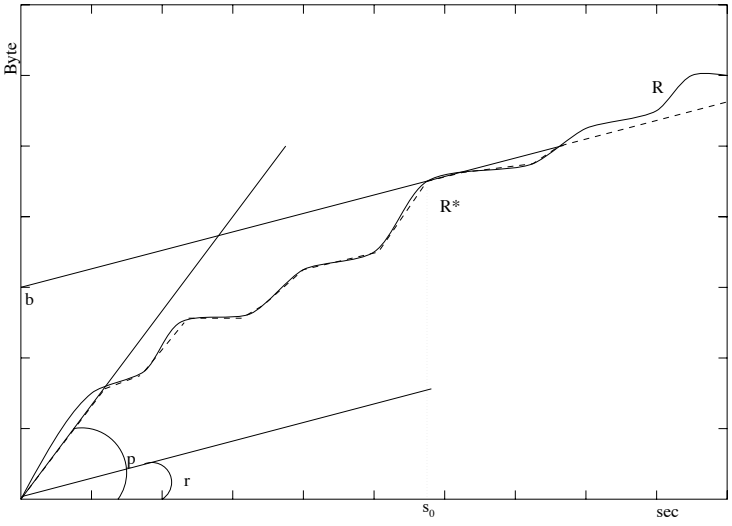
(a) there are several parameters (multidimensional traffic descriptor) and the optimal tradeoff is not obvious. We call this problem "the static VBR problem".

(b) at the transition times (where the parameter set is changed) the initial conditions in the leaky buckets and in the shaping buffer are different from zero. We call this problem "the dynamic VBR problem".

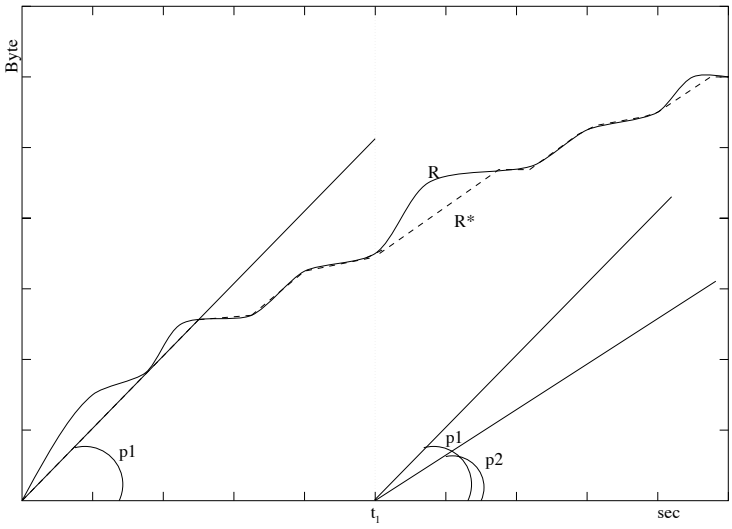
We, therefore, focus on the problem of finding the optimal change for sources with pre-recorded or classified traffic. The prediction of the future traffic is out of the scope of this thesis; see for example [6].



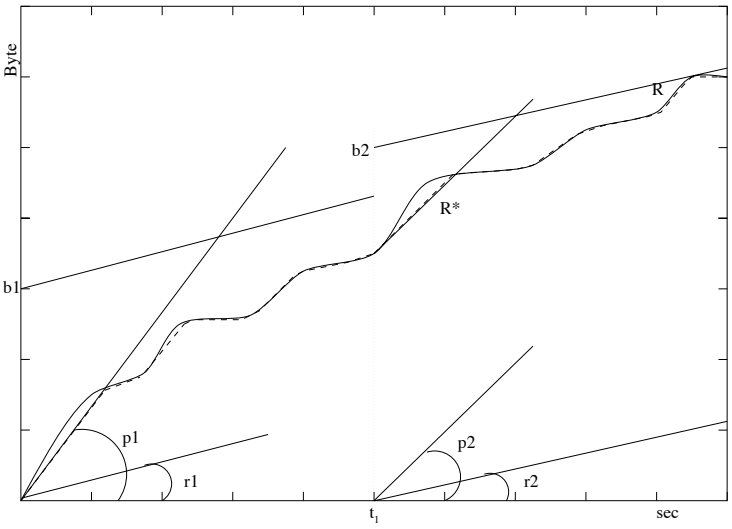
(a) CBR service: R^* is constrained by the peak p. At v_0 $p \cdot v_0 - R^*(v_0)$ represents the unutilised resources.



(b) VBR service: R^* is limited, in average, by the sustainable r. At time s_0 this limit is reached and S is forced to send at r.



(c) RCBR service: in the second interval S can reduce the peak from p_1 to p_2.



(d) RVBR service: in the second interval S can reduce both the peak from p_1 to p_2 and the sustainable from r_1 to r_2.

Figure 1.2: Services example: the four examples show the same input and resulting output

Example: the static and the dynamic VBR problems

As a simple example, let's consider the traffic generated by the source S. To ensure the QoS, as shown in Figure 1.2(d), the source attempts to shape the traffic conforming to renegotiable VBR traffic specifications. The shaping is done assuming a shaping buffer of capacity X.

The static VBR problem In the first interval ($[0, t_1]$) we encounter the problem of optimising the network resources needed for supporting the traffic generated by S. With a CBR service this problem is trivial: we can compute the optimal peak p as the deterministic equivalent capacity e_X [7], which takes into account the shaping done by the buffer X. However, as we have seen, the simple CBR specification leads to situations where the network resources are highly underutilised.

A VBR service allows the reduction of this effect with a multidimensional specification (p, r, b). The optimal peak rate p is still computed as the deterministic equivalent capacity e_X (see Section 3.3.3 and [7]), whereas there is a tradeoff to be made between the parameters of the leaky bucket (r and b, see Figure 1.3). For example, one may choose a larger bucket size and a smaller bucket rate, or vice versa, depending on the traffic flow and on the cost of the service. This is not an obvious optimisation problem.

The dynamic VBR problem In the second interval ($[t_1, t_2]$), we encounter an additional problem: how to describe and take into account the fact that the leaky bucket b and the shaping buffer X can be non-empty when S requests to change the service specification *time t_1). The bucket can be non-empty because it is reserved by the traffic that is going to be served. Some traffic can be in the buffer because it did not find service available when it arrived. In Figure 1.4 we plot the evolution of the backlog $w(t)$ and the bucket level $q(t)$ in $[0, t_1]$. At time t_1 both the bucket b and the buffer X are non-empty ($q(t_1) \geq 0, w(t_1) \geq 0$).

When we ignore this aspect (i.e. assume a zero bucket and buffer level at the beginning of the interval) the source is very likely to experience losses in the case of policing. This is due to the mismatch between the assumed bucket and buffer level

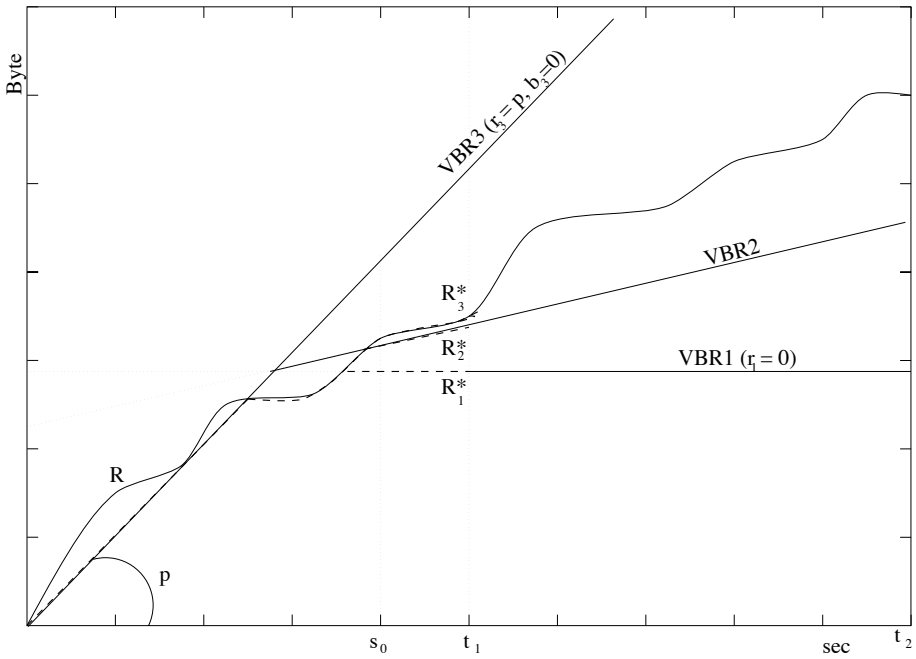


Figure 1.3: The static VBR problem. For a given input traffic $R(t)$, there are several connection descriptors that can carry it. At one end of the spectrum, it is possible to give a large value to the bucket rate, at the limit, make it a CBR (curve VBR_3: $r_3 = p$ and $b_3 = 0$); at the opposite end, a small rate ($r_1 = 0$), with a large bucket size is also possible (curve VBR_1). However, VBR_1 is not acceptable because, after time s_0 it would be necessary a buffer capacity larger than X. VBR_1 and VBR_2 are both valid and the optimum depends by the costs we want to minimise.

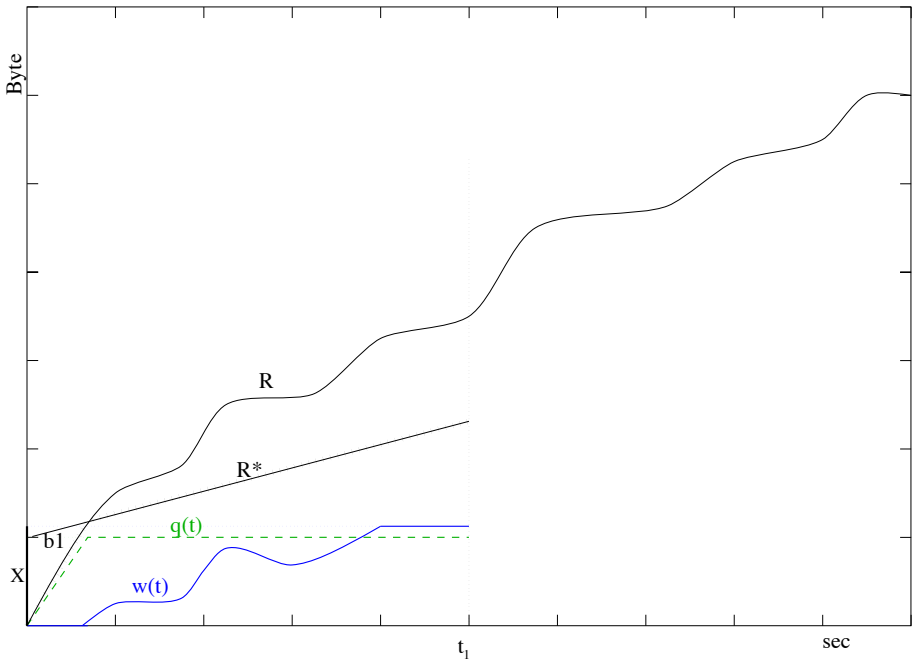


Figure 1.4: The evolution of the backlog $w(t)$ and the bucket level $q(t)$.

and the policed bucket and buffer level. This is illustrated in Figure 1.5.

Main Results of this Work

The main findings in this thesis are

- analytical solutions to the static VBR problem (see Chapter 3) and to the dynamic VBR problem (see Chapters 4 and 5).
- reports on the implementation of the renegotiable services:
  - Renegotiable CBR service: implementation of the host side on Linux (Arequipa) (see Chapter 2),
  - Renegotiable VBR service:
    - implementation of the host side on the EXPERT testbed (RM&R) (see Chapter 3),
    - design of the host side (Armida) (see Chapter 6).

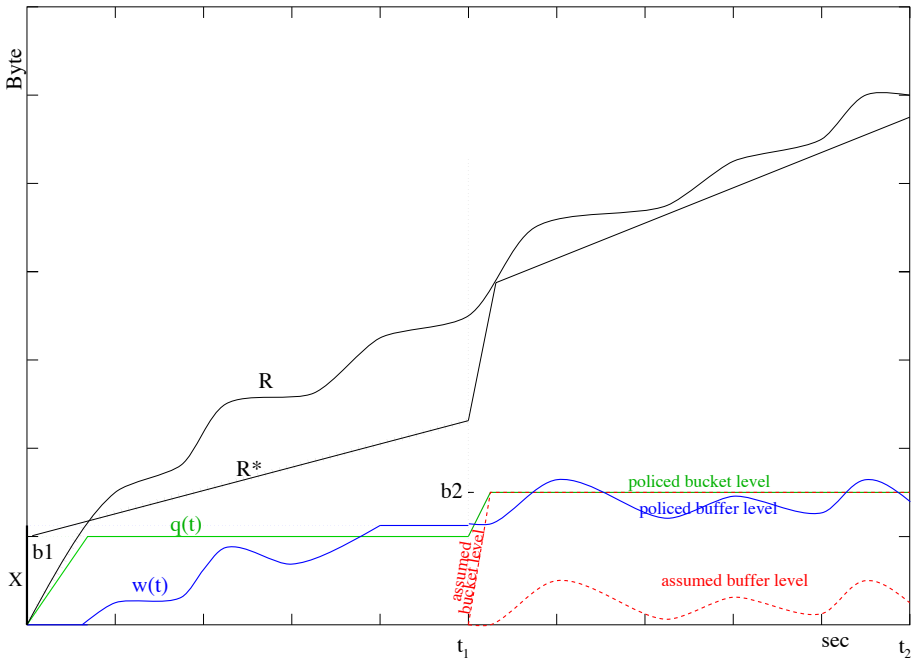


Figure 1.5: The losses experienced by S when the non zero initial conditions were ignored: the assumed buffer and bucket level (dashed lines) are significantly smaller than the policed ones (solid lines).

Technical Approach

The two technical approaches taken in this thesis are:

- Definition of mathematical models:
  - we use network calculus and min-plus algebra to define an analytical model in order to characterise the renegotiable VBR service. This results in simple, efficient algorithms to solve the dynamic VBR problem that can easily be implemented in real applications. See Chapters 4 and 5.
- Simulation and implementation in real cases:
  - we show the validity of the solutions proposed by means of extensive simulations. See Chapters 3 for the simulation of the static VBR problem and 5 for the simulation of the dynamic VBR problem;
  - we also prove the applicability of our solutions with implementation of real cases; see Chapters 2, 3, 6.

Novelty of this Work

The time varying leaky bucket shaper class is characterised by network calculus. To our knowledge, this is the first mathematical model that takes into account non zero conditions at the transient time; see Chapter 4.

The demonstration of the renegotiable CBR is the first example of an application that is able to tune the network QoS (ATM traffic parameters) at run time; see Chapter 2.

To our knowledge, the application described in Chapter 6 is the first design of an application using the renegotiable VBR service.

The results of this work have been published in several papers; see Appendix G for a complete list.

Work Breakdown

Chapter 2 : We describe how *Vic* accomplished ATM traffic parameters tuning over the ATM WAN of SWISSCOM, transferring live video from Lausanne to Basel and Zürich over switched, renegotiable ATM connections. For this purpose, we have modified the popular Mbone tool *Vic* (VIdeo Conferencing) [8] to use *Arequipa* (Application REQuested IP over ATM) [9]. The latter enables applications and in particular *Vic*, to request a direct ATM connection for its exclusive use and to directly control the traffic parameters of this connection. We have also implemented ATM Forum's UNI4.0 signaling and ITU-T's connection modification recommendation Q.2963.1, on end-systems, as well as on switches. This implementation, coupled with the *Arequipa* mechanism, allowed *Vic* to negotiate and renegotiate ATM bandwidth at will and at run time. This work demonstrates the possibility of optimising the network resources by QoS renegotiation: the CBR connection is renegotiated at the source node in order to conform to the traffic issued by the video conferencing application. To our knowledge, this is the first time any application has the capacity to tune ATM traffic parameters at run time.

Chapter 3 : We study the scenario where we have the multiplexing of several VBR input connections over one VBR connection. The multiplexing connection is

called a *VBR* virtual *trunk (VT)* and has a multidimensional connection descriptor. For a given aggregated input traffic, there are several connection descriptors that can carry it. Deciding among all these possibilities requires an additional criterion, minimising a cost objective. Given a cost function for the VBR trunk and a connection admission control (CAC) method for the input connections multiplexed over the VBR trunk, we focus on solving the static VBR problem. First, we show that under reasonable assumptions on the cost function, this optimisation problem can be reduced to a simpler one. Then we consider the homogeneous, loss-free case, for which we give an explicit CAC method. In this case, we find that, *for all reasonable cost functions*, the optimal VBR trunk is either of the CBR type or is truly VBR, with a burst duration equal to the burst duration of the input connections (which case is optimal depends on the cost function and the buffer size X). We take as an example of cost function the equivalent capacity of the VBR trunk [10] and solve the static VBR problem.

Then we design, simulate, implement and perform trials with the Resource Management and Routing (RM&R) architecture built upon the VT solution to the static VBR problem and a dynamic resources management scheme [11], [12], [13] and [14], which estimates the changes in the traffic.

This is a unique example of an advanced resource management and routing architecture simulated and tested in a real ATM environment.

Chapter 4 : We use network calculus and min-plus algebra to define an analytical model in order to define the renegotiable VBR service.

We first characterise a leaky-bucket shaper system with non-zero initial conditions in terms of input-output functions. Then, we study the class of *time varying shapers*. A shaper is time varying if the condition on the output is given by a time varying traffic contract. We focus on the class of time varying shapers called *time varying leaky bucket shapers*. Such shapers are defined by a fixed number of leaky buckets, whose parameters (rate and bucket size) are changed at specific transition moments. We assume that the bucket levels are kept unchanged at those transition moments ("no reset" assumption).

We define the bucket level and the backlog for the time varying leaky-bucket

shaper. By combining these results, we deduce a recursive input-output characterisation of the time varying leaky-bucket shaper.

Before this work, there where no models suitable for the dynamic VBR problem.

Chapter 5 : We introduce the renegotiable VBR service (RVBR) that is characterised by using the results on the time varying leaky-bucket shaper. A flow using the RVBR service is constrained by two leaky buckets: one defines the peak rate, the other defines the sustainable rate and the burst tolerance. Renegotiable VBR services are also studied in [15],[16],[17]; the focus is on describing a given traffic with as few leaky buckets as possible and thus applies to the optimisation of a network offering the RVBR service. Our approach, in contrast, focuses on the customer side of the RVBR service and provides an analysis of the various tradeoffs that can be made. Our work also differs by the systematic use of network calculus. We consider a basic scenario where a fresh input traffic is shaped in order to satisfy the leaky bucket constraints. We further apply our mathematical model to solve the dynamic VBR problem, assuming a perfect knowledge of future traffic.

We provide some algorithms that solve this problem, when the knowledge of the input traffic is limited to the next interval (*local optimisation problem*) and when we dispose of the complete input traffic description (*global optimisation problem*). For the local problem we propose two versions: one when the cost function is represented by a linear cost function and the other when we compare two solutions in terms of the number of connections with those parameters that would be accepted on a link with capacity C and physical buffer X.

Simulation experiments compare the local and global algorithms and show the validity of the local approach. We illustrate the impact of the "no-reset" assumption by analysing on some examples the losses that occur when the source chooses the opposite approach, namely the "reset" approach. Furthermore we simulate the RVBR service versus the renegotiable constant bit rate (RCBR) service and illustrate that the RVBR approach can provide substantial benefits. We also discuss the impact of the size of the renegotiation interval on the efficiency of the RVBR service.

Chapter 6 : We show that the RVBR service is suitable for stored video sources. We first simulate RVBR in the RSVP with Controlled Load (CL) [18] service case. In RSVP the sender sends a PATH message with a *Tspec* object that characterises the traffic it is willing to send. If we consider a network that provides a service as specified for the CL service, the *Tspec* takes the form of a double bucket specification [19] as given by the RVBR service. With RSVP as reservation protocol, the reservation has to be periodically refreshed. When the traffic is known in advance, the renegotiation can be done with the RVBR scheme. There is no additional signaling cost in applying a *Tspec* renegotiation at that point, even if there is some computational overhead due to the computation of the new parameters or to the call admission control, etc.

We simulate this scenario with real video traces by using a 4000 frame-long sequence composed of several video scenes that differ in terms of spatial and temporal complexities. We evaluate the effectiveness of the RVBR algorithm for linear cost function (*localOptimum1*) in terms of cost and backlog.

We also present the design of the implementation of the RVBR service in a Video on Demand application called ARMIDA. This application uses RSVP as reservation protocol and RVBR service to dynamically renegotiate the resources in the network.

We believe that this is the first design of an application that uses the renegotiable VBR service.

Conclusion and Appendices : In Chapter 7 we discuss our main findings, present the conclusion and possible future directions.

The Appendices B, C and D report on the architecture design, the simulation and the trials performed with the Resource Management and Routing (RM&R) architecture.

The Appendix E gives a technical overview of different networking technologies, such as the Internet, ATM and different approaches of how to run IP on top of an ATM network and assesses their potential to be used as an integrated services network. This work evidences the relevance that QoS has in current and future telecommunication technologies.

Chapter 2

Renegotiable CBR Service

This work in this chapter appeared in [20], [21], [22], [23] and [24].

2.1 Introduction

In this chapter, we study a real case where the source node shapes the traffic generated from a video conference application into a renegotiable CBR connection. This is implemented and demonstrated on a public ATM network.

The ATM Forum and the ITU [25, 26] define a large number of ATM connection types, ranging from constant bit rate (CBR) to Available bit rate (ABR). In the work reported here, we focus on the use of *renegotiable CBR* connections because it provides a simple means to offer a visible quality of service under explicit control from the end-user. Renegotiable CBR connections are connections with a maximum peak cell rate, which can be modified by the user at any time (see Section 2.2).

Arequipa or Application REQuested IP over ATM [9, 27] is a method for providing the quality of service of ATM to TCP/IP applications, assuming end-to-end ATM connectivity exists. The *Arequipa* mechanism does not require any changes in the network but two changes do need to be made at the end systems. First, in order to implement *Arequipa* , some changes need to be made to the TCP/IP protocol stack (on the end systems). Second, applications using *Arequipa* to obtain QoS need to be modified slightly to make use of our simple extension to the socket interface. The latter consists of just four calls for setting up, tearing down and

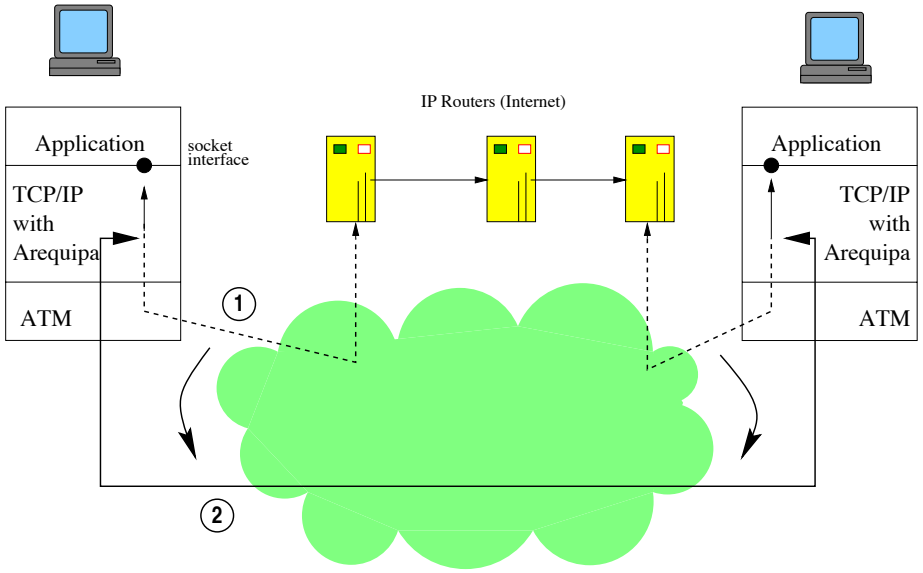


Figure 2.1: *Arequipa* capable applications: data transmission is switched from the default IP path (1) to a dedicated ATM connection(2). An *Arequipa* connection between two ATM attached hosts bypasses intermediate IP routers completely

modifying the traffic parameters of connections. The applications then make use of the end-to-end ATM connectivity and *Arequipa* to set up a direct switched virtual channel connection (SVC) from the sender to the receiver application (Fig. 2.1).

The first applications to be made Arequipa-capable were the Arena Web browser and the CERN httpd Web server [9, 27]. These were used to transfer live video across a trans-european WAN from Lausanne to Helsinki with guaranteed QoS [28]. Since then we have partially implemented UNI4.0 signaling and the Q.2963.1 connection modification capability on the end systems and on switches. Simultaneously we have extended the *Arequipa* mechanism and API to include the connection modification capability. With these an application can now modify the QoS parameters at run time, *after* connection setup.

We demonstrate this in the case of the Mbone VIdeo Conferencing tool *Vic* which we have modified to be Arequipa-capable. *Vic* was used [29] to video conference with QoS, in point-to-point mode, between Basel and Lausanne and also between Zürich and Lausanne. We describe details of this demo which was done over the ATM WAN of SWISSCOM. To our knowledge this was the first time any application had the ability to tune ATM traffic parameters at run time.

Arequipa only works in situations where end-to-end ATM connectivity exists, but this assumption is not unrealistic in Europe where many countries have extensive ATM already deployed, not only in the backbones but also in the LANs.

2.1.1 Chapter breakdown

The following section describes the main features of UNI4.0 signaling and Q.2963.1 connection modification capabilities. Sections 2.3 and 2.4 describe *Arequipa* and *Vic* respectively, while sections 2.5 and 2.6 describe details of implementation issues and the demo.

2.2 Renegotiable CBR Connections

ATM signaling is used by applications to set up SVCs with prescribed QoS. As mentioned in the introduction, we focus here on ATM connections of the CBR class with renegotiation, which we call renegotiable CBR. We refer to renegotiation as the connection modification capability defined in the ITU recommendation Q.2963.1. The latter relates to modifying the traffic parameter of an already active CBR connection. The only connection characteristic that can be modified according to Q.2963.1 is the peak cell rate (PCR).

In order to change the PCR of an active connection without renegotiation, the only possibilities are to (1) open a new connection and to close the old one once the new connection is available, or (2) to close the old connection first and to open the new one afterwards. In (1), the sum of the old and the new bandwidth is allocated for a moment, which may cause the modification to be rejected, although the new PCR alone would be acceptable. Also, data can reach the destination on two distinct paths, so the sequence of cells is no longer guaranteed and some synchronisation is required. In case (2), connectivity is interrupted for a short moment, and, if the new PCR cannot be supported by the ATM network, the connection may be lost entirely.

Renegotiation has neither of those drawbacks and also has shorter latency and less processing, as the modification messages are small and the ATM network only

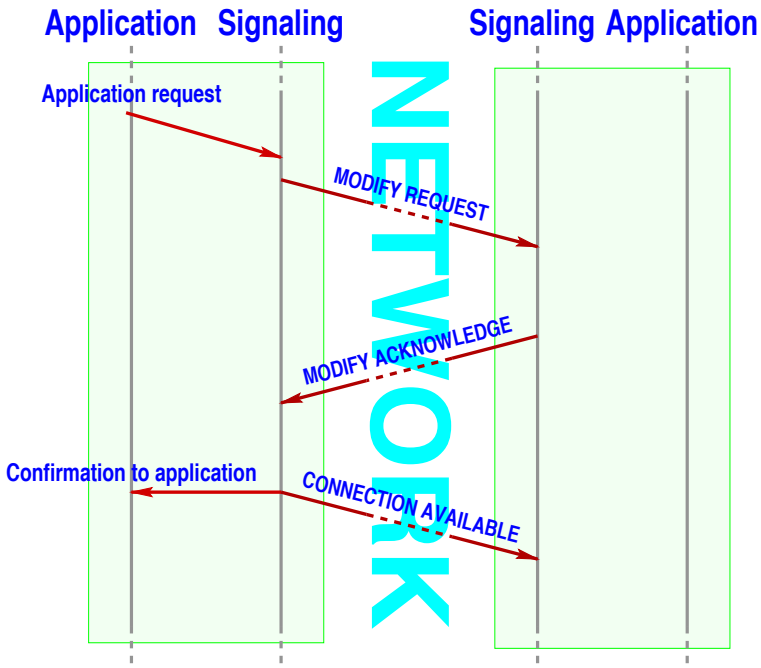


Figure 2.2: Message sequence during bandwidth renegotiation

needs to perform connection admission control but no addressing, routing or other processing required for call establishment.

The exchange of signaling messages during bandwidth renegotiation between two applications is shown in Fig. 2.2. When an application requests a modification, a MODIFY REQUEST is sent out to the called user provided local resources are available. Similarly a MODIFY ACKNOWLEDGE message is returned to the calling user only if resources are available at the called user. The local resources at the calling user are rechecked before the modification is considered to be agreed upon. Then a confirmation is sent to the application and the peak cell rate is changed. A renegotiation requesting a rate increase may fail, in which case the peak cell rate remains at its previous value.

Connection modification is going to be an important capability in future commercial broadband networks, where users will want to control the speed and quality of the data stream because they will be charged for network usage (e.g. bandwidth, time). Modification is especially suited for interactive multimedia applications where bandwidth usage is likely to vary considerably over time.

Renegotiable CBR service is used when we have changes during the connection life time, on a large scale basis. In a video-conference one typical example is the situation where a video moves from the image of a speaker to some presentation on transparencies or on a board.

Note that renegotiation is different from the initial QoS negotiation. The latter capability (defined in UNI 4.0) takes place at call setup time, and consists in allowing network nodes, or the called user, to negotiate the traffic descriptor requested by the calling user. Contrary to negotiation, re-negotiation occurs after the connection is set up.

Renegotiable CBR connections also differ from Available Bit Rate (ABR) connections [30]. With ABR connections, the maximum cell rate (allowed cell rate) is also variable, but here the value of this rate is dictated by the network, not the user. Also, with ABR connections, there is a concept of minimum cell rate. Extension of our work to ABR connections would be interesting but still remains to be done.

At the same time renegotiable CBR service differs from VBR service, where the VBR is better suited for bursty traffic and the renegotiable CBR service is useful when there are a small number of drastic changes on the QoS.

2.3 Arequipa

2.3.1 Overview of Arequipa

The first step in running IP over ATM is to have a means to carry IP packets on ATM. This is mainly an encapsulation issue, defined in RFC 1483 [31]. With this alone, IP can be run over ATM using PVCs. For SVCs, a way to resolve IP addresses to ATM addresses is needed. The IETF currently uses an approach called "classical IP over ATM" (CLIP) that is based on an extension of ARP called ATMARP [32]. The ATM Forum has defined a similar service called "LAN emulation" (LANE) [33] which tries to provide exactly the functionality one would obtain from a LAN, say an Ethernet. Neither CLIP nor LANE are designed to allow applications to benefit from the inherent QoS selection features offered by the underlying ATM network [34].

As described in section 2.1, *Arequipa* is a method for allowing ATM-attached hosts that have direct ATM connectivity to set up end-to-end IP over ATM connections, within the reachable ATM cloud, on request from applications and for the exclusive use by the requesting application. These applications use both an IP and an ATM stack to obtain direct ATM connections (SVCs) with guaranteed QoS. The QoS is guaranteed by the fact that each of these SVCs is used exclusively for one IP flow identified by a pair of sockets (eg. a TCP connection or a UDP stream).

For simplicity, and because multicast transmissions are much less commonly used than unicast transmissions, *Arequipa* presently only supports unicast (point-to-point) operation. Extending Arequipa to support multicast would be straightforward, but it would burden the application with session management tasks if ATM multicast modes other than leaf-initiated join were to be supported.

Arequipa does not require any modifications in the networks (routers, switches etc.) but as mentioned earlier, two important changes need to be made at the end-systems. Some changes need to be made to the TCP/IP stacks at the end systems (discussed briefly in section 2.5.3 and in detail in [27]) and applications need to be modified to use the *Arequipa* socket extensions. It is important to note that if an application is to be QoS aware at all, some code needs to be added somewhere, either in the applications themselves or in some proxies or gateways. We argue that modifying *Vic* to use *Arequipa* is not more complex than modifying it for RSVP [2, 35], and certainly less so than modifying it for native ATM [8].

Arequipa coexists with "normal" use of the networking stacks so that applications not requiring *Arequipa* need not be modified and continue to function as normal.

2.3.2 Functionality

Arequipa adds four simple new functionalities at the socket layer. Applications need to be modified to use just the following four calls in order to set-up, tear-down or modify their ATM SVCs.

- `arequipa_preset(socket, atmaddr, qos)`: establishing or preparing establishment of a new link-layer ATM connection to a given address with a given ATM service and QoS, to make sure that further data sent on the specified

socket, and only data sent on that socket, will use the new ATM connection. arequipa_preset sets up a bidirectional VCC, symmetric or asymmetric, and is only applicable to connection oriented sockets (eg. TCP or connected UDP sockets).

- `arequipa_expect(socket, {true, false})`: preparing a socket to use an incoming *Arequipa* connection for all its outgoing traffic. When a socket receives data from an *Arequipa* connection and arequipa_expect has been set to true, the socket is set to send all its data over the *Arequipa* connection. Again, `arequipa_expect` is only applicable to connected sockets.

- `arequipa_close(socket)`: implicit or explicit closing of *Arequipa* connections. An *Arequipa* connection can be explicitly closed using `arequipa_close` or implicitly closed when the corresponding socket is closed.

- `arequipa_renegotiate(socket, newqos)`: renegotiation of existing *Arequipa* connections. The QoS of an *Arequipa* connection can be modified (increased or decreased) using `arequipa_renegotiate`. The QoS is not modified until the modification is agreed upon.

The `arequipa_preset` and `arequipa_expect` calls are usually made as soon as the application opens sockets for network I/O. The `arequipa_renegotiate` call needs to be made every time the traffic parameters of the *Arequipa* connection are modified.

2.4 VIC

The Mbone [36, 37] VIdeo Conferencing tool *Vic* [38] has a flexible system architecture characterised, among other things, by network layer independence and an extensible user interface. We have used these two features in order to enable *Vic* to use *Arequipa*. The user interface has been modified to include elements for *Arequipa* control and a new network module has been added to set up the appropriate *Arequipa* connections when necessary.

The *Vic* distribution [8] contains three separate network modules for normal IP, native ATM and RTIP (Real-Time IP). Only one of these can be linked at a time, in the current version of *Vic*. The new "*Arequipa* " network module we have added defaults to the normal IP module if *Arequipa* fails for any reason.

With *Arequipa* we only use *Vic* in the point-to-point mode using standard unicast IP addresses, with connected UDP sockets, even though *Vic* was primarily intended as a multi-party conferencing application on the Mbone. For such point-to-point video transfer, both the sender and receiver of video need to know each others' IP address and to agree on a port number. This is normally settled out of band, just as in the case of the IP network module.

2.4.1 User Interface

The main window of *Vic* containing thumbnail views of the outgoing (loop-back) video as well as the incoming video is shown on the left side of Fig. 2.3. Each thumbnail picture is accompanied by identification text, frame and bit rate statistics, and a loss indicator (in parenthesis). The latter is inferred from sequence numbers of the incoming packets. Packets are lost either due to network drops or due to local socket buffer overflows resulting from CPU saturation. Fig. 2.3 is taken in Basel, using the new "*Arequipa* " network module but before starting *Arequipa*. It shows a loss rate of the incoming video from Lausanne (over the Internet) to be well over 50%, even over relatively low bit rates.

Details of the *Vic* control panel (obtained by clicking on the menu button in the main window) are shown on the right side of Fig. 2.3. We have extended the menu by adding a section on top for *Arequipa* control. By design only the sender of video can initiate an *Arequipa* session, by selecting the service category and the bandwidth (converted internally into PCR) and clicking on the *Arequipa* button [1].

In line with the spirit of keeping the visible QoS simple, the only user specifiable traffic parameter is the bandwidth and CBR/UBR the only choice of service categories. Once an *Arequipa* connection is in place, it it not possible to change the

[1] note that the *Arequipa* mechanism itself does not carry any constraint as to which side can initiate *Arequipa*.

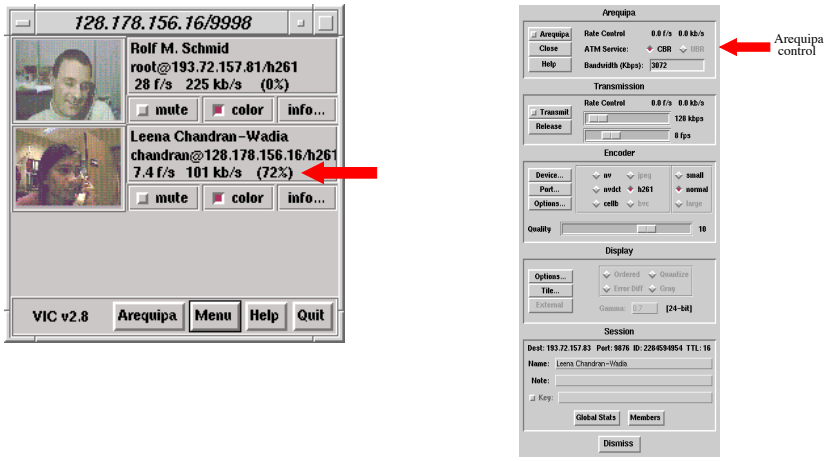


Figure 2.3: Main window and control panel of *Arequipa* capable *Vic*

service category (e.g. from CBR to UBR), but only to change the traffic parameter. Each time the content of the bandwidth entry is modified by the user, a call to `arequipa_renegotiate` is made automatically.

The capture (and encode) frame and bit rates are displayed above the QoS settings. The close button only closes the *Arequipa* connection and video transmission continues via the normal IP path. As in the unmodified *Vic* the release button has to be used to stop video transmission.

Arequipa can be started any time during (IP) video transmission, e.g. when picture quality deteriorates. If normal IP transmission has not already begun when *Arequipa* is started then *Vic* first starts it before switching over to *Arequipa*. Then, if *Arequipa* fails for any reason, transmission reverts to the normal Internet path.

2.4.2 Networking

As described in detail in [38] *Vic* uses the Real-time Transport Protocol, RTP [39], which is realised completely within *Vic* itself. RTP is divided into two components: the data delivery protocol and the control protocol RTCP. The former handles the actual media transport and the latter manages control information like sender identification, receiver feedback and cross-media synchronisation.

We have added a new network module which contains, apart from the normal

IP module, procedures for the exchange of ATM addresses, for opening, closing and renegotiating *Arequipa* connections. Note that in order to make the connection, the `arequipa_preset` call requires the knowledge of the ATM address of the peer machine (section 2.3.2. Details of the implementation of a procedure for the exchange of ATM addresses depends very much on the application itself e.g., in the case of a Web browser and a Web server we used HTTP for the address-exchange, see [27].). In the demo version of *Vic* this exchange was done out of band. A newer version contains a generic address-exchange module in which the exchange is performed over TCP when *Arequipa* is first requested. So far *Arequipa* connections have only been set up for the data channel. We decided against using an *Arequipa* for control traffic too, because the latter, being low volume, would typically only benefit from an *Arequipa* connection in extreme high-loss situations, e.g. caused by severe congestion, which should be rare.

2.5 Implementation

ATM signaling support, including *Arequipa* and the connection modification capability at the end systems has been implemented for PC's running the Linux operating system [40]. Signaling and connection modification has also been implemented on the ATMLightRing switch of ASCOM.

2.5.1 ATMLight Ring (ASCOM)

The ATMLightRing is a campus and metropolitan area backbone developed by ASCOM. Physically, it consists of a dual-fiber ring interconnecting a number of Access Nodes across a campus or city area. Each node provides standard ATM and non-ATM user interfaces to which various communication equipment with standard interfaces can be connected.

Logically, an ATMLightRing is a high-speed transport backbone providing full connectivity at each port. It appears to the network as a single distributed switch, thereby greatly reducing system management complexity. It allows interconnection of switches, routers, hubs, concentrators, servers, PBXs, workstations and WAN

access devices.

For the experiments with renegotiable ATM connections, the control software of the ATMLightRing was enhanced to support the UNI4.0 signalling capabilities together with the Q.2963.1 connection characteristics negotiation and modification.

2.5.2 ATM on Linux

ATM on Linux is a comprehensive implementation of ATM-related protocols, including the latest signaling as specified in ATM Forum UNI 4.0. This platform is also used for the reference implementation of *Arequipa* (see below) and for experiments with renegotiation as specified in Q.2963.1.

The ATM on Linux distribution containing full source code for kernel changes, system programs and test application is available publicly [40].

2.5.3 Arequipa

Arequipa has also been implemented in the Linux operating system and is part of the ATM on Linux distribution [41] of ICA. For the establishment and the release of ATM connections the signaling parts of classical IP over ATM have been reused. A new virtual network device for *Arequipa* has been created and a few modifications have been made to the socket layer.

The implementation, detailed in [42] builds upon the route cache entry in the socket descriptors. This entry stores a pointer to the interface to which all data sent from a socket has to be forwarded. This is normally done to avoid doing an IP route look-up every time a datagram is sent. By setting and locking this route cache to point to the *Arequipa* device it is ensured that any further data sent from the socket goes to the *Arequipa* device. An additional field has been added to the socket descriptor to store a pointer to the VCC on which the data should be transmitted. When it receives a datagram from IP, the *Arequipa* device simply sends the datagram on the VCC indicated in the socket descriptor.

`arequipa_preset` is implemented as a library function which does the following. First it asks the signalling demon to establish a VCC with a given destination and

QoS. Then it enters a pointer to that VCC in the socket descriptor and makes the IP route cache of the socket point to the *Arequipa* device.

`arequipa_expect` simply sets a variable to indicate whether the application wants to use an incoming *Arequipa* connection for its outgoing traffic. Every time a datagram is received on an *Arequipa* VCC, the variable is tested. If it is set, the route cache is set to point to the *Arequipa* device.

`arequipa_renegotiate` requests renegotiation for the VCC attached to the specified socket, using the common native ATM procedures. It blocks until the renegotiation completes (with or without success). Other processes or threads can continue sending and receiving on the socket while `arequipa_renegotiate` is in process. Note that there is no explicit notification for renegotiation initiated by the peer.

2.6 Demonstration over an ATM WAN

Vic was used to transfer live video from Lausanne to Basel and Zürich over SWISSCOM's public ATM network. The demo was a cooperative effort between the Web over ATM project [43] of the EPFL and the ACTS-EXPERT project funded by the European Commission, and was conducted on 24th October 1997.

2.6.1 Setup

The demonstration network consisted of three ATM equipped Linux terminals located in Lausanne, Basel and Zürich, connected to a (two node) ATMLightRing System at the EXPERT testbed in Basel. The three sites were interconnected by the VP connection provided by SWISSCOM's public network. The demonstration network between Basel and Lausanne is illustrated in Fig. 2.4. None of the intermediate switches except the ATMLightRing supported the UNI4.0 and the Q.2963.1 signaling capabilities. The VP connection from SWISSCOM provided seamless connectivity between the end systems and the ATMLightRing.

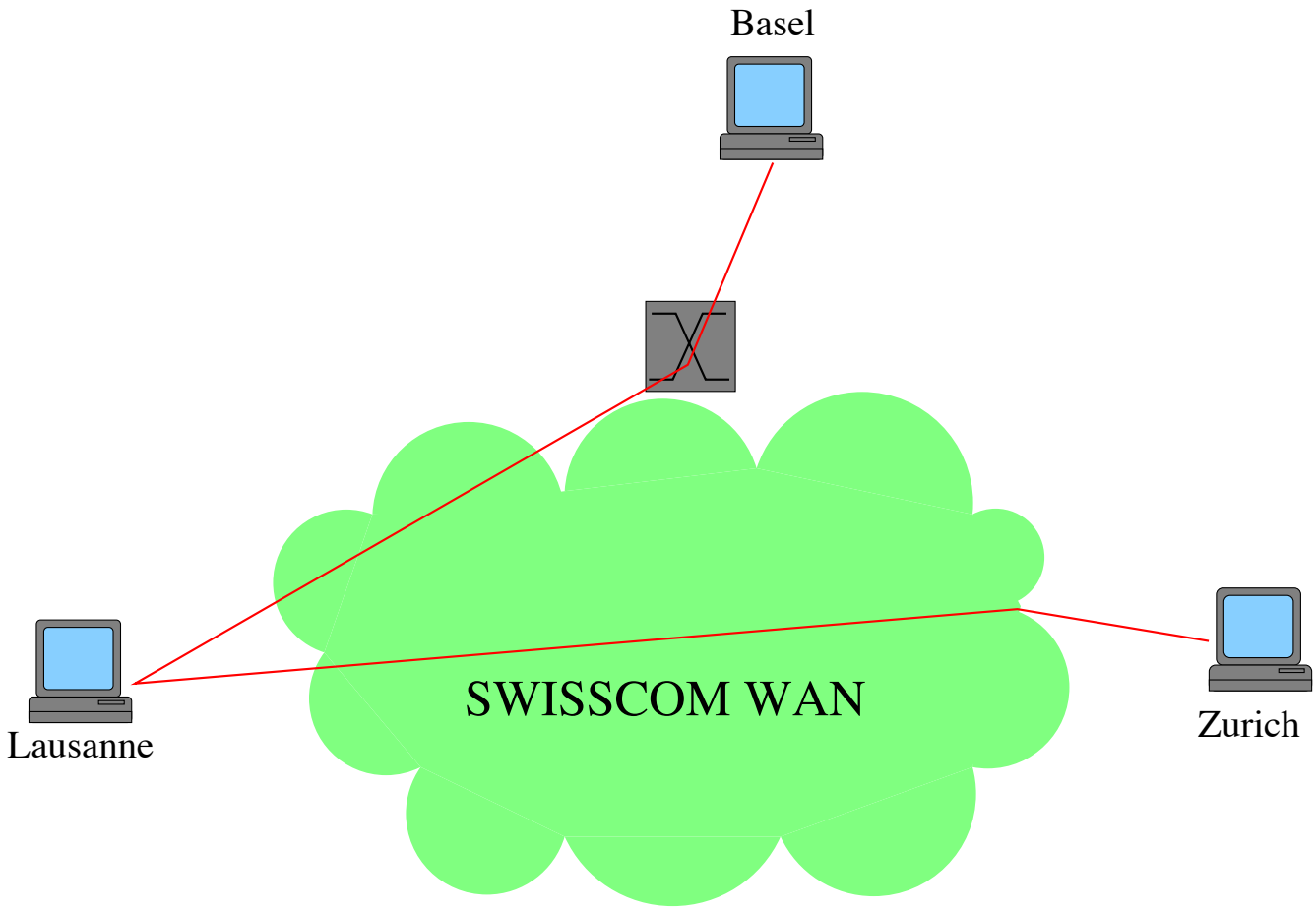


Figure 2.4: Topology of the network used in the demo

2.6.2 Observations

As shown by the loss indicator of the incoming video in Fig. 2.3, the loss rates over the Internet can often be well over 50% even at relatively low bandwidth. Switching over to *Arequipa* at such times results in a dramatic change in the quality of the video, because then network losses in congested routers no longer occur. At low bandwidths the loss rate drops to zero immediately. At much higher bandwidths losses make their appearance once again, this time due to CPU saturation. The threshold at which losses appear depend on the system configuration of course, but also on the type of encoding being used.

2.7 Conclusion

We have shown that in areas where end-to-end ATM connectivity already exists, there is a relatively simple way for IP applications to use the QoS of ATM, specifically, by using *Arequipa* and its simple API. We have demonstrated this in the case of *Vic* which also became the first instance of an application to tune bandwidth at run time.

We argue that in some cases, applications can benefit from renegotiable CBR service, especially when a single connection is used for video transfer where distinct sequences with different QoS requirements are identifiable: for instance, a tele-teaching session where the image of the professor's speech is followed by some illustration (e.g transparencies).

If many changes in resources requirements occur frequently, then the renegotiable CBR service is probably not the more appropriate service, from a pure billing point-of-view.

With end-to-end ATM connectivity, there is an alternate way of providing hard QoS guarantees to applications. This specifically enables them to use ATM, resulting in a so-called *native* ATM application. In the case of *Vic*, this has been done and *Vic* can run on ATM using the FORE SPANS API. In general, the effort involved in converting an IP application into a native ATM one can often be significant. In the case of *Vic* this was certainly so. The native ATM network module is completely

different from the normal IP module, whereas the *Arequipa* network module is merely an extension of it, using the same sockets and so on.

The IETF way of providing quality of service (QoS) guarantees to the applications in IP networks today is to use the ReSerVation Protocol (RSVP) [2]. As with *Arequipa* and with native ATM, the applications need to be modified in order to make them QoS sensitive. *Vic* [38] has also been RSVP enabled [35] and can provide soft guarantees on the QoS, depending on the extent of RSVP deployment on the intermediate routers. With the work reported here, we believe to have demonstrated that presenting explicit quality of service to applications and their users is indeed simple and can be deployed as soon as networks exist which support reservations.

Some other issues related to this chapter are discussed in next chapters. Among them:

1. solutions for QoS renegotiation with VBR connections (Chapters 3, 5 and 6).

2. solutions for QoS renegotiation over heterogeneous networks (e.g networks using ATM, RSVP or SRP protocols) involving horizontal and vertical mapping among the different protocols used to obtain QoS. This is the aim of the work of the ACTS-DIANA project [44]. Part of this work is reported in the Section 6.4 of Chapter 6.

Chapter 3

The static VBR problem

This work in this chapter appeared in [45], [46], [47], [48] and [49].

3.1 Introduction

In this chapter we solve the problem of negotiating an optimal VBR service for the incoming traffic by introducing the virtual trunk (VT) concept.

Then we present an architecture that aims to provide dynamic VBR service to the input traffic while using the proposed solution to the static VBR problem derived with the VT concept. This is obtained by combining this solution with a dynamic resources management scheme that estimates the changes in the traffic. The result is a virtual trunk that changes its own connection descriptor dynamically.

We summarised results and limitations of this architecture. The detailed architecture and the complete simulation and trials results are presented in Appendices B, C and D.

3.1.1 VBR over VBR, Multiplexing and Virtual Trunks

We consider the multiplexing of several variable bit rate (VBR) connections (called "the input connections") over one variable bit rate connection (called "the VBR trunk"). This occurs, for example, with ATM when a number of VBR virtual channel connections (VCCs) are multiplexed over one virtual path connection (VPC)

[50], which is also of the VBR type. Another example is the multiplexing of several IP flows with reservations (using a protocol such as RSVP [51] or ST.II [52]) over one ATM VCC. A generalisation to any type of input traffic is given in the introduction of Chapter 4.

We are interested in such multiplexing scenarios because we believe that reducing the number of connections (or reserved flows if RSVP is used) is a key feature that will be needed in all large scale networks. This is because connection handling costs, especially network management overhead, processing, and memory is not negligible and increases almost linearly with the number of connections handled at one point. One solution is to aggregate connections at all points where possible. Connection aggregation simplifies all aspects of connection handling, provided that it is possible to dynamically change the attributes of the multiplexed connections [53] [54]. Aggregation can take place: (1) at an ATM node performing aggregation of VCCs over a VPC; (2) at an IP router aggregating several reserved flows over one ATM connection; (3) at an IP router aggregating several reserved flows over one reserved flow (tunnelling). We call a *Virtual Trunk* (VT) the connection that multiplexes a number of other connections; the word "trunk" refers to the fact that those connections also have attributes of network internal links, as defined for example with P-NNI [55]. In case (1), VTs are VPCs, in case (2), VTs are VCCs, and in case (3), they are IP tunnels with reserved resources. In this chapter we use mainly ATM terminology, which applies strictly to case (1) only (VT can thus be equated to VPC). Translation to cases (2) and (3) should nevertheless be straightforward. We call a *multiplexer* the node that multiplexes several input connections on one output VT.

Virtual trunks have traditionally been considered as Constant Bit Rate connections, though this restriction is not mandatory. In contrast, using other traffic types has obvious benefits. In this chapter, we consider VTs of the VBR type. The rationale for using VBR VTs is the following: integrated services packet networks provide resource reservation; however, they will not allocate its peak rate to every individual connection, but perform resource overbooking. At the lowest level, overbooking uses both buffering (traffic peaks are temporarily stored) and statis-

tical multiplexing (based on the expectation that traffic peaks do not all occur at the same time). [56]. If only CBR VTs are used, then access or edge nodes that multiplex small or medium numbers of connections are not able to perform a large amount of statistical multiplexing because efficient statistical multiplexing requires a small ratio between connection rate and the VT bit rate [57]. Furthermore, the CBR VTs have to be allocated their peak rate by intermediate nodes that multiplex them in turn, since such nodes do not have any information about the individual input connections. With CBR VTs, overbooking is thus performed mainly by burst absorption at the multiplexer. In contrast, if VBR VTs are used, then it is possible to let bursts go through the multiplexer and count on statistical multiplexing inside the network, where the number of connections and the trunk bit rates are larger. Quantifying this statement is not simple; it requires the definition of a connection admission control (CAC) method for connections over a VBR VT; it is beyond the scope of this chapter.

In the rest of the chapter, we consider only VBR VTs and simply refer to them as "VBR Trunks". We consider only input connections of the VBR type (which includes CBR but leaves aside ABR or UBR connection types). As explained in detail later, we focus on the problem of how to define the VBR trunk parameters in order to admit VBR input connections while minimising the cost of the VBR trunk.

A virtual trunk is considered as a connection by the network supporting it and as a trunk by the connections it supports. Two sets of parameters are associated to virtual trunks: *connection descriptor* and *trunk state*.

- **Connection descriptor** is composed by the traffic and class parameters that describe the traffic characteristic of the VT when it is considered as one single connection. It is used by the supporting network to accept VTs.

- **Trunk state** (also called metrics) is the set of trunk state parameters reflects the static and dynamic characteristics of the VT. It is used for accepting connections on the virtual trunk.

In this chapter, the connection descriptor for VBR VTs (and for the VBR input connections) consists of the sustainable bit rate (Mbit/s), the burst tolerance (s),

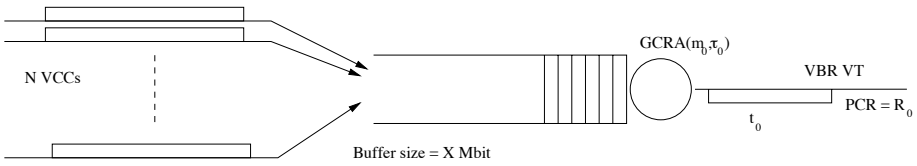


Figure 3.1: Reference Configuration

the peak bit rate (Mbit/s), and the cell delay variation (s) [58], herein referred with the tuple (m, τ, R, CDV)[1] in order to be compliant with the ATM simulation and trials presented in the later in this chapter.

The trunk state depends on the CAC method used to accept input connections on the virtual trunk (we call this method "VT-CAC"). In Section 3.3, we give a VT-CAC for the homogeneous, loss-free case, based on fluid models. VT-CACs for heterogeneous cases and for supporting statistical multiplexing with losses, are summarised in Chapter 4 and presented in [59].

3.1.2 The VBR-over-VBR Optimisation Problem

The reference configuration used in this chapter is shown on Figure 3.1. A multiplexer, fed with a number of input connections of the VBR type, multiplexes them into one VBR connection (the VBR trunk), using a buffer of size X. There is no explicit assumption, so far, on the service discipline for the buffer. But we assume that the buffer output is regulated so that the resulting traffic conforms to $GCRA(1/R_0, CDV_0)$ and $GCRA(1/m_0, \tau_0 + CDV_0)$ [60].

The connection descriptor is multidimensional. For a given mix of input connections, there are several parameter sets that can carry them. This problem already exists for CBR virtual trunks, where several values of (peak rate, cell delay variation tolerance) are possible [61]. Here, we neglect cell delay variation tolerance issues and focus on supporting burst tolerance. At one end of the spectrum, it is possible to give a large value to the sustainable rate of the VBR trunk, at the limit, make it a CBR trunk; at the opposite end, a small sustainable cell rate, with a large burst tolerance is also possible. Lastly, the peak cell rate of the VBR trunk also influences

[1] from Section 3.3 we neglect CDV

all other parameters. Deciding among all these possibilities requires an additional criterion, minimising a cost objective. In our reference model, the cost objective is given by a function of the VBR trunk connection descriptors only. Given a cost function for the VBR trunk and a connection admission control (called VT-CAC) method for the input connections multiplexed over the VBR trunk, our problem is to find the VBR trunk connection descriptor that minimises the cost function and is able to accept a given set of input connections. We call this special case of the static VBR problem the *VBR-over-VBR* optimisation problem and *VT solution* the solution of this problem obtained by using the VT concept.

3.1.3 Dynamic Virtual Trunks

We assume that the multiplexer accepts incoming connection requests in real time and is able to change its own VT connection descriptor dynamically (by negotiation with the network supporting the virtual trunk). An example of dynamic virtual path connections that uses ABT/DT [62] for reserving resources in the CBR VT case is defined in [54]. As a result, we require that the computation of the optimal VBR trunk connection descriptor be simple enough to be performed in real time.

Therefore, we address the problem of providing a dynamic VBR service to the input traffic while using the VT concept. This is performed by coupling the VT solution with a dynamic resources management scheme [11], [12], [13] and [14], which estimates the changes in the traffic. This results in a virtual trunk that changes its own connection descriptor dynamically (by negotiation with the network supporting the virtual trunk).

Here we present the design, simulation, implementation and trials performed with the Resource Management and Routing (RM&R) architecture built upon this combination of the VT solution and the dynamic resources management scheme.

To our knowledge, this is a unique example of an advanced resource management and routing architecture that was simulated and tested in a real ATM environment.

In the reference ATM network, a VT is a virtual path connection (VPC) setup by the network for reducing connection awareness at the transit nodes. A virtual

trunk is therefore considered as a connection by the network supporting it (the VP network) and as a logical trunk by the connections supported. We have a tunnelling scenario where a number of VBR Virtual Channel Connections (VCCs) are multiplexed onto a VBR VPC at a node that acts as a general shaper and the VBR VPCs are multiplexed on the network. This is called the *VBR-over-VBR* approach. We compute the VPC traffic descriptor according to the VT solution.

The simulation and trial results confirm the validity of the dynamic approach.

3.1.4 Chapter breakdown

In Section 3.2, we define the VBR-over-VBR optimisation problem formally and show how it can be simplified by identifying a subset that necessarily contains all possible optimal solutions. This is true under reasonable assumptions for the cost function. In order to further progress in the solution, we need a VT-CAC method; in Section 3.3, as a starting point, we propose such a method for the simple case where all input connections are identical and there are no losses. We study the properties of this VT-CAC and apply the results of Section 3.2. We obtain that the optimal VBR trunk in that case is either a CBR trunk or a VBR trunk with burst duration equal to that of the input connections. We also obtain a simple relation (Eq. 3.5) that relates the total buffer size at the multiplexer, the burst tolerance of the input connections of the VBR trunk and the gain obtained by having a VT sustainable cell rate higher than that of the aggregate input. In Section 3.4, we complete the study in the case where the cost function is the equivalent capacity [10] of the VBR trunk (considered as one connection). The equivalent capacity is one cost function that reflects the cost of the VBR trunk to the supporting network. We also give in Section 3.4.3 a complete example illustrating the various aspects of the method presented in this chapter. In Section 3.5 we give an overview of the architecture obtained by coupling the VT solution and the dynamic resources management scheme. Its complete description is given in Appendix B. We report on the simulation and trials result with this architecture.

3.2 Reduction of the VBR-over-VBR Optimisation Problem

Having given the motivation for multiplexing a set of VBR connections on a single VBR connection we can now define in more detail the problem we investigate and then perform a first reduction. For a VT trunk, we indicate with z its trunk state and with $y = (m, \tau, R, CDV)$ its connection descriptor.

We assume that the VT-CAC for the trunk under consideration can be expressed by a real valued function $F(y, z)$. This function returns a non-negative value if the trunk with descriptors y can accept the traffic described by the trunk state z and a negative value otherwise. We denote by $c(y)$ the cost function that gives the cost of a connection with descriptors y. This function is given and we make the following (common sense) assumption on it:

1. $c(y)$ is non-decreasing with respect to all components of y, namely, if $y \preceq y'$ then $c(y) \leq c(y')$;

The problem described in the previous section can be formalised as follows: given trunk state z, we want to find among all connection descriptors y for which $F(y, z) \geq 0$, the connection descriptor y_{opt} which minimises $c(y)$, if it exists. Define the *feasible region* $\mathcal{FR}(z)$ for every trunk state z as:

$$\mathcal{FR}(z) = \{\text{connection descriptors } y : F(y, z) \geq 0\}$$

We can now express our problem as follows:

$$\text{find} \quad y_{opt} \in \mathcal{FR}(z) : \forall\, y' \in \mathcal{FR}(z) c(y') \geq c(y_{opt}) \tag{3.1}$$

It is convenient to use the *partial order* on the set of connection descriptors defined by:

$$y{=}(m, \tau, R, \text{CDV}) \preceq y'{=}(m', \tau', R', \text{CDV'}) \text{ iff } \begin{cases} m \leq m' \\ \tau \leq \tau' \\ R \leq R' \\ \text{CDV} \leq \text{CDV'} \end{cases}$$

We now make the following (common sense) assumptions on the VT-CAC that, together with the assumption that the cost function is non-decreasing, will allow us to show that y_{opt} can be found in a set much smaller than $\mathcal{FR}(z)$.

Assumptions

1. the VT-CAC function $F(y, z)$ is continuous with respect to y;
2. the set of connection descriptors y for which $F(y, z)$ and $c(y)$ are defined is a closed, convex subset of $(\mathcal{R}^+)^d$, with d equal to the number of coordinates of the vector y, and it contains $y_0 \preceq y \ \forall y \in \mathcal{FR}(z)$ for which $F(y_0, z) \leq 0$.

Now we can proceed with the reduction of the set of values for y where the optimum is found, if it exists. Let us assume that there exists y_{opt} that solves the VBR-over-VBR optimisation problem 3.1. Assume that $F(y_{opt}, z) > 0$. Consider the set $A = \{\alpha_i\}$, $A \subseteq [0, 1]$ such that $F(y_0 + \alpha_i(y_{opt} - y_0), z) \geq 0$. By the assumptions on F, this set contains the value 1 since y_{opt} is in the feasible region and, by the assumptions on F, it is non-empty, closed and thus compact. Therefore, A has a minimum value, call it a. If $F(y_0 + a(y_{opt} - y_0), z) > 0$, then necessarily $a > 0$ because for $a = 0$ $F(y_0 + a(y_{opt} - y_0), z) = F(y_0, z) \leq 0$ for the second assumption on $F(., z)$. By the continuity of $F(., z)$, we can find some a' such that $0 < a' < a$ and $F(y_0 + a'(y_{opt} - y_0), z) > 0$; we have a contradiction because a was assumed to be minimum. Therefore $F(y_0 + a(y_{opt} - y_0), z)$=0. Now by the non-decreasing property of c, we have that $c(y_0 + a(y_{opt} - y_0))$=$c(y_{opt})$. This is not possible because y_{opt} is assumed to be optimum. Therefore, if the optimum exists, then it is certainly reached at a point y with $F(y, z)$=0.

We further reduce the set of possible solutions by considering *non dominated* points in $\mathcal{FR}(z)$. We say that $y \in Y$ is non-dominated in the set Y if $\forall y' \in Y$, $y' \preceq y \ \Rightarrow \ y'$=$y$.

Let us assume again that there exists a traffic descriptor y_{opt} that solves the VBR-over-VBR optimisation problem (3.1). Consider now the set $\mathcal{E}$ of connection descriptors y' that are feasible and dominate y_{opt}, namely, $\mathcal{E}$=$\{y' \in \mathcal{FR}(z) \ : \ y' \preceq y_{opt}\}$. By the non-decreasing property of c, all points in this set are also optimal.

We now proceed with showing that at least one point in this set is non-dominated in $\mathcal{FR}(z)$. This set is closed and by the second assumption, $\mathcal{E}$ is non-empty and

compact. Therefore, there exists at least one point y_1 in $\mathcal{E}$ that minimises the first coordinate. Indicate with $p_i(y)$ the i^{th} coordinate of a traffic descriptor y. Call $\mathcal{E}_1$ the set of all y' in $\mathcal{FR}(z)$ that dominate y_1. Obviously, $\mathcal{E}_1 \subset \mathcal{E}$ and $p_1(y)=p_1(y_1)$ for all $y \in \mathcal{E}_1$. By applying the same procedure recursively we build a sequence of decreasing sets $\mathcal{E}_k$ and points y_k, such that $p_1(y)=p_1(y_k), \cdots, p_k(y)=p_k(y_k)$. Ultimately, when k equals d, we have $\mathcal{E}_d=\{y_d\}$ and thus y_d is non-dominated in $\mathcal{FR}(z)$ and realizes the optimum for c.

In summary, we define the set $\mathcal{S}(z)$ (for Solution space) by:

$$\mathcal{S}(z)=\{y : F(y,z)=0 \text{ and } \forall\ y' \in \mathcal{FR}(z) : y' \preceq y \ \Rightarrow\ y'=y\}$$

We have shown that if there exists a solution to problem given in Equation (3.1), then it is in the solution space $\mathcal{S}(z)$. Our problem can thus be reformulated in the following way:

$$\text{find } y_{opt} \in \mathcal{S}(z) : \forall\ y' \in \mathcal{S}(z) : c(y') \geq c(y_{opt}) \tag{3.2}$$

This simplification is independent of the cost function, provided that the common sense assumptions are satisfied. Under this form it is, in general, easier to find y_{opt}. The solution space is a limited subset of the feasible region and it depends on a smaller number of variables since at least one can be expressed as function of the others from $F(y,z)=0$. The condition that the elements of S(z) be non-dominated in $\mathcal{FR}(z)$ further restricts the solution space.

3.3 Homogeneous, Loss-less VT-CAC: General Results

Here we apply the reduction of the preceding section to the homogeneous case, specifically when all input connections are identical. We give an explicit function F for that case, based on a loss-less (or worst case) CAC. We assume the worst case traffic of one input connection as the pattern consisting of a burst at the maximum rate for the maximum allowed time, followed by a silent period (ON/OFF). We know

that this is not the general worst case [63], [64] and [65]. But in the homogeneous case, it requires the same amount of resources than the effective worst case, and it is easier to study.

This section clearly represents only a first step towards the resolution of the general case, however, it is complex enough to be worth investigating in detail.

First, we give an algorithm for VT-CAC, then we apply the results of Section 3.2.

3.3.1 VT-CAC function for the Homogeneous, Lossless Case: requiredBuf

In this Section we present a deterministic CAC function to decide the acceptance of VBR traffic, regulated by a shaping buffer with a fixed buffer size X under no cell loss. We assume that the buffers are large compared to the size of the cells, such that we can ignore the Cell Delay Variation Tolerance. We also assume that the traffic is homogeneous; meaning that all the input connections multiplexed on the VBR trunk have the same connection attributes: m, τ and R. The number of input connections is indicated by N. The VT attributes are thus defined by:

- Trunk state: $z = (N, m, \tau, R)$
- Connection descriptor: $y = (m_0, \tau_0, R_0)$.

The VT traffic is smoothed by the associated shaping buffer such that it conforms to $GCRA(1/m_0, \tau_0)$, as shown in figure 3.1.

We define requiredBuf(y, z) as the buffer size required for accepting the input traffic on the VT with zero cell loss. Thus a connection can be accepted iff

$$X - \text{requiredBuf}(y, z) \geq 0$$

which defines the function F.

To avoid cell loss, we consider the worst case: the input connections are synchronised and send data all together at the peak cell rate until the GCRA reacts. At the beginning, the buffer is assumed to be empty. We analyse the problem from the aspect of the required buffer size, identifying six different situations. Two cases are evident:

- if $Nm > m_0$, the buffer length must be infinite, requiredBuf=∞ (CASE 1)
- if $NR < m_0$, there is no need of buffer, requiredBuf=0 (CASE 2).

Beyond these two cases, we examine the quantity of traffic that can be absorbed by the VT burst and we deduce the buffer size required to buffer the remaining traffic. The burst lengths are given [60] by:

$$t_{burst} = \frac{\lfloor \tau/(T - 1/R) + 1 \rfloor}{R}$$

where $T = 1/m$. We assume that the effect of integer cells (the factor $+1$ in the numerator), is negligible compared to the burst size. When $NR < R_0$, the burst length of the VT is considered for traffic equal to NR, because this is the maximum traffic generated by input connections. Thus, the burst length of the VT is given by:

$$t_0 = \tau_0 m_0/(NR - m_0)$$

When $NR > R_0$, the burst length of the VT is considered for traffic equal to R_0, because this is the maximum traffic that the VT can absorb. Thus, in this case, the burst length of the VT is given by:

$$t_0 = \tau_0 m_0/(R_0 - m_0)$$

The burst length of the input connections is given by:

$$t_c = \tau m/(R - m)$$

Either $t_0 \geq t_c$ or $t_0 < t_c$, moreover, we have to consider $NR > R_0$ and $NR < R_0$:

- $NR < R_0$, $t_0 < t_c$ (CASE 3, Figure 3.2)
  We see easily that:
  requiredBuf = $(NR - m_0)(t_c - t_0)$
- $NR < R_0$, $t_0 > t_c$ (CASE 4)
  requiredBuf = 0

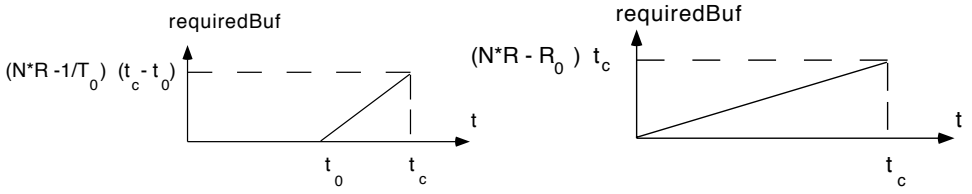


Figure 3.2: Buffer filling when $NR < R_0$, $t_0 < t_c$ (CASE 3) and when $NR > R_0$, $t_0 > t_c$ (CASE 6), respectively.

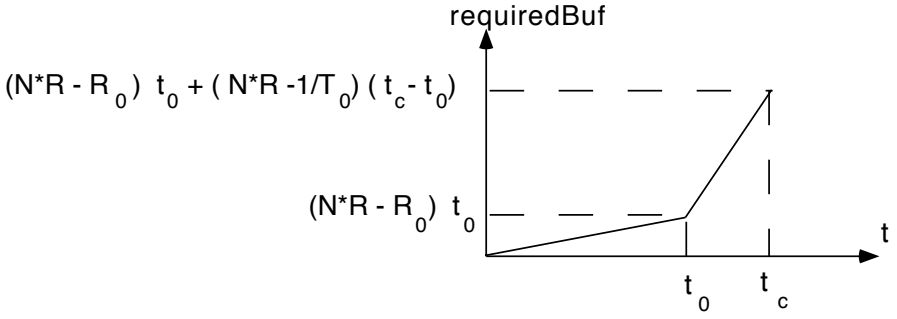


Figure 3.3: Buffer filling when $NR > R_0$, $t_0 < t_c$ (CASE 5).

- $NR > R_0$, $t_0 < t_c$ (CASE 5, Figure 3.3)

  Figure 3.3 shows that:

  requiredBuf $= (NR - R_0)t_0 + (NR - m_0)(t_c - t_0)$

- $NR > R_0$, $t_0 > t_c$ (CASE 6, Figure 3.2)

  Figure 3.2 shows that:

  requiredBuf $= (NR - R_0)(t_c)$

RequiredBuf is thus defined by the following algorithm:

Algorithm 3.1 : requiredBuf

if $Nm > m_0$ **then** requiredBuf $= \infty$ CASE 1
else if $NR \leq m_0$ **then** requiredBuf $= 0$ CASE 2
else if $NR \leq R_0$ **then**
  if $t_0 < t_c$ **then** requiredBuf $= (NR - m_0)(t_c - t_0)$ CASE 3
    else requiredBuf $= 0$ CASE 4

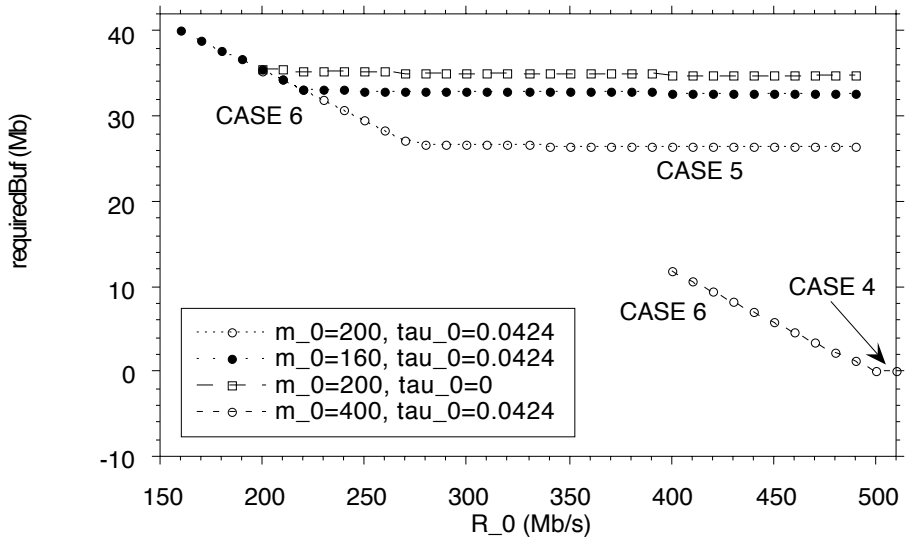


Figure 3.4: Evolution of requiredBuf versus R_0 in the worst case with τ=0.0424 sec., $m = 3$ Mbit/s, R=10 Mbit/s, N=50

else if $t_c \geq t_0$ **then**

requiredBuf $= t_0(NR - R_0) + (t_c - t_0)(NR - m_0)$ CASE 5

else requiredBuf $= t_c(NR - R_0)$ CASE 6

3.3.2 Analysis of the RequiredBuf Function

RequiredBuf has some interesting aspects that we discuss in this section. In Figure 3.4, 3.5 and 3.6, we plot requiredBuf versus each of the three connection attributes of the VT m_0, τ_0 and R_0. Analysing the curves in Figure 3.4, we note that R_0 affects the buffer size only for values smaller than the rate of the input connections burst ($NR = 500$) and the burst length of the VT smaller than the the burst length of the input connections. In this case, requiredBuf decreases when R_0 increases. The slope of the curve is $\partial\text{requiredBuf}/\partial R_0 = -t_c$, constant and negative for every value of m_0, τ_0. In the other cases, requiredBuf remains constant for any value of R_0, therefore it is useless to increase R_0. The slope of the curve is always zero.

Note that R_0 must always be larger than or equal to m_0. When $R_0 = m_0$ or $\tau_0 = 0$, the type of the VT connection is CBR. The buffer has to absorb all the bursts from the input connections exceeding m_0. In these cases, requiredBuf only

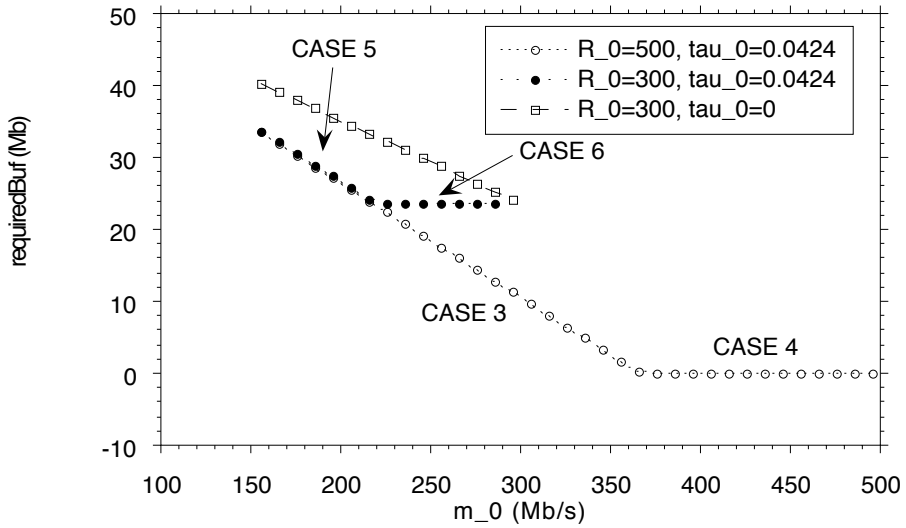


Figure 3.5: Evolution of requiredBuf versus m_0 in the worst case with τ=0.0424 sec., $m = 3$ Mbit/s, R=10 Mbit/s, N=50

depends on m_0.

As shown by Figure 3.5 and Figure 3.6, m_0 and τ_0 affect requiredBuf only for cases in which $t_0 \geq t_c$ because m_0 and τ_0 influence the burst length and not the rate of the VT. For this reason, after having reached the equality between the burst lengths, any increase of m_0 or τ_0 is not significant. In fact, for smaller input connections burst lengths (CASE 3 and CASE 5) $\partial\text{requiredBuf}/\partial(m_0) = -t_c - \tau_0$ and $\partial\text{requiredBuf}/\partial\tau_0 = -m_0$ in both cases. In the other cases, $\partial\text{requiredBuf}/\partial(m_0)$=$\partial\text{requiredBuf}/\partial\tau_0$=0.

The feasible region for a fixed buffer size B, VT connection descriptor $y = (m_0, \tau_0, R_0)$, N input connections with attributes $\{m, \tau, R\}$, thus VT trunk state $z = (N, m, \tau, R)$, is given by:

$$\mathcal{FR}(z) = \{y : B - \text{requiredBuf}(y, z) \geq 0\}$$

In Section 3.2 we showed that, under some reasonable assumptions on the VT-CAC and cost functions, the VBR-over-VBR optimisation problem can be simplified. We will now try to apply this simplification and to find the solution space when the VT-CAC is requiredBuf.

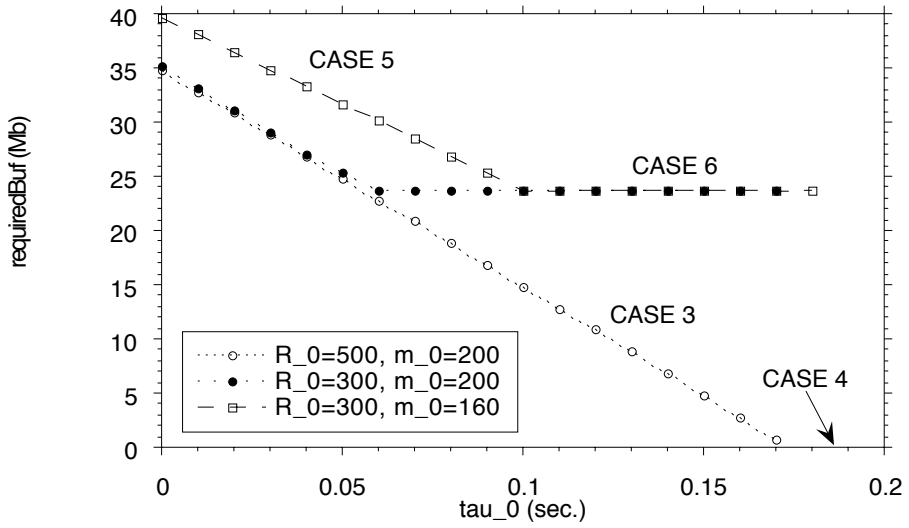


Figure 3.6: Evolution of requiredBuf versus τ_0 in the worst case with τ=0.0424 sec., $m = 3$ Mbit/s, R=10 Mbit/s, N=50

3.3.3 Solution Space $\mathcal{S}(z)$

In Section 3.2 we used two common sense assumptions on $F(y, z)$ to reduce our optimisation problem. The assumption of continuity of requiredBuf is proven in Appendix A. The other assumption, namely that there be a descriptor y_0 in the definition domain of $F(., z)$ for which $F(y_0, z) \leq 0$, requires some special attention. The domain of connection attributes for which requiredBuf is defined is $R_0 \geq m_0$ and $\tau_0 \geq 0$ by definition and $m_0 \geq Nm$ for reasons of stability. Depending on the value of X and z it may happen that there is no y_0 for which $F(y_0, z) \leq 0$. Physically, this is the case when the shaping buffer is large enough to absorb all the bursts of the input connections ($X \geq N\tau m$). Any VT with a sustainable rate larger or equal to Nm is thus sufficient to support the output traffic without loss, $F(y, z) > 0$ for all y of the definition domain of $F(., z)$ and the reduction is not applicable. However, since all y satisfy the CAC function, y_{opt} is simply the lower bound of the definition domain of requiredBuf. Thus we have that for X larger than $N\tau m$, the optimal VT connection descriptor is a CBR connection with a sustainable rate of Nm:

$$X > N\tau m \Rightarrow y_{opt} = (Nm, 0, Nm) \ \text{(CBR)} \tag{3.3}$$

When $X \leq N\tau m$ we can always find a y_0 for which $F(y_0, z) \leq 0$, for example $(NR - X/t_c, 0, NR - X/t_c)$. The assumptions of Section 3.2 are thus valid and we can apply our reduction: the solution space $\mathcal{S}$ is a subset of $\mathcal{FR}$ and contains only the non dominated y in $\mathcal{FR}$ for which $F(y, z) = 0$.

From the equality $B - \text{requiredBuf}(y, z) = 0$, we can express one variable in function of the others. We chose to express τ_0 in function of m_0 and R_0 (see [66]):

$$X - \text{requiredBuf}(y, z) = 0 \Rightarrow \tau_0(m_0, z) = \frac{NRt_c - X}{m_0} - t_c$$

We see that τ_0 is independent of R_0 and can be expressed in function of m_0 and the trunk state only. Furthermore we see that for $m_0 > NR - X/t_c$, $\tau_0(m_0, z)$ must be negative and outside the definition domain of requiredBuf for F to be zero. Physically this means that, for a value of m_0 above $NR - X/t_c$, the part of the input bursts which is above m_0 does not fill the shaping buffer. The implication is that for a connection descriptor y, m_0 must be less or equal to $NR - X/t_c$ for y to be in the solution space $\mathcal{S}$.

$$F(y, z) = 0 \Rightarrow m_0 \leq NR - X/t_c$$

We also find that, given m_0 and $\tau_0(m_0, z)$, $F(y, z)$ is 0 for any value of R_0 larger than $NR - X/t_c$, as demonstrated in [66]. The solution space is made of non dominated elements of $F(y, z) = 0$, thus R_0 must be $NR - X/t_c$, which dominates all larger values of R_0. Note that this value is always in the definition domain of requiredBuf since m_0 is can not be larger than $NR - X/t_c$.

We can thus express the solution space by the following equation:

$$\mathcal{S} = \{(m0, \frac{NRt_c - X}{m_0} - t_c, NR - X/t_c)\} \tag{3.4}$$

$$\text{with } Nm \leq m_0 \leq NR - X/t_c \text{ and } X < N\tau m$$

Note that for m_0 being at its upper bound, the VT becomes a CBR connection.

Discussion: In this section we have seen that when the shaping buffer exceeds the size of the input bursts ($N\tau m$) the connection descriptors which minimise the

the cost of the VT are simply those of a CBR connection with sustainable rate equal to the sum of the sustainable rates of the input (Nm). If the shaping buffer is smaller than the input bursts, we can reduce feasible region of requiredBuf from an open three-dimensional space to a limited one-dimensional solution space. The condition that $F(y, z)$ must be zero allows to express one variable (τ_0) in function of one other (m_0). Furthermore, the condition that the solution space be made of non-dominated elements allows to fix the third variable (R_0) to the lowest bound, which is independent of the other two variables. One physical implication of this is that R_0 being at the lowest bound, the duration of the burst at the output of the multiplexer is at the highest bound, which is when the output burst has the same duration as the input burst (see Corollary 1 in [66]). From this property, we deduce a simple equation which is valid for the optimal solution for any cost function that conforms to the assumptions of Section 3.2. The equation relates the buffer size at the multiplexer, the burst tolerance of the input connections and of the VT and the sustainable cell rate of the VT, thus for $y{=}y_{opt}$:

$$X = N\tau m - \tau_0 m_0 - (m_0 - Nm)t_c \tag{3.5}$$

In particular, when $m_0 = Nm$, we have that the burst of the input traffic is completely absorbed by the buffer and the burst of the VPT.

3.4 Homogeneous, Loss-less VT-CAC: $c(y)$ equal to Equivalent Capacity

In this section we continue the analysis of the homogeneous case with a specific example as cost function. We consider a system that uses *requiredBuf* for the input connection admission control over VTs, and the *Equivalent Capacity* function [10] for the cost function. We show how the computation is reduced and simplified by applying the results of Section 3.2.

3.4.1 Cost Function: Equivalent Capacity

The cost function we use here is the Equivalent Capacity function defined in [10]. It is defined as the the rate necessary for achieving a desired buffer overflow probability ϵ, on a given physical link, given a physical link buffer size X^l. Note that X^l is *not* related to the buffer size, noted X, at the multiplexer. It the context of this Section, it should be simply interpreted as a parameter of the cost function that influences the cost of a given connection descriptor of the VT. If X^l is very large, then the cost is mainly influenced by the VT sustainable rate m_0; if it is very small, then it is mainly influenced by the peak rate R_0. In contrast, X influences the output of the requiredBuf function.

The equivalent capacity c_0, for a VT connection descriptor $y = (m_0, \tau_0, R_0)$ is given by:

$$c_0 = R_0 \frac{Y_0 - X^l + \sqrt{[Y_0 - X]^2 + 4X^l \rho_0 Y_0}}{2Y_0} \tag{3.6}$$

where

$$Y_0 = \ln(\frac{1}{\epsilon})\tau_0 m_0 \text{ and } \rho_0 = \frac{m_0}{R_0} \tag{3.7}$$

We do not prove the monotonicity of Equivalent Capacity. We just argue that, as a typical effective capacity function, EC must increase when any one of its parameters increase.

3.4.2 Application of the Space Reduction

In Section 3.3.3, we have identified the solution space $\mathcal{S}(z)$. We can now formulate the general solution for in this specific case as an optimisation problem depending on one single variable m_0, as follows:

$$\text{find } m_0 \in [Nm, NR - X/t_c] \text{ that minimises } g(m_0) \tag{3.8}$$

where $g(m_0)$ is given by:

$$\frac{\ln(\frac{1}{\epsilon})(t_c(NR - m_0) - X) - X^l + \sqrt{\frac{4Xt_c m_0}{B - NRt_c}(B + t_c(NR - m_0)) + (\ln(\frac{1}{\epsilon})(t_c(NR - m_0) + B - X))^2}}{2t_c \ln(\frac{1}{\epsilon})(t_c(NR - m_0) + B)/(B - t_c NR)} \tag{3.9}$$

This function is a non decreasing function, thus, in absence of constraints on m_0, the solution of the VBR-over-VBR optimisation problem can be easily found for the lower bound of m_0 ($m_0 = Nm$), as illustrated in the next section.

3.4.3 Numerical Example

Here we provide three numerical examples of the VBR-over-VBR optimisation problem where the cost function used is equivalent capacity and also show the complete interaction of the elements of the method defined in this chapter.

In the first example, the parameters used for defining the equivalent capacity function are $X^l = 100$ Mb and the cell loss probability $\epsilon =$ 1.0E-05. The capacity of the shaping buffer at VT1 is $X = 2Mbit$ and defines a feasible region $\mathcal{FR} = \{\text{requiredBuf}(y, z) \leq 2\}$. The current attribute values for the VT are:

$$z = (N, m, \tau, R) = (8, 3, 0.77, 20),$$

$$y = (m_0, \tau_0, R_0) = (87.2162, 0.1529, 145.2941),$$

The equivalent capacity of the VT is thus 89.3416 Mbit/sec.

Assume now that two new input connection requests arrive at the virtual trunk. By accepting the two new connections, the trunk state z would become:

$$z' = (N, m, \tau, R) = (10, 3, 0.77, 20);$$

This would move the trunk attribute out of the feasible region. (requiredBuf(y , z')=7.44 Mb). Thus we want to find a new connection descriptor such that the new connections can be accepted, namely, the new VT connection attributes belong to $\mathcal{FR}(z)$ and the cost function on the links is minimised.

From Section 3.4.2 we know that our solution is the one that minimises $g(m_0)$. We set R_0 to its lower bound $R_0 = NR - X/t_c = 185.2941Mb/s$. The resulting solution space is plotted in Figure 3.7. Minimisation of g is found by minimising m_0. In this case, we find that the minimum of cost is 127.079 Mb/s; it is obtained for $(30, 0.704, 185.2941)$, as shown on the dashed curve in Figure 3.8.

In a second example, the traffic input is the same, but we assume a larger value for the parameter X^l ($X^l = 500Mb$) of the equivalent capacity function, which means

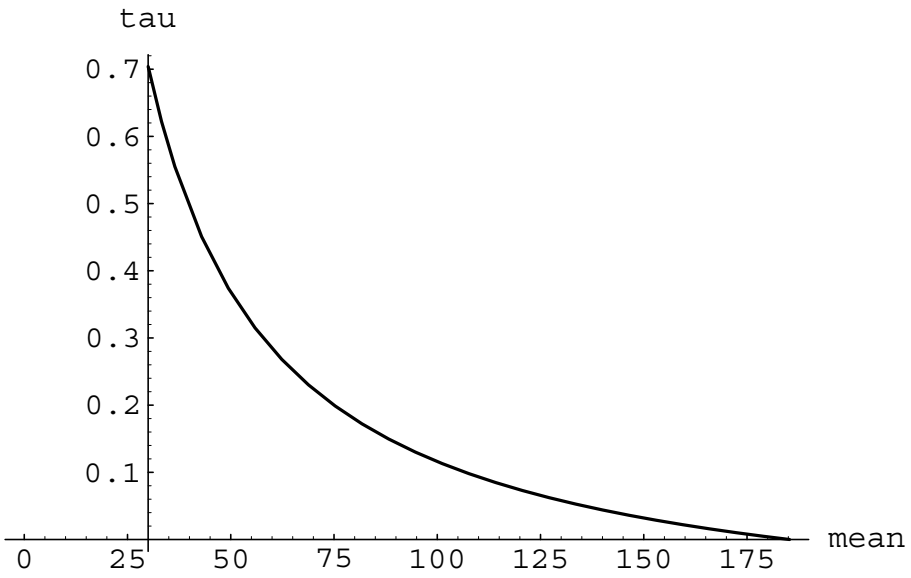


Figure 3.7: The solution space $\mathcal{S}(z)$ for the numerical example. The third parameter R_0 is equal to $NR - X/t_c$ =185.2941 Mb/s.

that the cost of a large burst tolerance is not as high as in the first example. All other parameters are kept unchanged. Thus we expect that the optimal solution will have a smaller cost. The numerical result confirm this expectation: the minimum ($c = 47.0774$) obtained for $(30, 0.704, 185.2941)$ is smaller than before. This case is represented by the dotted curve in figure 3.8.

In the last case we assume a still larger value for X^l ($X^l = 1000Mb$) with all other parameters kept unchanged. As expected, the cost of the optimal solution $(30, 0.704, 185.2941)$ still decrease and becomes very close to m_0 ($c = 32.6137$). As expected, in figure 3.8, the curve relative to this case, the solid curve, is an increasing straight line.

3.5 The Resource Management and Routing architecture

The solution to the static VBR problem, derived with the virtual trunk concept (called *VT solution*), is designed for use in a static way, specifically at the initial

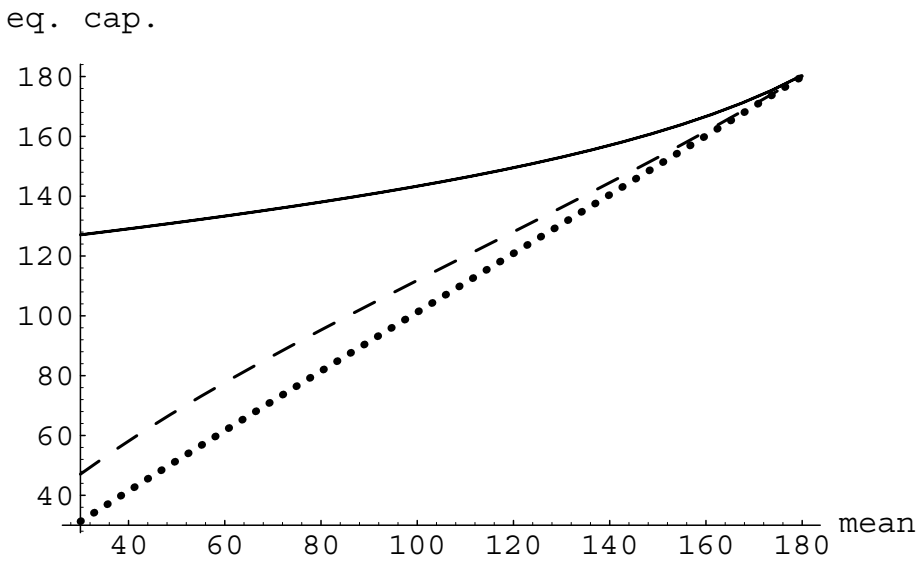


Figure 3.8: The cost function on the solution set $\mathcal{S}(z)$, for three different values of the cost function parameter X^l: $X^l = 100$ (dashed curve), $X^l = 500$ (dotted curve), $X^l = 1000$ (solid curve). Small values of X^l give a high cost to VTs with large burst tolerance. The optimal VT parameter is obtained for the minimum of the sustainable bit rate ("mean" on the figure). If bursts are more expensive (smaller X^l) then the optimal virtual trunk with the same sustainable cell rate has higher cost. The peak rate optimal value is fixed by the results of Section 3.3.

phase (negotiation). Although this performs better than a static CBR solution, we argue that renegotiation is a mandatory feature.

Therefore, we consider the problem of providing dynamic VBR service to the input traffic while using the VT solution.

The scenario is an ATM network: a VT is a virtual path connection (VPC) considered as a connection by the VP network supporting it and as a logical trunk by the supported connections. VTs are of VBR type.

Here, for each VPC, we have a tunnelling scenario where a number of homogeneous VBR Virtual Channel Connections (VCCs) are multiplexed onto a VBR VPC at a node that acts as a general shaper and the VBR VPCs are multiplexed on the network. This, in line with the work in the previous sections, is called the *VBR-over-VBR* approach. In the more traditional VBR-over-CBR approach, VBR VCCs are multiplexed onto a CBR VPC. In this case, it is no longer possible to

multiplex the VPCs on the network.

We use the VT solution for computing the VT connection descriptor for the next period. To this aim we estimate the number N of homogeneous connection for the next interval. This is performed with a dynamic resources management scheme [11], [12], [13] and [14], which estimates the changes in the traffic. The combination of the VT solution and the dynamic resources management scheme results in a virtual trunk that changes its own connection descriptor dynamically (by negotiation with the network supporting the virtual trunk).

The Resource Management and Routing (RM&R) architecture is built upon this combination of the VT solution and the dynamic resources management scheme. The details of the RM&R are given in Appendix B. We simulate this architecture and then implement it for performing trials on the EXPERT testbed located in Basel.

To our knowledge, this is a unique example of an advanced resource management and routing architecture simulated and tested in a real ATM environment.

By simulation, we compare the VBR-over-VBR approach to a more traditional VBR-over-CBR approach. We find that the novel approach using VBR VTs performs better than the traditional one using CBR VTs. The gain in our example is not very significant, because of the rather inefficient method (Equivalent Capacity) used for calculating the effective bandwidth and the unsophisticated use of a static allocation scheme to achieve a dynamic reallocation approach. The simulations are reported in Appendix C.

In Appendix D we present the conceptual description of the implementation, as well as relevant parts of its specification. Then we report on the trials performed on a ATM network. The results show advantages of the bandwidth reallocation over approaches that are limited to static allocation.

However, in both simulation and trials cases, the results were obtained under special conditions, where basically no losses were experienced. Therefore, if, on one hand, these results justify the choice of a dynamic resource management scheme, on the other hand, they also highlight the necessity of a more sophisticated and appropriated scheme for the renegotiation as presented in next chapters.

In fact, the RM&R architecture is based on the assumption that, at the moment when the VPC traffic descriptors are changed, there occur no losses if, for each VPC, the shaping buffer and the bucket are resetted. This is equivalent to assuming that, at this moment, there is no traffic into the bucket and into the shaping buffer. In this specific case, this assumption is valid. In fact, because of the approximation introduced in the various algorithms, the VPC traffic descriptors are always much larger than the real optimum. Therefore, at the the transition moment, the VPC has been transmitting for a long time at the sustainable rate (or smaller rate), experiencing no losses when renegotiated.

Situations where there are losses could not be reproduced with our architecture, because of the approximation introduced by the various algorithms. At the the transition moments, the buffers and buckets were always empty producing no losses when renegotiated.

In spite of that, it is clear that the usage of any solution to the static VBR problem in a dynamic scenario can potentially produce relevant losses in a system, which is not affected by similar approximation problems as we demonstrate in Chapter 5.

3.6 Conclusion

We analysed in this chapter one of the consequences of having VBR trunks in an integrated services network, which we argued is an essential feature for reducing connection handling costs. We have formalised the problem of determining optimal VBR trunk connection descriptors, given a CAC method for accepting input connections on the VBR trunk (VT-CAC). We have shown how the optimisation problem can be reduced to a simpler problem and applied the result to the homogeneous case. For the specific case of a cost function equal to the equivalent capacity, we derived a complete analysis, with simple, closed form formulas that can easily be implemented for real time computation. We shown that, for the homogeneous case, and for all reasonable cost functions, the optimisation problem can be reduced to a one-dimensional problem.

Then we presented the Resource Management and Routing (RM&R) architecture

that provides dynamic VBR service to the input traffic. This architecture is obtained by combining the VT concept with a dynamic resources management scheme that estimates the changes in the traffic. The result is a virtual trunk that changes its own connection descriptor dynamically.

We summarised simulation and trial results, as well as the limitations of this architecture, presented in detail in Appendices B, C and D.

As already mentioned, to our knowledge, this is a unique example of an advanced resource management and routing architecture that was simulated and tested in a real ATM environment.

Chapter 4

Time Varying Leaky Bucket Shapers

This work in this chapter appeared in [67] and [68].

4.1 Introduction

4.1.1 Network Calculus background

The network calculus theory ([69], [70], [71]) provides powerful tools to manage guaranteed services. The concepts of an *arrival curve* and a *service curve* allows us to characterise a shaper in terms of mathematical functions and min-plus algebra.

Consider a data flow, described by the number of bits sent in $[0, t]$ $R(t)$. Given a wide-sense increasing function $\alpha(.)$, we say that a flow $R(t)$ has an *arrival curve* of $\alpha(.)$ iff for all $s \leq t$: $R(t) - R(s) \leq \alpha(t - s)$, for all s and t [69].

Given a nondecreasing function $\sigma(.)$, we say that a system S offers to a flow $R(t)$ a *service curve* $\sigma(.)$ iff, for all $t \geq 0$, there exists $0 \leq v \leq t$ such that $R^*(t) - R(t) \geq \sigma(t - v)$, where $R^*(.)$ represents the output of the system [72], [70], [71].

The *min-plus convolution* of $R(.)$ and $\sigma(.)$ is defined as $(R \cdot \sigma)(t) = \inf_v\{R(v) + \sigma(v - t)\}$

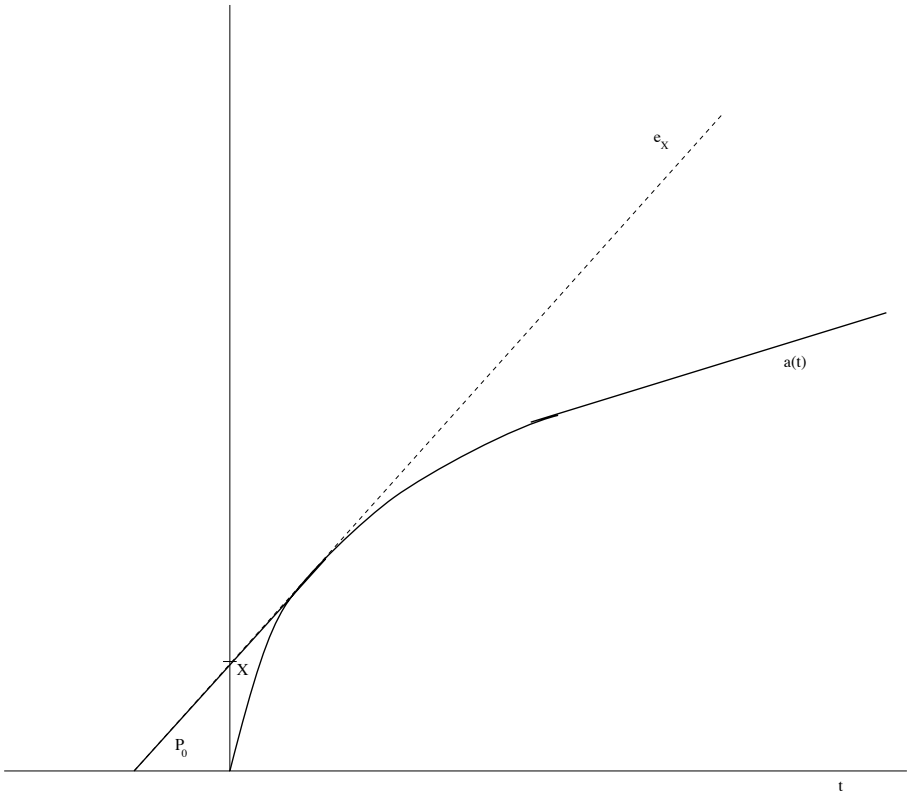


Figure 4.1: The static VBR optimisation problem seen with network calculus

Time Invariant Shapers We can easily express the static VBR problem in terms of network calculus concepts.

The class of shapers suitable for modelling this problem is the *time invariant shaper* class. A shaper is time invariant if the traffic constraint is defined by a fixed traffic contract. Time invariant shapers are extensively studied in [70] and [7].

The tunnelling scenario presented and solved in Chapter 3 can be reformulated as follows:

- the input traffic is expressed with an aggregated arrival curve α
- the time invariant shaper, which shapes the input traffic into the virtual trunk with arrival curve $\sigma = \min(P_0 t, m_0 t + \tau_0)$, offers to the input traffic a service curve σ (Theorem 3 of [7]).

As illustrated in Figure 4.1, the requirement on the service curve σ, given the aggregate arrival curve α of the input traffic is (Theorem 2 of [7]):

$$\min(P_0 t, m_0 t + \tau_0) + X \geq \alpha(t) \qquad \forall t \geq 0 \tag{4.1}$$

That allows to derive the same results as in Section 3.

Corollary 1 (Optimal P_0 for VBR Virtual Trunk) *Defined the arrival curve of the input traffic $\alpha(t)$, independently from the cost function, the smallest value for the peak cell rate of the VBR virtual trunk under buffer constraint expressed by Equation 4.1 is $P_0 = e_X$, with e_X the deterministic equivalent capacity corresponding to the arrival curve $\alpha(t)$ assuming a buffer size X:*

$$e_X = \sup_{t \geq 0} \frac{\alpha(t) - X}{t}$$

Corollary 2 (Optimal Service Curve) *Given $\alpha(t)$, X, the constraints at equation (4.1) and a cost function on the virtual trunk parameter $c(.)$, the service curve $\sigma(t)$ for the virtual trunks which minimise $c(.)$ and are feasible with the buffer requirements is obtained by solving the problem with only one variable (m_0) defined as follows:*

$$\textit{minimise}\ c_{m_0}(m_0) = c(m_0, \tau_0(m_0), e_X) \tag{4.2}$$

and the optimal solution is given by:

$$y_0 = (m_0, \tau_0(m_0), e_X) \tag{4.3}$$

These equations are independent from the nature of the input traffic that is expressed in terms of its aggregated arrival curve α. Therefore, all the previous results apply not only to homogeneous VBR traffic, but to any tunnelling scenario with a generic input traffic with arrival curve α [7].

However, time invariant shapers are not suitable for modelling the dynamic VBR problem. In this case the input traffic changes dynamically and the network resources must change consequently. With time invariant shapers, it is not possible to characterise the traffic that is present in the shaper at the transient moments.

Therefore, in this chapter, we use network calculus concepts to model a class of shapers suitable to define the renegotiable VBR service. This class is a special class of time varying shaper systems that we call the *time varying leaky-bucket shapers*.

4.1.2 Notation

A time varying leaky-bucket shaper is defined by a fixed number J of leaky bucket specifications with bucket rate r^j and bucket depth b^j, where $j = 1, \ldots, J$ and a shaping buffer of fixed capacity X. At specified time instants t_i, $i = 0, 1, 2, ..,$ the parameters of the leaky buckets are modified.

The observation time is thus divided into intervals and $I_i = (t_i, t_{i+1}]$ represents the i-th interval. For each $t \geq 0$ there exists an $i \in \mathbb{N}$ such that $t \in I_i$. The time instants t_i are given, but the length of the intervals can be variable as, for example, in the case where it is estimated by means of some measurement.

Inside each interval the system does not change. The parameters of the j-th leaky buckets valid in the interval I_i are indicated by (r_i^j, b_i^j). The combination of those parameters takes the form of the shaping function σ_i in I_i, defined as

$$\sigma_i(u) = \min_{1 \leq j \leq J} \{\sigma_i^j(u)\} = \min_{1 \leq j \leq J} \{r_i^j \cdot u + b_i^j\}$$

A time varying leaky-bucket shaper is completely defined by:

- the number J of leaky buckets

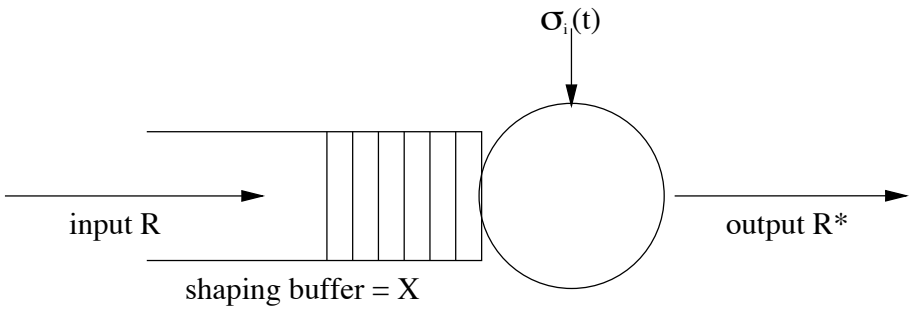


Figure 4.2: Reference Model for a time varying leaky-bucket shaper. The traffic shaping at time $t \in I_i$ is done at source according to the service curve σ_i valid in I_i.

- the time instants t_i at which the parameters changes
- the buckets parameters (r_i^j, b_i^j), for each j and each interval I_i
- the fixed shaping buffer capacity X

We call *input traffic function* the function $R(t) : \mathbb{R}^+ \to \mathbb{R}^+$ that represents the amount of traffic that has entered in the system in time interval $[0, t]$. R is the traffic before the shaping. $R^*(t)$ is the *output function* that represents the number of bytes seen on the output flow in time interval $[0, t]$. R^* is the traffic after the shaping. We assume to know the input traffic $R(t)$ expected in the future either because pre-recorded or by means of an exact prediction function. However the traffic prediction is not the focus of this work. We further assume that at time $t_0 = 0$ the system is idle ($R(0) = 0$).

To define the time varying leaky-bucket shapers at the transient times between two adjacent intervals we could take two opposite approaches: either we reset all the buckets and restart in the next interval from zero initial conditions ("reset" approach), or we keep the level of the buckets and restart from that level at the next interval ("no reset" approach). If we take the first approach, the time varying leaky-bucket shaper can be reduced to a sequence of independent shapers and studied as described above [73], [7]. However, as we described in Section 4.3, this approach cannot guarantee the service.

Therefore, here we adopt the second approach. There are two more reasons for this. First, in the special case where the time varying leaky-bucket is constant, we

should find a system identical to the ordinary, time invariant, leaky bucket shaper [73], [7]. And this should be possible only with the second approach. Second, the "no reset" approach is in line with the Dynamic Generic Cell Rate Algorithm (DGCRA) used to specify conformance at the UNI for the available bit rate (ABR) service of ATM [25], [74]. We examine later in the chapter the practical implication of the "no-reset" approach (Section 5.3.4).

Our class of time varying shapers is a special case of the general concept of time varying shapers, defined in [75]. A general time varying shaper can be defined as follows. Given a function of two time variables $W(s,t)$, the time varying shaper forces the output $R^*(t)$ to satisfy the condition

$$R^*(t) \leq R^*(s) + W(s,t)$$

for all $s \leq t$, possibly at the expense of buffering some data. This condition can be expressed using the min-plus linear operator associated to W and defined as the mapping $S \rightarrow S \cdot W$ with $(S \cdot W)(t) = \inf_s\{S(s) + W(s,t)\}$. The shaper is an optimal shaper if it maximises its output among all possible shapers [75]. A time invariant shaper is a special case; it corresponds to $W(s,t) = \sigma(t-s)$, where σ is the shaping curve.

General results of min-plus algebra say that the input-output characterisation of a time-varying shaper is given by

$$R^* = R \cdot \bar{W}$$

where function R is the input, R^* the output and $\bar{W}$ is the sub-additive *closure* of W [76, 77]. Another, equivalent, formulation is:

$$R^*(t) = \inf\{R(t), (R \cdot W)(t), (R \cdot W \cdot W)(t), (R \cdot W \cdot W \cdot W)(t), \dots\} \tag{4.4}$$

Our class of time varying shapers fits in that general framework. It can be easily shown that a time varying leaky bucket shaper corresponds to

$$W(s,t) = \min_{1 \leq j \leq J}\{\int_s^t r_j(u)du \quad + b_j(t)\} \tag{4.5}$$

with $r_j(t)$ and $b_j(t)$ defined as the instantaneous bucket rate and depth at time t, namely $r_j(t) = r_j^i$ and $b_j(t) = b_j^i$ for the index i such that $t_i < t \leq t_{i+1}$.

4.1.3 Chapter breakdown

In this chapter, we want to obtain the input-output characterisation of the time varying leaky bucket shapers. This is equivalent to computing $\bar{W}$, when W is given by Equation (4.5). We could try to obtain $\bar{W}$ from a direct application of Equation (4.4), however this is not a very practical approach. Instead, we obtain $\bar{W}$ from a number of intermediate steps, which provide representations that can easily be applied to a practical computation and give some insights about the system.

To this end, in Section 4.2, we study a shaper system defined by J unchanging leaky buckets, but whose initial conditions (initial bucket levels and initial buffer content) are not zero. We call this model a *leaky bucket shaper with non-zero initial conditions*. We find the input-output characterisation of this model; for this we use min-plus algebra ([69], [78], [7], [77]). Then we apply this iteratively to derive the input characterisation of a time varying leaky bucket shaper (Section 4.3).

4.2 Leaky Bucket Shaper with Non-Zero Initial Conditions Model

In this section we study a leaky-bucket shaper with non-zero initial conditions. This system has the advantage that can easily be studied with network calculus. We derive its input-output characterisation, which can be expressed in terms of the shaping function σ and the initial conditions. We first define the bucket level $q^j(t)$ and the backlog $w(t)$. Then we combine the results and we solve the time varying leaky-bucket shaper model. The deriving input-output characterisation is recursive: at each time $t \in I_i$ we can compute the output $R^*(t)$ with the definition of the system in I_i and the condition at time t_i.

4.2.1 Leaky-Bucket Shaper with Non-Zero Initial Conditions Model

The main result in this section is the characterisation of the leaky-bucket shaper with non-zero initial conditions given in Theorem 1. With non-zero initial conditions

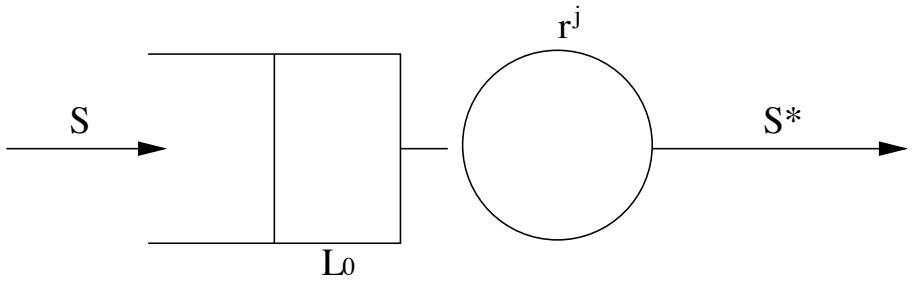


Figure 4.3: Reference Model for a leaky bucket. The traffic S is leaky bucket compliant iff the buckets does not overflow.

we refer to the fact that both the buffer and the buckets present an initial level different from zero. We solve these two cases separately and combine them at the end. The first step is to characterise a shaper system with non-zero initial buffer level. Then we study the case of a shaper system defined by a fixed number J of leaky bucket specifications (r^j, b^j) and that at time $t = 0$ the buckets are non empty. The initial bucket level for the j-th bucket is indicated with q_0^j. We call this system a leaky-bucket shaper system with non-zero initial conditions. When a bit enters the system it is put into the bucket, which is drained at rate r^j, as illustrated in Figure 4.3. A given flow S is conform to a leaky bucket specification when the bucket does not overflow. If we denote with $q(t)$ the bucket level of the bucket at time t, we recall the following characterisation. A flow S is compliant to a leaky bucket with a leaky bucket specification (r, b) when $q(t) \leq b \; \forall t \geq 0$.

We first present a result that is valid for generic shaper systems.

Proposition 1 (Shaper with non-zero initial buffer) *Consider a shaper system with shaping curve σ. Assume that σ is sub-additive and $\sigma(0) = 0$. Assume the initial buffer content of the shaping buffer is given by w_0. The shaper system has no memory of the past. Then the output R^* for a given input R is*

$$R^*(t) = \sigma(t) \wedge \inf_{0 \leq s \leq t} \{(R)(s) + w_0 + \sigma(t-s)\} \quad \forall t \geq 0 \tag{4.6}$$

The condition that σ is sub-additive and $\sigma(0) = 0$ is a technical assumption which is not limiting in practice, since any shaping curve can be replaced by a function satisfying the condition [79, 70]. In particular, the shaping functions associated with leaky buckets do satisfy these assumptions.

Proof:

First we derive the constraints on the output of the shaper. σ is the shaping function thus, for all $t \geq s \geq 0$

$$R^*(t) \leq R^*(s) + \sigma(t-s)$$

and given that the bucket at time zero is not empty, for any $t \geq 0$, we have that

$$R^*(t) \leq R(t) + w_0$$

At time $s = 0$, no data has left the system and this can be expressed with the burst delay function δ_0 defined as follow

$$\delta_0(t) = \begin{cases} 0 & t \leq 0 \\ +\infty & t > 0 \end{cases}$$

Thus, for all $t \geq 0$

$$R^*(t) \leq \delta_0(t)$$

The output is thus constrained by

$$R^* \leq \sigma \otimes R^* \wedge R + w_0 \wedge \delta_0$$

where $\otimes$ is the min-plus convolution operation, defined by $(f \otimes g)(t) = \inf_s f(s) + g(t-s)$. Since the shaper is an optimal shaper, the output is the maximum function satisfying this inequality. We know from min-plus algebra [79, 76] that the solution is given by

$$\begin{aligned} R^* &= \sigma \otimes [(R + w_0) \wedge \delta_0] \\ &= [\sigma \otimes (R + w_0)] \wedge [\sigma \otimes \delta_0] \\ &= [\sigma \otimes (R + w_0)] \wedge \sigma \end{aligned}$$

which after some expansion gives the formula in the proposition. □

In practice this proposition says that, whenever a buffer contains some traffic, this has to be considered as a peak arriving at time $t = 0$. The effect of the peak is the factor $\sigma(t)$ in the representation of the output. An easy derivation is the following corollary.

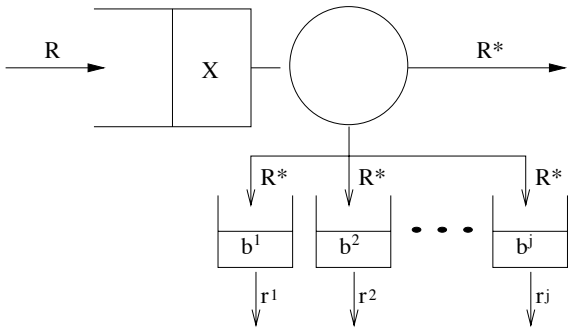


Figure 4.4: Flow R^* compliance is assured at all J leaky buckets.

Corollary 3 (Backlog for a shaper with non-zero initial buffer) *The backlog of a flow S into a buffer drained at rate r with initial level equal to L_0 is given by*

$$L(t) = \max \left[\begin{array}{l} \sup_{0<s\leq t} \{S(t) - S(s) - r \cdot (t-s)\} \\ [S(t) - r \cdot t + L_0] \end{array} \right] \quad t \geq 0 \qquad (4.7)$$

Definition 1 *A given traffic S is compliant to the specification of a leaky-bucket shaper system with non-zero initial conditions if it is compliant to all J leaky buckets.*

From Proposition 1 this results in the following corollary.

Corollary 4 (Compliance to J leaky buckets with non-zero initial bucket levels) *A flow S is compliant to J leaky buckets with leaky bucket specifications (r^j, b^j), $j = 1, 2 \ldots J$ and initial bucket level q_0^j iff*

$$\begin{aligned} S(t) - S(s) &\leq \min_{1\leq j\leq J} [r^j \cdot (t-s) + b^j] \quad \forall 0 < s \leq t \\ S(t) &\leq \min_{1\leq j\leq J} [r^j \cdot t + b^j - q_0^j] \qquad\qquad \forall t \geq 0 \end{aligned}$$

Now we proceed to characterise a leaky-bucket shaper system with non-zero initial bucket levels.

Proposition 2 (Leaky-Bucket Shaper with non-zero initial bucket levels) *Consider a shaper system defined by J leaky buckets (r^j, b^j), with $j = 1, 2 \ldots J$ (leaky-bucket shapers). Assume that the initial bucket level of the j-th bucket is given by*

q_0^j. *The initial level of the shaping buffer is equal to zero. The output* R^* *for a given input* R *is*

$$R^*(t) = \min[\sigma^0(t), (\sigma \otimes R)(t)] \quad \forall t \geq 0 \tag{4.8}$$

where σ *is the shaping function*

$$\sigma(u) = \min_{1 \leq j \leq J} \{\sigma^j(u)\} = \min_{1 \leq j \leq J} \{r^j \cdot u + b^j\}$$

and σ^0 *is defined as*

$$\sigma^0(u) = \min_{1 \leq j \leq J} \{r^j \cdot u + b^j - q_0^j\}$$

Proof:

The output is compliant to all the J leaky buckets. From Corollary 4, this is

$$\begin{aligned} R^*(t) - R^*(s) &\leq \sigma(t-s) \quad \forall 0 < s \leq t \\ R^*(t) &\leq \sigma^0(t) \qquad\qquad\quad \forall t \geq 0 \end{aligned}$$

Considering that $\sigma^0(u) \leq \sigma(u)$ we have that we can extend the validity of the first equation to $s = 0$. Additionally the system is conservative

$$R^*(t) \leq R(t) \quad \forall t \geq 0$$

Thus we have the following constraints:

$$\begin{aligned} R^*(t) &\leq R(t) \qquad\qquad\quad \forall t \geq 0 \\ R^*(t) - R^*(s) &\leq \sigma(t-s) \quad \forall 0 \leq s \leq t \\ R^*(t) &\leq \sigma^0(t) \qquad\qquad\quad \forall t \geq 0 \end{aligned}$$

Given that the system is a shaper, $R^*(\cdot)$ is the maximal solution satisfying those constraints. Using the same min-plus result as in Proposition 1, we obtain:

$$R^*(t) \leq [(\sigma \otimes R^*) \wedge (R \wedge \sigma^0)](t)$$

It derives that R^* is given by

$$\begin{aligned} R^*(t) &= [\overline{\sigma} \otimes (R \wedge \sigma^0)](t) \\ &\text{as } \sigma \text{ is sub-additive } [1] \\ &= \sigma \otimes (R \wedge \sigma^0)(t) \\ &= [\sigma \otimes \sigma^0](t) \wedge [\sigma \otimes R](t) \\ &\text{as } \sigma^0(u) \leq \sigma(u), \text{ this is} \\ &= [\sigma^0 \wedge (\sigma \otimes R)](t) \end{aligned}$$

□

Finally we derive the characterisation of a leaky-bucket shaper that starts with non-zero initial conditions.

Theorem 1 (Leaky-Bucket Shaper with non-zero initial conditions) *Consider a shaper system defined by J leaky buckets (r^j, b^j), with $j = 1, 2 \ldots J$ (leaky-bucket shaper). Assume that the initial buffer level of the shaping buffer is given by w_0 and the initial bucket level of the j-th bucket is given by q_0^j. The output R^* for a given input R is*

$$R^*(t) = \min\{\sigma^0(t), w_0 + \inf_{u>0}\{R(u) + \sigma(t-u)\}\} \quad \forall t \geq 0 \tag{4.9}$$

with

$$\sigma^0(u) = \min_{1 \leq j \leq J}(r^j \cdot u + b^j - q_0^j)$$

Proof:
The proof comes directly from Propositions 1 and 2. □

An intuitive interpretation that generalises Equation (4.9) is to say that any shaper system starting with non-zero initial conditions offers a service that is either the service offered by an ordinary leaky-bucket shaper, taking into account the initial level of the buffer, or, if smaller, a service imposed by the initial conditions, independently from the input. For the class of the leaky-bucket shaper with non-zero initial conditions, we are also able to define the service imposed by the initial conditions as function of the buckets level.

4.2.2 Example

Assume to have a leaky-bucket shaper with non-zero initial conditions defined by 3 leaky buckets

- leaky bucket LB1 with $(r_1 = 2, b_1 = 0)$
- leaky bucket LB2 with $(r_2 = 1, b_2 = 1)$
- leaky bucket LB3 with $(r_3 = \frac{1}{2}, b_3 = 3)$

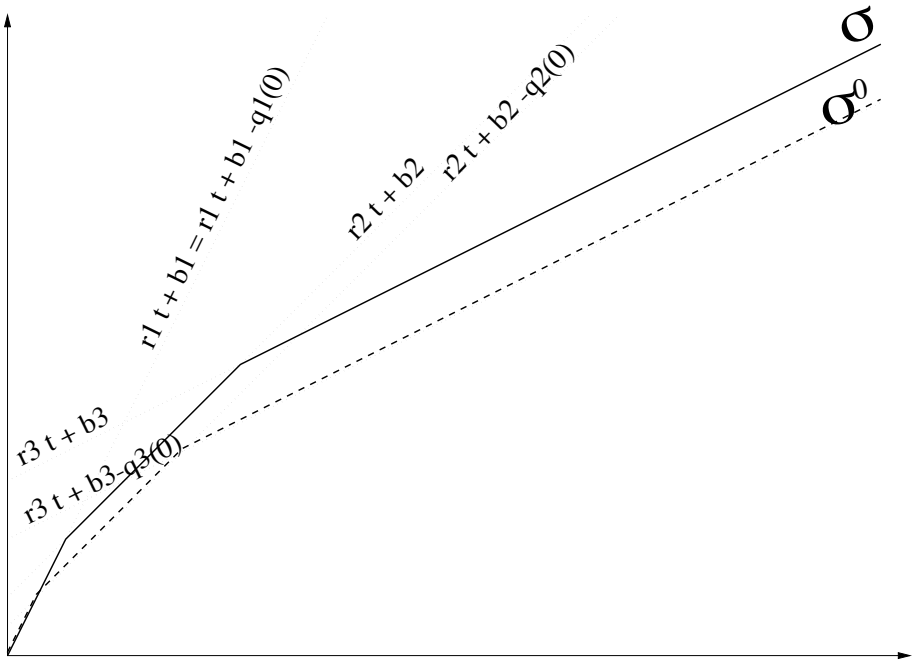


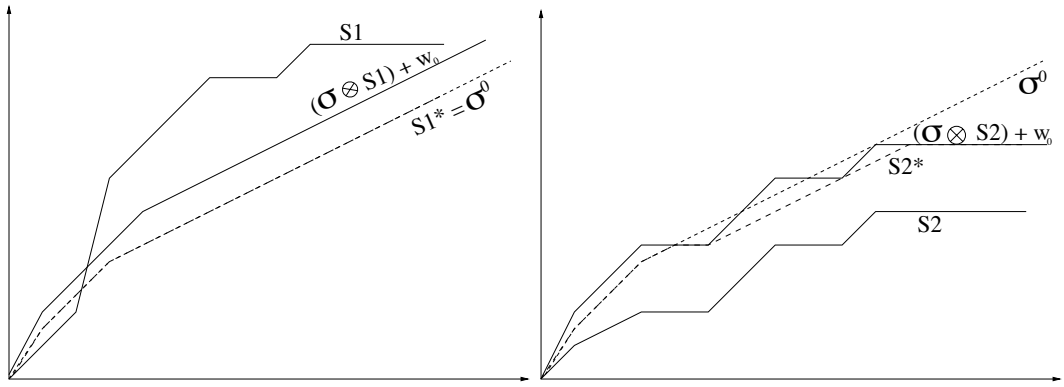
Figure 4.5: Functions σ and σ^0 resulting from LB1, LB2 and LB3 leaky bucket specification and the initial conditions.

and a shaping buffer of capacity $X = 4$. Assume the initial conditions are as follows:

- the level of the bucket LB1 is zero
- the level of the bucket LB2 is equal to $\frac{1}{2}$
- the level of the bucket LB2 is equal to 1
- the initial level of the shaping buffer is $w_0 = 2$

The shaping function σ and the function σ^0 are illustrated in Figure 4.5. Then we analyse the cases of input flows $S1$ and $S2$.

Case 1: In the beginning the amount of traffic issued with $S1$ is not very large and the buckets can handle it without using the buffer anymore, regardless of the initial bucket levels and the initial level of the buffer. Indeed, the quantity of input is smaller than the output, thus the buffer empties. At time $t = 3$ the flow $S1$ arrives with a large amount of traffic. For this reason, after this time, the buckets cannot handle all the traffic and the buffer starts to fill again. At



(a) Output of S1 (b) Output of S2

Figure 4.6: Output $S1^*$ and $S2^*$ of a shaper system with non-zero initial conditions for $S1$ and $S2$.

time $t = 6$ the buffer is full. Every time the output coincides with the function σ^0. This case is illustrated in Figure 4.6(a). With respect to Equation (4.9), $S1^*$ is computed as $\sigma_0(t)$ for any t. This means that the constraint imposed by the initial conditions is always more strong than the action of the shaping function on $S1$.

Case 2: The flow $S2$ presents always a quantity of traffic that can be absorbed by the leaky buckets without using the buffer, even considering the initial conditions. The output coincides with σ^0 in the beginning and with the flow $(S2 \otimes \sigma) + w_0$ for $t > 11$ to the end. For $t \in [4, 11]$, $S2^* = (\sigma_0)(t)$ for $t \geq 5$. The shaping buffer empties at time $t = 4$, varies for $4 \leq t \leq 11$, empties again at $t = 11$ and remains empty after that time. Figure 4.6(b) shows $S2$ and $S2^*$. This is an example of a case where the shaping done by σ is sometimes more relevant than the constraint imposed by the initial conditions.

4.3 Time Varying Leaky-Bucket Shaper Model

In this section, we model the time varying leaky-bucket shaper, we solve this model. This is used in Chapter 5 to deduce the input-output characterisation of

the RVBR service. As introduced in Section 4.1, the time varying leaky-bucket shaper is defined by a fixed number J of leaky buckets and a shaping buffer of fixed capacity X. The parameters of the leaky buckets are not constant, but change at time instants t_i. Consequently the shaping function of this system depends on the time interval and, for each interval I_i, is given by

$$\sigma_i(u) = \min_{0 \le j \le J} (r_i^j \cdot u + b_i^j)$$

As also mentioned in the introduction, the buckets are not reset and we take into account the traffic present at the transient periods. At the time instant t_i, where the leaky bucket parameters are changed, we keep the leaky bucket level $q^j(t_i)$ unchanged.

$q^j(t)$ can be seen as the backlog of a buffer with a variable rate r_i^j, therefore can be computed from Corollary 4.7, in terms of the output $R^*(s)$ for all $t_i \le s \le t$, the rate of the shaper $r_i^j(\cdot)$ in the interval of t and the bucket level $q^j(t_i)$ at the beginning of this interval.

Proposition 3 (Bucket Level) *Consider a time varying leaky-bucket shaper. The bucket level $q^j(t)$ of the j-th bucket is*

$$q^j(t) = \max \left[\begin{array}{l} \sup\limits_{t_i < s \le t} \{R^*(t) - R^*(s) - r_i^j \cdot (t-s)\}, \\ \left[R^*(t) - R^*(t_i) - r_i^j \cdot (t - t_i) + q^j(t_i)\right] \end{array} \right] \quad t \in I_i \tag{4.10}$$

Proof:
This is a direct application of Corollary 3 after a shift in time. Let us introduce the following notation

- for $t \in I_i$ let $\tau = t - t_i$
- for $s \in I_i$, $s \le t$ let $s' = s - t_i$
- $x^*(t - t_i) = R^*(t) - R^*(t_i)$ is the amount of traffic that enters the bucket in $[t_i, t]$.

With this notation we recast $q^j(t)$ as the backlog of a flow x^* into a buffer drained at rate r_i^j with initial level equal to $q^j(t_i)$. Thus, from Corollary 3 we have

$$L(\tau) = \max \left[\begin{array}{l} \sup\limits_{0<s'\leq\tau} \{x^*(\tau) - x^*(s') - r_i^j \cdot (\tau - s')\} \\ \Big[x^*(\tau) - r_i^j \cdot \tau + q^j(t_i)\Big] \end{array} \right] \quad \tau \geq 0$$

Hence, reintroducing the original notation, we obtain

$$q^j(t) = \max \left[\begin{array}{l} \sup\limits_{0<s\leq t} \{R^*(t) - R^*(t_i) + R^*(t_i) - R^*(s) - r_i^j \cdot (t - t_i + t_i - s)\} \\ \Big[R^*(t) - R^*(t_i) - r_i^j \cdot (t - t_i) + q^j(t_i)\Big] \end{array} \right] \quad t \geq 0$$

that gives Equation (4.11). □

We can now characterise a time varying leaky-bucket shaper in the interval I_i by using the input-output characterisation given for a leaky-bucket shaper with non-zero initial conditions. The initial conditions are represented by $q^j(t_i)$ and $w(t_i)$, which are respectively the bucket level and the backlog that are found by the traffic arriving in the interval I_i. Consequently, we also derive the backlog at any time $t \in I_i$ in terms of the input $R(s)$ for all $t_i \leq s \leq t$, the shaping function $\sigma_i(\cdot)$,the bucket level and the backlog at the beginning of this interval I_i, $q^j(t_i)$ and $w(t_i)$, respectively.

Theorem 2 (Time Varying Leaky-Bucket Shapers) *Consider a time varying leaky-bucket shaper with shaping curve σ_i in the interval I_i. The output R^* for a given input R is*

$$R^*(t) = \min \Big[\sigma_i^0(t - t_i) + R^*(t_i), \inf_{t_i<s\leq t}\{\sigma_i(t - s) + R(s)\}\Big] \tag{4.11}$$

where σ_i^0 is defined as

$$\sigma_i^0(u) = \min_{1\leq j\leq J} \Big[r_i^j \cdot u + b_i^j - q^j(t_i)\Big]$$

The backlog at time t is

$$w(t) = \max \left[\begin{array}{l} \sup\limits_{t_i<s\leq t} \{R(t) - R(s) - \sigma_i(t - s)\}, \\ R(t) - R(t_i) - \sigma_i^0(t - t_i) + w(t_i) \end{array} \right] \quad t \in I_i \tag{4.12}$$

Proof:
To demonstrate it we recall the time shift with the notation used in Proposition 3 and we add

- $x(t-t_i) = R(t) - R(t_i)$ that is the amount of traffic that entered in the system in $[t_i, t]$.

With this notation we recast the time varying leaky-bucket shaper as a leaky-bucket shaper with non-zero initial conditions. In this case the initial bucket level of the j-th bucket is equal to $q^j(t_i)$ as given in Equation (4.10) and the buffer level is equal to $w(t_i)$. The input-output characterisation of this system is given by Equation (4.9), thus

$$x^*(\tau) = \sigma_i^0(\tau) \wedge [\sigma_i \otimes x'](\tau)$$

where

$$x'(\tau) = \begin{cases} x(\tau) + w(t_i) & \tau > 0 \\ x(\tau) & \tau \leq 0 \end{cases}$$

Hence, reintroducing the original notation, we obtain

$$R^*(t) - R^*(t_i) = \left[\sigma_i^0(t-t_i) \wedge \inf_{t_i < s \leq t} \{\sigma_i(t-s) + R(s) - R(t_i) + w(t_i)\}\right]$$

thus

$$R^*(t) = \left[[\sigma_i^0(t-t_i) + R^*(t_i)] \wedge [\inf_{t_i < s \leq t} \{\sigma_i(t-s) + R(s) - R(t_i) + w(t_i)\} + R^*(t_i)]\right]$$

that gives Equation (4.11).
Consequently, the backlog at time t results

$$\begin{aligned} w(t) &= R(t) - R^*(t) && t \geq 0 \\ &= R(t) - \min\left[\sigma_i^0(t-t_i) + R^*(t_i), \inf_{t_i < s \leq t} \{\sigma_i(t-s) + R(s)\}\right] \\ &= \max\begin{bmatrix} \sup_{t_i < s \leq t} \{R(t) - R(s) - \sigma \cdot (t-s)\} \\ [R(t) - R^*(t_i) - \sigma^0 \cdot (t-t_i)] \end{bmatrix} && t \geq 0 \end{aligned}$$

that is Equation (4.12). □

In practice, for the class of time varying leaky-bucket shapers, this theorem gives the closure of W discussed in the introduction. Even this result has an intuitive

interpretation that can be generalised for the class of time varying shapers. The output of a time varying shaper in any interval is either driven by σ_0 as combination of the shaping function and the past history, or is computed by taking into account the level of the shaping buffer at the beginning of the interval. This definition is evidently recursive because it depends on the output and on the past history, which are themselves computed with the same formulas. For a discussion on linear time varying shapers see [80].

4.4 Conclusion

Time invariant shapers are not suitable for modelling a situation where traffic changes dynamically and the network resources changes consequently and, thus, for modelling the dynamic VBR problem. This happens because, with time invariant shapers, it is not possible to take into account the traffic that is present in the shaper at the transient moments.

To model such a situation we introduce a class of time varying shapers that we call *time varying leaky-bucket shapers*.

For the class of time varying leaky-bucket shapers, we have found an explicit representation of the output in terms of the input function (input-output characterisation). This is obtained by iterating the input-output characterisation we derive for the class of leaky-bucket shapers with non-zero initial conditions. Before this work, there where no models suitable for the dynamic VBR problem. This innovative result forms the basis of the RVBR service, described in the next chapter.

Chapter 5

Application to the Dynamic VBR Optimisation Problem: Renegotiable VBR Service

This work in this chapter appeared in [67], [68], [81] and [82].

5.1 Introduction

We recall the Renegotiable Variable Bit Rate (RVBR) service, as defined in the Introduction. RVBR is specified as a variable bit rate service whose parameters are changed at periodic renegotiation moments. An example of this service is the Integrated Service of the IETF with the Resource reSerVation Protocol (RSVP), where the negotiated contract may be modified periodically [5].

5.1.1 RVBR characterisation as time varying leaky bucket shaper

As already mentioned, a flow using the RVBR service is constrained by two leaky buckets: one defines the peak rate, the other defines the sustainable rate and the burst tolerance. We consider a basic scenario where a fresh input traffic is shaped in order to satisfy the leaky bucket constraints. Shaping is assumed to be done

using an optimal shaper, with a limited buffer size X [79]. The input traffic may be generated by one source, or it may be an aggregate of sources, in which case the shaper models a service multiplexer. Using VBR in a shaper may be advantageous in all cases where the input traffic is bursty and the network is able to achieve a statistical multiplexing gain on many such input flows [83].

In certain other work, video traffic is carried by networks using Renegotiated CBR service [6], [20]. Contrary to RCBR, with RVBR at any renegotiation time the sources must select three parameters and not just a peak rate. However, the VBR specification better matches the intrinsic characteristics of video sources without requesting high buffering delay [84]. In RCBR service, the selection of parameters, limited to one single parameter, can still lead to very poor resource usage. Assume, for example, that a source not-tolerant to large delays or losses is expected to transmit very bursty traffic. With a simple peak specification, the only option is to request a large peak rate.

Renegotiable VBR services are also studied in [15],[16],[17]; the focus is on describing a given traffic with as few leaky buckets as possible, and thus applies to the optimisation of a network offering the RVBR service. Our approach, in contrast, focuses on the customer side of the RVBR service and provides an analysis of the various tradeoffs that can be made. Our work also differs by the systematic use of network calculus. This results in simple, efficient algorithms that can easily be implemented in real applications.

In our model scenario, the RVBR parameters are renegotiated periodically; at every renegotiation, there is a tradeoff to be made between the various parameters that define the two leaky buckets in the next interval. For example, one may choose a larger burst tolerance and a smaller sustainable rate, or vice versa, depending on the predicted traffic flow and on the cost of the service. This is in contrast with the renegotiated constant bit rate (CBR) service, where only one rate has to be chosen. Our primary goal in this chapter is to analyse this tradeoff. In particular, we propose a method to select, for the next interval, the parameters that minimise a given linear cost function. This is the dynamic VBR problem described in the Introduction. We analyse the RVBR service using the time varying leaky-bucket shapers. We derive

the input-output characterisation of the RVBR service as a special case of the time varying leaky bucket shaper. An RVBR source is a time varying leaky-bucket shaper with two renegotiable leaky buckets ($J = 2$); one with rate r_i and depth b_i and the second with rate p_i and depth always equal to zero, plus a buffer of fixed size X. In real life, examples of this service are traffic shaping done at the source sending over VBR connections as defined in [4] and Internet traffic that takes the form of IntServ specification with RSVP reservation [2], [85]. In the next chapter, we show that the RVBR service indeed can be used to renegotiate resource reservation for Internet traffic with RSVP.

The definition of the RVBR service is straightforward as a special case of time varying leaky-bucket shapers, as defined in Chapter 4, where $J = 2$. Therefore, in the Equations (4.11) and (4.12), σ_i and σ_i^0 are given by

$$\sigma_i(u) = \min(p_i \cdot u + b_i^1, r_i \cdot u + b_i^2) \tag{5.1}$$

$$\sigma_i^0(u) = \min(p_i \cdot u + b_i^1 - q^1(t_i), r_i \cdot u + b_i^2 - q^2(t_i)) \tag{5.2}$$

In conclusion of this section, we recall that the DGCRA is an example of time varying leaky-bucket shapers. We only mention that the output of a node regulated by the DGCRA is equivalent to the output of a time varying leaky-bucket shaper with $J = 2$. The obvious proof is left to the reader.

5.1.2 Chapter breakdown

In next section, we solve the dynamic VBR problem with the RVBR service. For the RVBR service, this is equivalent to the problem of computing the RVBR parameters for the next interval.

We provide two approaches to this problem, when the knowledge of the input traffic is limited to the next interval (*local optimisation problem*, or simply local problem,) and when we dispose of the complete input traffic description (*global optimisation problem*, or global problem). For the local problem we propose two versions: one, when the cost function is represented by a linear cost function and the other, when we compare two solutions in terms of the number of connections

with those parameters that would be accepted on a link with capacity C and physical buffer X. For the two versions of the local problem and for the global problem we provide an algorithm.

In Section 5.3, we compare the two resulting algorithms and show the validity of the local approach by simulation. We simulate the RVBR service versus the renegotiable constant bit rate (RCBR) service and illustrate that the RVBR approach can provide substantial benefits. We also discuss the impact of the size of the renegotiation interval on the efficiency of the RVBR service. Finally, we illustrate the impact of the "no-reset" assumption by analysing on some examples the losses that occur when the source chooses the opposite approach, namely the "reset" approach.

5.2 RVBR Service: the Dynamic VBR Problem

In this section, we analyse the problem of computing leaky bucket parameters for the RVBR service, because we want to use RVBR service for RSVP with CL service scenario. Therefore, we study the case of a source that wants to reserve the resources for the next interval. For the RVBR service, this is equivalent to the problem of computing the RVBR parameters for the next interval.

The parameters optimisation for the RVBR service is not a trivial problem. For example some input traffic could be specified from a large r_i and a small b_i, as well as from a small r_i and a large b_i. This problem can be reduced to an optimisation problem by introducing a cost function that associates a cost to each feasible choice of σ_i. We can approach this optimisation problem in different ways. We can minimise the cost of σ_i at each interval I_i given the status of the system at t_i and the input function $R(t)$ in I_i (local optimisation problem). Alternately, we can minimise the cost of the global sequence of σ_i given the complete input function $R(t)$ (global optimisation problem). The solution of the local optimisation problem is a sequence of local optimal σ_i. The result of the global optimisation problem is the optimal sequence of σ_i. The local optimisation problem and the global optimisation problem require different information. In the first case we only need the information related to the next interval and for special cost functions, we can provide mathematical

formulas to solve it.

The global optimisation problem requires the knowledge of the whole traffic profile and an approach similar to the one used for the local optimisation problem is prohibitive. Hence, we develop a Viterbi-like method to solve it. A solution to the global optimisation problem can be seen as a theoretical limit to the solution to the local one.

For the first version of the local problem we show the application of the local scheme to traffic conforming to the CL service with RSVP reservation protocol.

5.2.1 Local Optimisation Problem

We consider the problem of computing the bucket specifications for the next interval. In particular, referring to the Equations (5.1) and (5.2), b_i^1 is assumed to be fixed and in order to simplify the notation, equal to zero. Therefore we indicate the RVBR parameters at the interval I_i with p_i, r_i and b_i.

General results

From the previous chapter, we know that we can formalise this problem in terms of the system conditions at time t_i and the input in the interval I_i, namely:

- the output $R^*(t_i)$
- the bucket level $q(t_i)$
- the input $R(t_i)$
- the input $R(t)$ for $t \in I_i$

We want to find the shaping function $\sigma_i(u)$. We also assume that $q^j(t) \leq b_i$ for $t \in I_i$ holds and that we guarantee the service, namely $w(t) \leq X$. From Equation (4.12) of Proposition 2, we obtain

$$\begin{aligned} R(t) - R(s) &\leq \sigma_i(t-s) + X && t \in I_i, t_i < s \leq t \\ R(t) - R(t_i) &\leq \sigma_i^0(t-t_i) - w(t_i) + X && t \in I_i \end{aligned}$$

That can be rewritten as

$$\begin{aligned}
&p_i(t-s)+X \geq R(t)-R(s) && t \in I_i, t_i < s \leq t \\
&p_i(t-t_i)+X-w(t_i) \geq R(t)-R(t_i) && t \in I_i \\
&r_i(t-s)+b_i+X \geq R(t)-R(s) && t \in I_i, t_i < s \leq t \\
&r_i(t-t_i)+b_i+X-w(t_i)-q(t_i) \geq R(t)-R(t_i) && t \in I_i
\end{aligned}$$

The equations give a necessary and sufficient condition for a minimum p_i

$$p_i = \max\left(\sup_{t,s\in I_i}\frac{R(t)-R(s)-X}{t-s}, \sup_{t\in I_i}\frac{R(t)-R(t_i)-X+w(t_i)}{t-t_i}\right) \tag{5.3}$$

In analogy to the work in [7] this can be seen as the *deterministic equivalent capacity* of the arrival stream in I_i taking in account the backlog at time t_i.

This means that, given that p_i is computed independently from r_i and b_i, the problem of finding a complete optimal parameter set (p_i, r_i, b_i) for the RVBR service is reduced to the problem of finding the optimal parameters r_i and b_i. This is an important aspect of RVBR service. In fact the deterministic equivalent capacity p_i is also the minimal peak rate selection for RCBR service. Therefore the two parameters r_i and b_i can only lead to better performance.

We assume that r_i and b_i are limited not to exceed some maximum value that is fixed over time (thus valid for all i), that we indicate with r_{max} and b_{max}.

We define with β_i a function that, for each $s \in I = [0, t_{i+1} - t_i]$, computes the maximum amount of traffic sent over the any interval of size s, taking in account the conditions at time t_i.

$$\beta_i(s) = \max\left(\begin{array}{c}\sup\limits_{0\leq v\leq t_{i+1}-t_i-s}\{R(v+s)-R(v)\} \\ R(s+t_i)-R(t_i)+w(t_i)+q(t_i)\end{array}\right)$$

Therefore at each interval I_i, our problem is to minimise a cost function $c(\cdot)$ in the acceptance region defined by

$$\begin{aligned}
&0 \leq r_i \leq r_{max} \\
&0 \leq b_i \leq b_{max} \\
&b_i + r_i \cdot s + X - \beta_i(s) \geq 0 \quad \forall s \in I
\end{aligned} \tag{5.4}$$

where $I = [0, t_{i+1} - t_i]$. One important condition that must be respected [73, 78] is

$$b_{max} \geq \sup_{s \in I}\{\beta_i(s) - r_{max} \cdot s - X\} \tag{5.5}$$

otherwise there are no feasible solutions for r_i and b_i and this must be true at any interval.

As stated in [78] the feasible region can be simplified, in order to facilitate the computation of the optimum. At each interval I_i we apply

$$\begin{aligned} x &= r_i \\ y &= b_i + X \end{aligned}$$

with this change of variable the problem can be rewritten as

$$\begin{aligned} &0 \leq x \leq \min(r_{max}, p_i) \\ &0 \leq y - X \leq b_{max} \\ &y \geq -\check{\beta}_i(x) \end{aligned} \tag{5.6}$$

where $\check{\beta}_i$ is the concave conjugate of β_i

$$\check{\beta}_i(x) = \inf_{s \in I}\{xs - \beta_i(s)\}$$

We note that β_i is sub-additive because can be seen as the minimum arrival curve of the function

$$f(t) = \begin{cases} R(t) & t \in (t_i, t_{i+1}] \\ R(t_i) + w(t_i) + q(t_i) & t = t_i \end{cases} \tag{5.7}$$

Additionally, if $x \geq \max_{s \in I} \frac{\beta_i(s)}{s}$ then $-\check{\beta}_i(x) = \beta_i(0)$ therefore we have that $-\check{\beta}_i$ is wide-sense decreasing.

Now, following the resolution scheme described in [78] we study the optimisation region defined by Equation (5.6) as intersection of two regions R_1 and R_2 respectively given by

$$R_1 = \left\{ \begin{array}{l} 0 \leq x \leq r_{max} \\ 0 \leq y - X \leq b_{max} \end{array} \right\}$$

$$R_2 = \{y \geq -\check{\beta}_i(x)\}$$

The optimum, as illustrated in Figure 5.1, is found at the intersection of regions

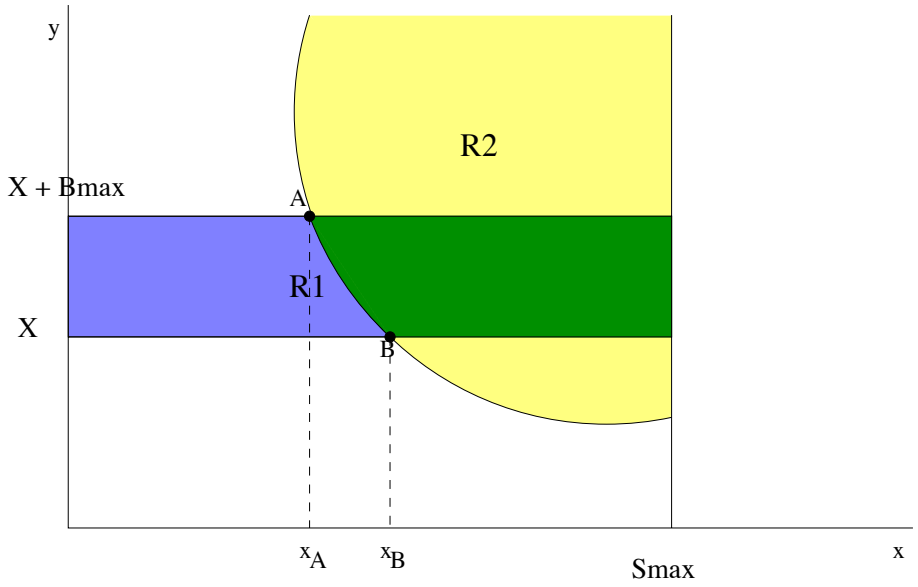


Figure 5.1: Local problem version 1: the optimum is found at the intersection of regions R_1 and R_2

R_1 and R_2. If the cost function is non decreasing in x and y the optimum is on the border of R_2, given that any other point has higher cost. Then, if we define the points A and B that delimit the border of R_2

$$A = \begin{cases} x_A = \sup\limits_{s \in I, s>0} \dfrac{\beta_i(s) - X - b_{max}}{s} \\ y_A = b_{max} + X \end{cases}$$

and

$$B = \begin{cases} x_B = \sup\limits_{s \in I, s>0} \dfrac{\beta_i(s) - X}{s} \\ y_B = X \end{cases}$$

Now to derive an optimal value for r_i and b_i we need to know the optimality criterion used by the network to evaluate the costs of allocating given r_i and b_i. For a generic cost function $c(r_i, b_i)$ we have

$$\text{minimise } c(x, -\check{\beta}_i(x)) \text{ in the region } x_A \leq x \leq \min(x_B, r_{max}, p_i) \tag{5.8}$$

We apply two different cost functions, obtaining two versions of this problem. The first one, where we assume that the cost to the network is given by a linear cost

function, is intended for applications. In Section 6.2.1, we show that the resulting algorithm can be used by an application that uses RSVP as the reservation protocol and specifies the traffic conforming to CL service. The second version is introduced in order to compare the sequence of local optimal solutions that results from the local problem to the one from the global optimisation problem. In fact this is used in Section 5.3.1, where we compare the solutions to the local and the global optimisation problem.

First Version: Linear cost function

In this first version we assume that the choice of the network is driven by a linear cost function. When the cost function is linear the optimisation problem is to minimise $c(r_i, b_i) = u \cdot r_i + b_i$, for fixed values of u. Given that $c(\cdot)$ is linear, as stated above, the optimum is on the border of R_2.

The problem of Equation (5.8) becomes

$$\text{minimise } ux - \check{\beta}_i(x) \text{ in the region } x_A \leq x \leq \min(x_B, r_{max}, p_i) \tag{5.9}$$

In this problem if u is non-positive the minimisation function is wide-sense decreasing and in this case the solution is given by $\min\{x_B, \min(r_{max}, p_i)\}$. If $u > 0$ and the minimum x_0 of the minimisation function is in the interval $[x_A, \min\{x_B, \min(r_{max}, p_i)\}]$ the optimum is for x_0. In particular, if $\beta_i(\cdot)$ is concave, $x_0 = \sup_{s \in I} \frac{\beta_i(s) - \beta_i(u)}{s - u}$. If x_0 is not feasible for the region defined in Equation (5.9) we can have $x_0 \leq x_A$ and in this case the optimum is found at x_A. Otherwise $x_0 \geq \min(x_B, \min(r_{max}, p_i))$ and therefore the optimum is $\min(x_B, \min(r_{max}, p_i))$.

Finally, we can summarise these results in the algorithm *localOptimum1* that finds the optimal solution as described above. The algorithm is given for $\beta_i(\cdot)$ concave. When this does not hold it is substituted by $\beta_i'(\cdot)$, which it is a concave arrival curve of $f(t)$ as given in Equation (5.7).

As mentioned above, p_i is independent and can be computed as the deterministic equivalent capacity of $R(t)$ in this interval.

Algorithm 5.1 localOptimum1$(X, \{R(t)\}_{t \in I_i}, b_{max}, r_{max}, u, w(t_i), q(t_i), t_{i+1})$

if $b_{max} < \sup_{s \in I}\{\beta_i(s) - r_{max} \cdot s - X\}$ **then** there is no feasible solution;

else {

$p_i = \max\left(\sup_{t,s\in I_i} \frac{R(t)-R(s)-X}{t-s}, \sup_{s\in I_i} \frac{R(t_i)-R(s)-X+w(t_i)}{t_i-s}\right)$

if $u \leq 0$ **then** {

$x_0 = min(r_{max}, p_i)$;

}

else {

$x_0 = \sup_{s\in I} \frac{\beta_i(s)-\beta_i(u)}{s-u}$;

$x_A = \sup_{s\in I, s>0} \frac{\beta_i(s)-X-b_{max}}{s}$;

$x_B = \sup_{s\in I, s>0} \frac{\beta_i(s)-X}{s}$;

if $(x_0 \geq \min(x_B, r_{max}, p_i))$ **then** $x_0 = \min(x_B, r_{max}, p_i)$;

else if $(x_0 \leq x_A)$ **then** $x_0 = x_A$;

}

$r_i = x_0$;

$b_i = \sup_{s\in I}\{\beta_i(s) - X - s \cdot x_0\}$;

}

Second Version: maximum number of accepted connections

In this second version we take a different approach. In fact here, for any solution (p_i, b_i, r_i) we compute the number N_i of homogeneous connections, specified by (p_i, b_i, r_i), acceptable by a link with fixed capacity C and buffer with fixed size B . The cost of each solution is represented by the reciproc of N_i, therefore the minimum is obtained for the maximum number of connection accepted[1].

As illustrated in Figure 5.2, the number of connection accepted has to be such that

$$N_i \cdot (\min(p_i t, r_i t + b_i)) \leq C \cdot t + B \quad \forall t \geq 0$$

and this is equivalent to

$$N_i \leq \frac{C}{p_i}$$
$$N_i \leq \frac{C}{r_i}$$

[1]In reality N_i should be an integer, but given that we use it only for computing the cost of a traffic descriptor, we accept that N_i takes any positive real value

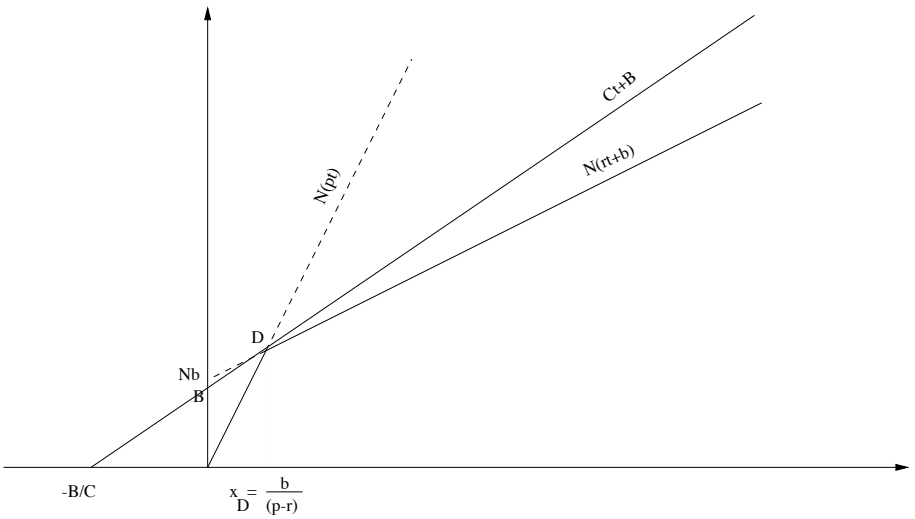


Figure 5.2: Local problem version 2: the optimum is found either at the intersection between $y1 = C \cdot t + B$ and $y2 = r \cdot t + b$ or for $\frac{B}{b}$

Given that the second one is more restrictive, we simply have that

$$N_i \leq \frac{C}{r_i} \tag{5.10}$$

Moreover, at limit, $y1 = C \cdot t + B$ intersects $y2 = N(r \cdot t + b)$ at the point

$$D = \begin{cases} x_D = \frac{b_i}{(p_i - r_i)} \\ y_D = \frac{N_i p_i b_i}{(p_i - r_i)} \end{cases}$$

Therefore we derive that

$$\frac{N_i p_i b_i}{(p_i - r_i)} \leq C \cdot \frac{b_i}{(p_i - r_i)} + B$$

That gives

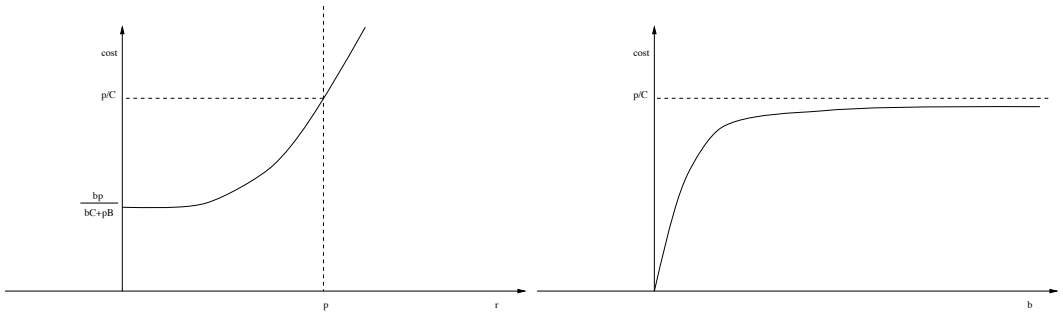
$$N_i = \max\left(\frac{B \cdot (p_i - r_i) + b_i C}{b_i p_i}, \frac{C}{r_i}\right) \tag{5.11}$$

and thus, from Equation (5.11) the cost for the solution (p_i, b_i, r_i) is

$$\min\left(\frac{b_i p_i}{B \cdot (p_i - r_i) + b_i C}, \frac{r_i}{C}\right) \tag{5.12}$$

This is equivalent to say that we associate to each accepted connection with traffic descriptor equal to (p_i, b_i, r_i) a leaky bucket specification with rate equal to

$$\hat{r} = \frac{C b_i p_i}{B \cdot (p_i - r_i) + b_i C} \tag{5.13}$$



(a) Evolution of the cost function for fixed y

(b) Evolution of the cost function for fixed x

Figure 5.3:

and a bucket size equal to

$$\hat{b} = b_i \cdot \frac{(p_i - \hat{r})}{(p_i - r_i)} \tag{5.14}$$

We observe that this function is increasing in both x and y, as illustrated by Figures 5.3(a) and (b).Thus the optimum is on the border of R_2. The problem of Equation (5.8) becomes

$$\text{minimise } \min\left(\frac{(\check{\beta}_i(x) - X)p_i}{B \cdot (p_i - x) + C \cdot (\check{\beta}_i(x) - X)}, \frac{x}{C}\right) \text{ in the region } x_A \leq x \leq \min(x_B, r_{max}, p_i) \tag{5.15}$$

When the system evolves we have to take in account the evolution of the physical buffer of capacity B. At at time t_i it contains some traffic from the past. Hence we do not dispose of the complete capacity B, but only of part of it, indicated by B_i. B_i is the difference between the buffer size and the backlog at that time

$$B_i = B - W(t_i)$$

where $W(t_i)$, indicated with N_{i-1} the number of connections accepted at the previous interval, is computed with a modified version of Equations (4.12) as follows

$$W(t_i) = \max\left[\begin{array}{l} \sup\limits_{t_{i-1} \leq s \leq t_i} \{N_{i-1} \cdot (R^*(t_i) - R^*(s)) - C \cdot (t_i - s)\}, \\ N_{i-1} \cdot (R^*(t_i) - R^*(t_{i-1}))C \cdot (t_i - t_{i-1}) + W(t_{i-1}) \end{array}\right] \tag{5.16}$$

Algorithm 5.2 localOptimum2$(X, \{R(t)\}_{t\in I_i}, b_{max}, r_{max}, u, w(t_i), q(t_i), t_{i+1}, C, B, W(t_i))$

if $b_{max} < \sup_{s\in I}\{\beta_i(s) - r_{max}\cdot s - X\}$ **then** there is no feasible solution;
else {
  $p_i = \max\left(\sup_{t,s\in I_i}\frac{R(t)-R(s)-X}{t-s}, \sup_{s\in I_i}\frac{R(t_i)-R(s)-X+w(t_i)}{t_i-s}\right)$
  if $u \le 0$ **then** {
    $x_0 = min(r_{max}, p_i)$;
  }
  else {
    x_0 that minimise min $\left(\frac{(\check{\beta}_i(x)-X)p_i}{B\cdot(p_i-x)+C\cdot(\check{\beta}_i(x)-X)}, \frac{x}{C}\right)$;
    $x_A = \sup_{s\in I, s>0}\frac{\beta_i(s)-X-b_{max}}{s}$;
    $x_B = \sup_{s\in I, s>0}\frac{\beta_i(s)-X}{s}$;
    if $(x_0 \ge \min(x_B, r_{max}, p_i))$ **then** $x_0 = \min(x_B, r_{max}, p_i)$;
    else if $(x_0 \le x_A)$ **then** $x_0 = x_A$;
  }
  $r_i = x_0$;
  $b_i = \sup_{s\in I}\{\beta_i(s) - X - s\cdot x_0\}$;
}

We are aware that this second algorithm does not consider any statistical multiplexing. However, it is only used to compare the sequence of local optima with the sequence resulting as solution to the global optimisation problem, as defined in Section 5.2.2. The comparison results are give in Section 5.3.1.

5.2.2 Global Optimisation Problem

In Section 6.2.1 we illustrate how to build a complete solution with the local algorithm as a sequence of local optima. This solution is not necessarily the optimal sequence. In fact a sequence of local optimal solutions is very likely to cost more than an optimal sequence.

Assume that at a certain interval we find a solution that optimises the network resources but at the cost of a very large buffer occupancy. At a later interval, because of the lack of buffer space, there might not be feasible solutions (see Equation (5.5)),

or the optimal solution might have a very high cost. It is clear that an algorithm for finding the optimal sequence is too expensive in terms of time and memory occupation and we argue that such an algorithm can be useful only for evaluating other schemes. In order to have a theoretical optimum to compare with the solution of the local scheme, we studied an algorithm based on a Viterbi-like algorithm [86, 87]. A similar study is presented in [6] for RCBR service.

We keep valid all previous assumptions and the RVBR service is still renegotiated at every interval $I_i = (t_i, t_{i+1}]$, $i \in \mathbb{N}$. The fixed set $\mathcal{S} = \{s_l;\ l = 1, 2, \ldots K^3\}$ contains the possible RVBR service parameter sets we can select at each interval. A RVBR service parameter set s_l is described as

$$s_l = \begin{bmatrix} p_h \\ r_k \\ b_j \end{bmatrix} \quad h, k, j \in [1, K]$$

We note that the peak p_h can reasonably assume values that are different from the deterministic equivalent capacity at Equation (5.3), because a larger peak in the interval I_i can lead to a minor utilization of the buffer and this could permit the reduction of the cost at the next intervals. The linear cost function introduced for the local problem in Section 5.2.1, $c(p_h, r_k, b_j) = u \cdot r_k + b_j$, is based on the fact that there is only one solution for the peak, thus it is not usable. It is evident that it is not possible to find any function to give a global cost comparison. Therefore we adopt a "call admission control" approach. We compare the two algorithms in terms of the number of connections that would be accepted on a link with capacity C and physical buffer of size B. Given a parameters selection s_l we indicate with N_i^l the number of homogeneous connections with parameters s_l accepted at interval I_i in *localOptimum2* as defined in Section 5.2.1.

In the Viterbi-like algorithm a *node* of the trellis represent a state encountered by the system. Here we need to distinguish two states whenever they are reached with different cost or usage of network resources. The trellis diagram is also spread over time thus a node is represented by a 5-tuple $n = (i, w(t_i), q(t_i), N_{tot}, W(t_i))$. i indicates that this state is reached at time t_i; $w(t_i)$ and $q^j(t_i)$ are computed with Equations (4.12) and (4.10) respectively; $W(t_i)$ is computed with Equation (5.16)

and N_{tot} indicates the sum of the number of accepted connections as computed in Equation (5.11) for each state traversed to reach n

$$N_{tot} = \sum_{j=0}^{i} N_j$$

A transition from a node $n = (i, w_n(t_i), q_n(t_i), (N_{tot})_n, W_n(t_i))$ to node $m = (i + 1, w_m(t_{i+1}), q_m(t_{i+1}), (N_{tot})_m, W_m(t_{i+1}))$ happens whenever there exists an element $s_l = \{p_h, r_k, b_j\}$ in $\mathcal{S}$ such that m derives from n by serving the traffic over the interval I_i with a RVBR service described by s_l

$$w_m(t_{i+1}) = \max \left(\begin{array}{l} \sup\limits_{t_i \le s \le t_{i+1}} [R(t_{i+1}) - R(s) - \sigma_i(t_{i+1} - s)], \\ R(t_{i+1}) - R(t_i) - \sigma_i^0(t_{i+1} - t_i) + w_n(t_i) \end{array} \right)$$

with

$$\sigma_i(u) = \min\{p_h \cdot u, r_k \cdot u + b_j\}$$

and

$$\sigma_i^0(u) = \min\{p_h \cdot (u), r_k \cdot u + b_j - q_n(t_i)\}$$

and

$$q_m(t_{i+1}) = \max \left(\begin{array}{l} \sup\limits_{t_i < s \le t_{i+1}} \{R^*(t) - R^*(s) - r_k \cdot (t_{i+1} - s)\}, \\ R^*(t_{i+1}) - R^*(t_i) - r_k \cdot (t_{i+1} - t_i) + q_n(t_i) \end{array} \right)$$

and

$$W_m(t_{i+1}) = \max \left(\begin{array}{l} \sup\limits_{t_i \le s \le t_{i+1}} \{N_i \cdot (R^*(t_{i+1}) - R^*(s)) - C \cdot (t_{i+1} - s)\}, \\ N_i \cdot (R^*(t_{i+1}) - R^*(t_i)) - C \cdot (t_{i+1} + W_n(t_i) \end{array} \right)$$

In this case we have an *edge* from n to m and the associated number of accepted connections is

$$N_i^l = \max \left(\frac{B_i \cdot (p_h - r_k) + b_j C}{b_j p_h}, \frac{C}{r_k} \right)$$

Note that we do not keep track of the used traffic parameter s sets, because we use this algorithm only for comparison purpose. That way the format of the node do not need to include the traffic parameter selection information.

We call *path* the sequence of edges from the initial node $n_0 = (0, 0, 0, 0, 0)$ to a node n. The goal is to find the path able to accept the largest number of connections

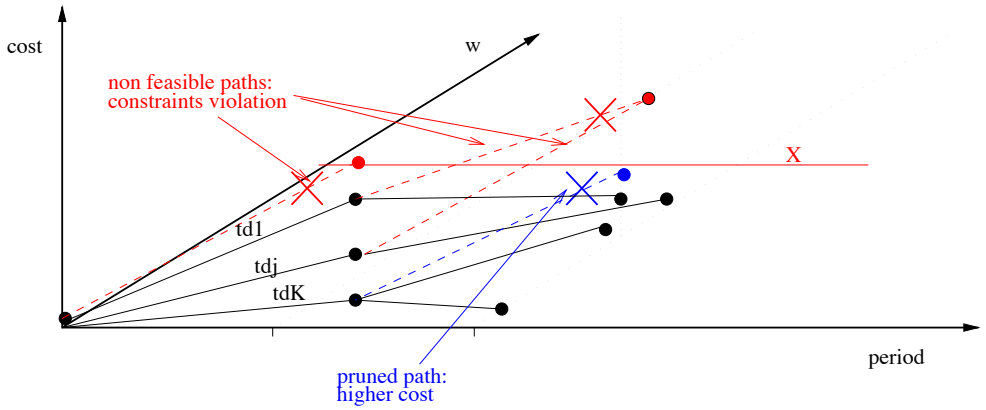


Figure 5.4: An example of the trellis: some path is not added to the trellis and some other is eliminated.

over the time among all paths from n_0 to a final node, i.e. a node that represents a status of the system when the input traffic stops.

We define a node $n = (i, w(t_i), q(t_i), N_{tot}, W(t_i))$ to be *feasible* if the constraints are respected for all $t \leq t_i$

$$0 \leq w(t) \leq X$$
$$0 \leq q(t) \leq b_j$$

An edge from a feasible node n to node m is feasible if m is feasible.

We also define a node $n = (i, w_n(t_i), q_n(t_i), (N_{tot})_n, W_n(t_i))$ to be *non optimal* when there exists a node $m = (i, w_m(t_i), q_m(t_i), (N_{tot})_m, W_m(t_i))$ such that

$$w_n(t_i) \geq w_m(t_i)$$
$$q_n(t_i) \geq q_m(t_i)$$
$$(N_{tot})_n \leq (N_{tot})_m$$
$$W_n(t_i) \geq W_m(t_i)$$

All the paths to a non optimal node n are non optimal.

We limit the exponential growing of the trellis first by creating only feasible edges and nodes and then by pruning the paths and the nodes that are not optimal. Consequently at each interval some path is not added to the trellis (because it is not feasible) and some other is eliminated (because it is not optimal), as represented in Figure 5.4. At the end we select one of the paths that reaches one of the nodes with the greatest number of accepted connections N_{tot}.

This can be resumed in the following algorithm

Algorithm 5.3 globalOptimum$(X, \{R(t)\}, \{s_l\}, C, B)$

$n_0 = (0, 0, 0, 0, 0)$

for $(i = 1; i \leq I; i++)$ **then** {

for $(l = 1; l \leq K^3; l++)$ **then** {

create all the feasible edges corresponding to s_l from nodes at i to nodes at $i + 1$;

prune all non optimal nodes and edges;

}

}

select one path ending in a node with the greatest N_{tot};

where I denotes the index of the last renegotiation time t_I.

5.3 Evaluation of the RVBR Service

In this section we apply the previous algorithms to discuss a number of issues related to the RVBR service. The simulation scenario corresponds to the real case of a MPEG2 video-trace transmitted over a network with RSVP as reservation protocol. This scenario is described and analysed more in detail in next chapter. Here we are not interested to implementation details that are, therefore, omitted.

The trace is a 4000 frame-long sequence, composed of several video scenes that differ in terms of spatial and temporal complexities. The traffic generated by the video is transported by a trunk regulated by a RVBR service (p, r, b) with shaping buffer X. The video, with a total size of 550 Mbits, is transmitted in 163 seconds (25 frames pro second), without any scheduling.

5.3.1 Comparison of the Local and Global Algorithms

Here we simulate the *localOptimum2* algorithm as defined in Section 5.2.1 against the globalOptimum algorithm proposed in Section 5.2.2 to give a measure of the optimality of the former in terms of cost. We ignore the renegotiation cost, because this cost is the same in both the local and the global case. The algorithm proposed for the global problem, the Viterbi-like algorithm, works with a discrete set of values

whereas the algorithm proposed in Section 5.2.1 for the local problem can result in any value. For reason of comparison, we forced the local algorithm (*localOptimum2*) to work with the same discrete set of values. The two resulting complete sequences are comparable because the two algorithms select the optimum as the set $s_l \in S$ that permits the acceptance of the largest number of connections on a link of capacity C with associated buffer size B.

In Figures 5.5-5.7 we illustrate the behaviour of the two schemes in terms of the number of connection accepted for the optimal solution. Again for reason of space we only show the following scenarios:

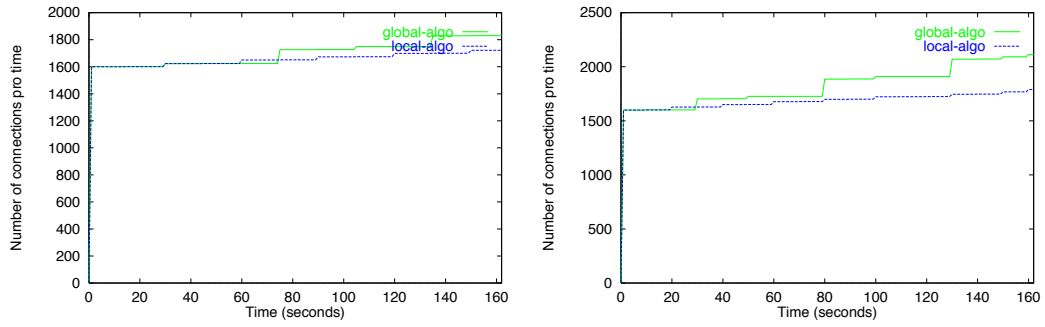
Scenario 1: $X = 12$ Mbits, $B = 40$ Mbits, $C = 20$ Mbps

Scenario 2: $X = 5$ Mbps, $B = 20$ Mbits, $C = 10$ Mbps

Scenario 3: $X = 0.6$ Mbits, $B = 10$ Mbits, $C = 60$ Mbps

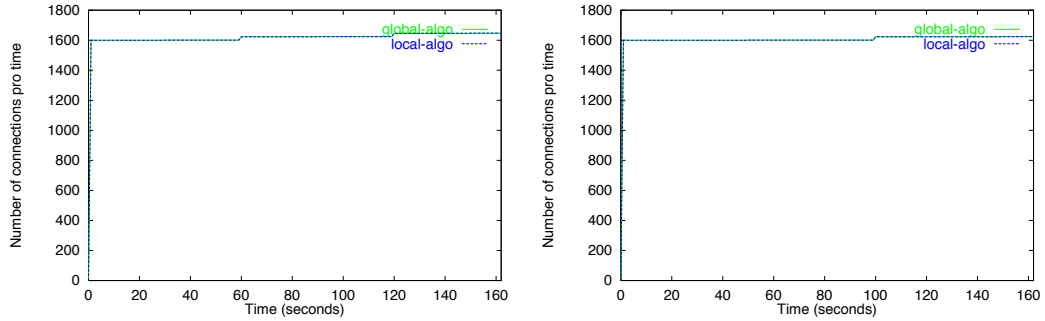
In the local scheme, an incorrect renegotiation affects the future. This is even more valid in the example illustrated here, because the input traffic is significantly bursty for large periods. Despite this, our algorithm does not deviate significantly from the theoretically optimal one. We can see that even for high numbers of renegotiations (i.e. short renegotiation periods: sub-figures (a) and (b)) it presents a behaviour not too far from the optimum, while for larger renegotiation periods the two solutions are frequently the same. It is important to notice that there is no relation between the behaviour experienced during two different renegotiation periods. This is evident when we analyse the renegotiation at 10 and 15 seconds, and it is due to the fact that the solution is optimal inside the interval.

In terms of buffer usage we observe even a better result for the local algorithm. The average occupation percentage we obtain when using the local algorithm is always very close to that of the optimal algorithm for all the renegotiation periods we analysed.



(a) renegotiation every 10 seconds

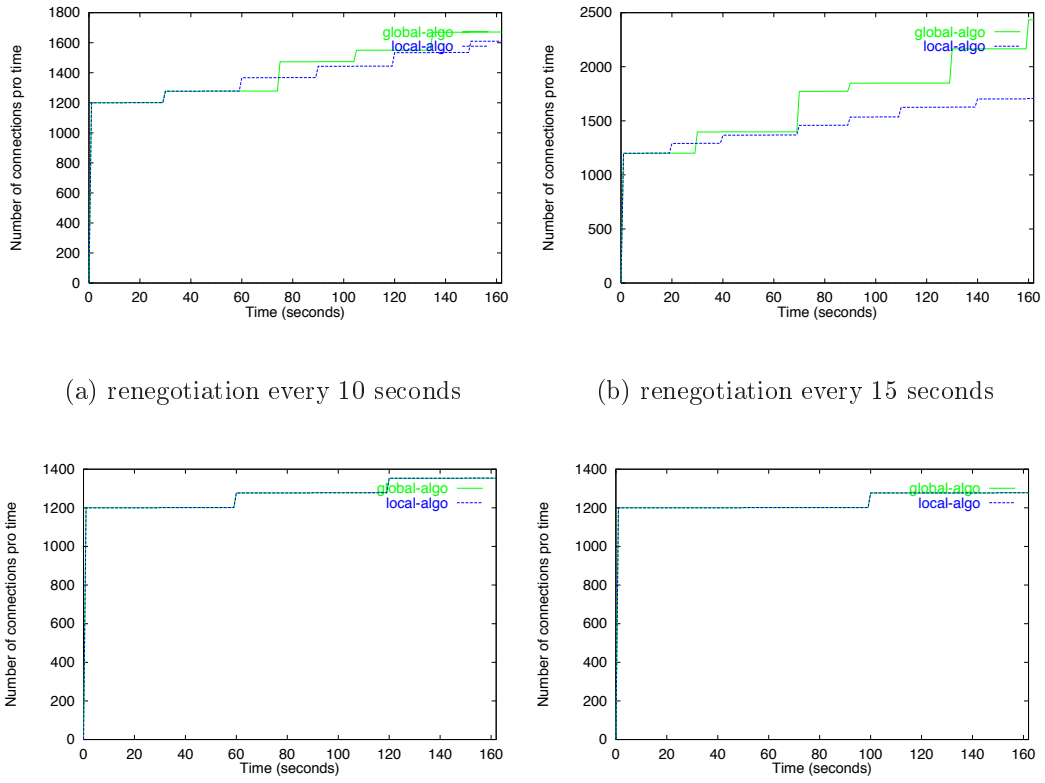
(b) renegotiation every 15 seconds



(c) renegotiation every 30 seconds

(d) renegotiation every 50 seconds

Figure 5.5: Scenario1: comparison of the number of connection accepted by the local scheme (solid curve) and the global Viterbi-based (dashed curve) scheme for different renegotiation period.



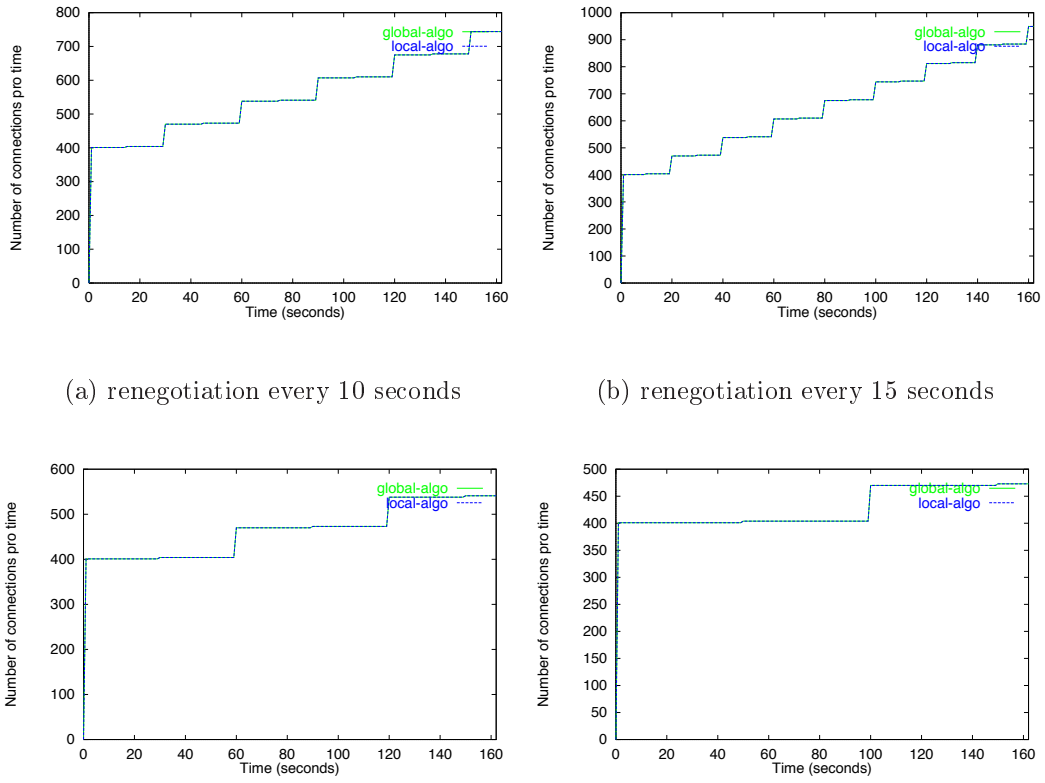
(a) renegotiation every 10 seconds

(b) renegotiation every 15 seconds

(c) renegotiation every 30 seconds

(d) renegotiation every 50 seconds

Figure 5.6: Scenario2: comparison of the number of connection accepted by the local scheme (solid curve) and the global Viterbi-based (dashed curve) scheme for different renegotiation period.



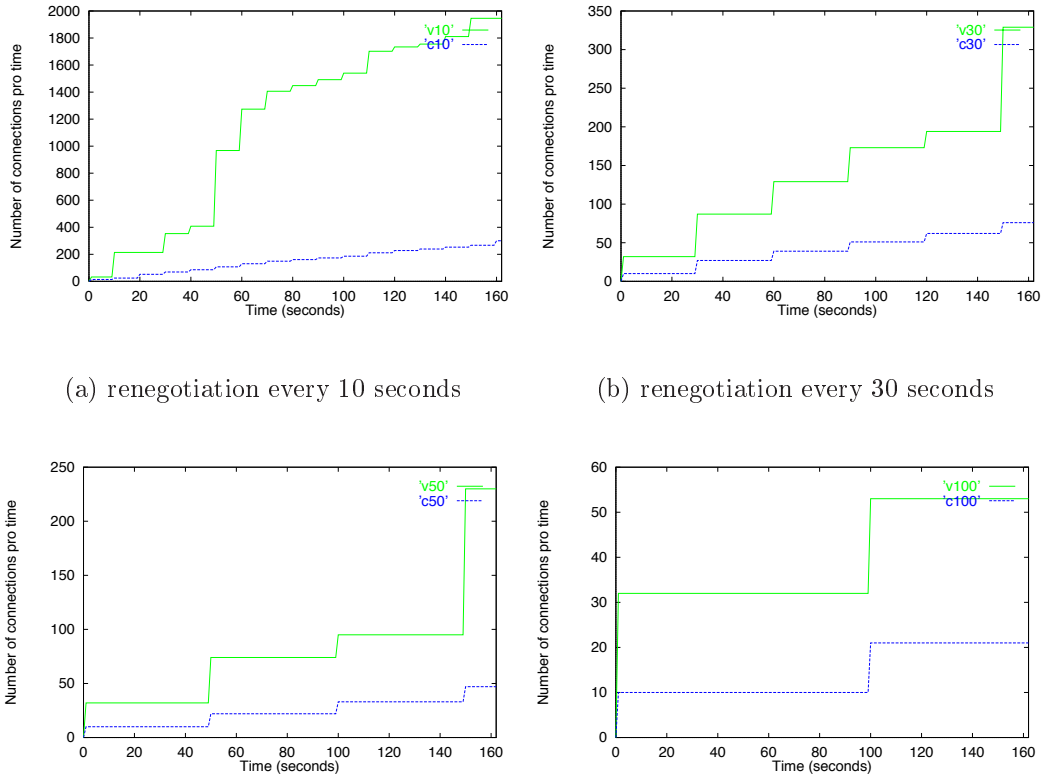
(a) renegotiation every 10 seconds

(b) renegotiation every 15 seconds

(c) renegotiation every 30 seconds

(d) renegotiation every 50 seconds

Figure 5.7: Scenario3: comparison of the number of connection accepted by the local scheme (solid curve) and the global Viterbi-based (dashed curve) scheme for different renegotiation period.



(a) renegotiation every 10 seconds

(b) renegotiation every 30 seconds

(c) renegotiation every 50 seconds

(d) renegotiation every 100 seconds

Figure 5.8: Number of connections accepted by a link of capacity $C = 500$ Mbits and physical buffer size $B = 60$ Mbits for the RVBR service (solid curve) and the RCBR one (dashed curve) at different renegotiation period.

5.3.2 Renegotiable VBR Service versus Renegotiable CBR Service

In this section we simulate the local scheme based on renegotiable VBR service against a renegotiable CBR service. In the CBR case we can renegotiate only the peak rate, i.e. a constant rate, for each interval. We simulate the two services for different renegotiation periods and we show the benefits of the VBR approach in terms of connection accepted as described in Equation (5.11). Again, we ignore the renegotiation cost, because it is equal for both services. Looking at Figure 5.8, we observe that the reduction of the number of connection accepted for the RCBR service is significant. This can be easily explained by the difficulty of shaping bursty

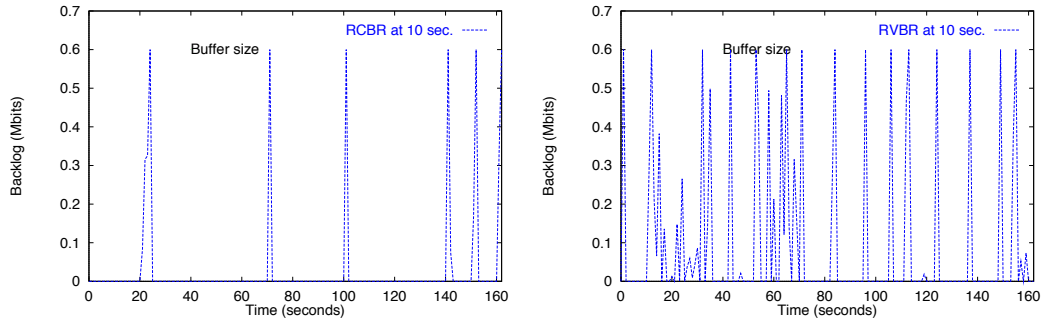


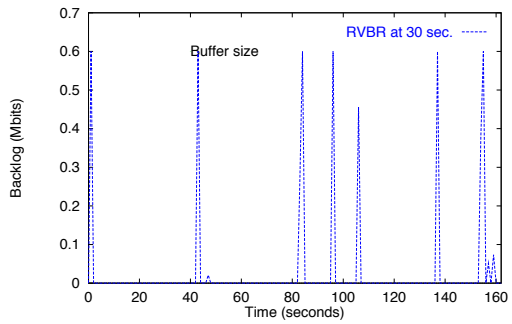
Figure 5.9: Buffer utilisation for a quite small renegotiation period of 10 seconds: the RVBR service approach (on the right) is clearly better than the CBR approach (on the left).

traffic with a simple rate specification [84]. Obviously this fact is more evident when we do not renegotiate frequently. Therefore, the larger difference is present in the larger renegotiation period cases. For the same reason, in Figure 5.9 and 5.10 we see that the the buffer in the CBR case is really under-used. Thus, as expected, there are obvious benefits in using the RVBR service instead of the RCBR one.

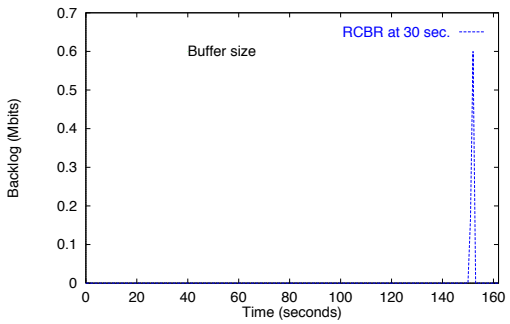
5.3.3 Discussion on the Impact of the Renegotiation Interval Size

One factor we varied, in order to analyse different results, is the renegotiation period. The renegotiation period can range from instantaneous renegotiation (1 second in our case) to no renegotiation. The analysis of some intermediate points permits the study of the evolution in terms of this factor. We use in this analysis the local problem algorithm, with the MPEG2 input traffic used in the previous sections. Figure 5.11 illustrates an example of the different costs we obtained with different renegotiation periods.

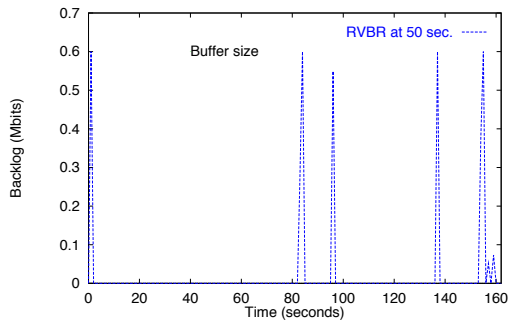
As expected, in general it happens that the larger the renegotiation period is, the higher the cost of the traffic specification. With a local approach this is not always true. In fact the local optimum of a larger period is less expansive than the sum of the cost for a smaller renegotiation period on the same interval. In fact the optimum is local inside the interval. This effect is better illustrated in Figure 5.11(b). Here



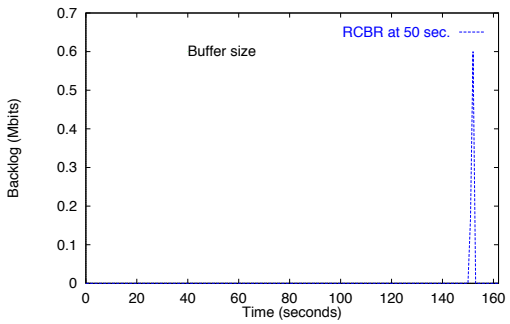
(a) VBR renegotiation every 30 seconds



(b) CBR renegotiation every 30 seconds

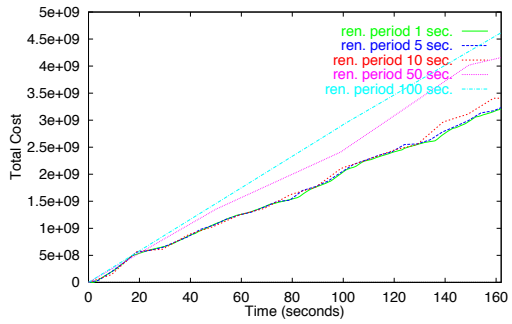


(c) VBR renegotiation every 50 seconds

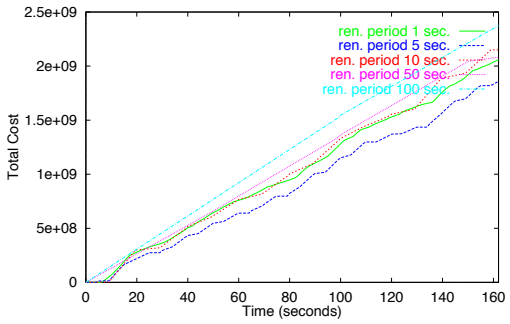


(d) CBR renegotiation every 50 seconds

Figure 5.10: Buffer utilisation for more large renegotiation periods: the RCBR service (on left) is unable to use the buffer. The peak selected is to high and in those cases the buffer results always empty.



(a)



(b)

Figure 5.11: Evolution of the cost versus renegotiation period.

the curve representing the cost of reallocating every 10 seconds is often above most of the other curves.

The curves presented until now do not includes any cost for the renegotiation in terms of signalling, etc. When we consider, as part of the problem, having a renegotiation cost, we find a tradeoff between the advantage of the renegotiation and its cost. This issue cannot be universally solved because it depends on the input traffic.

Here we discuss certain aspects of this problem. If we assume a fixed renegotiation cost γ and indicate with $\hat{p}^j$, $\hat{r}^j$ and $\hat{b}^j$ the average values for the peak, the rate and the bucket renegotiating j times, we can represent the optimal renegotiation period with

$$T_n = \left\lceil \frac{T}{n} \right\rceil$$

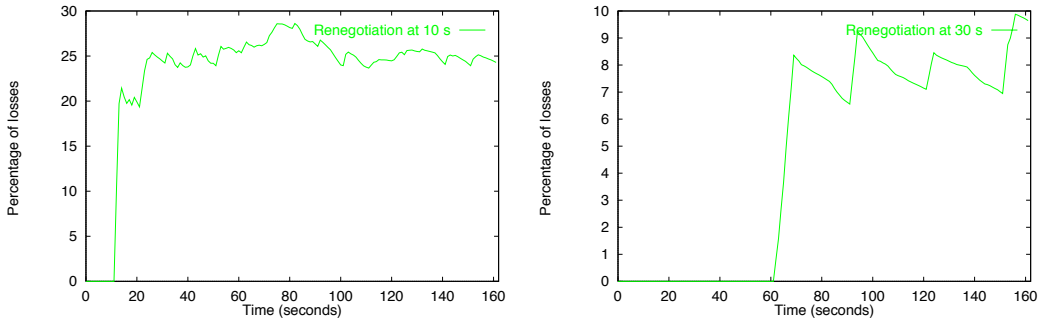
where T represents the lifetime of the input traffic and

$$n = \{m \in \mathbb{N} : [m \cdot (\gamma + u \cdot \hat{r}^m + \hat{b}^m)] \text{ is minimum}\}$$

However, given that r_i and b_i are computed on the basis of the traffic expected in the next interval it is evident that n depends not only on γ, but also on the input traffic profile. Moreover, the problem of defining the optimum fixed renegotiation period requires the knowledge of the complete reallocation sequence for each m. This is in contrast with the local approach we propose.

By applying the Viterbi-like algorithm presented in Section 5.2.2 with an instant renegotiation period and with additional cost for renegotiating (as it is done, for example, in [6]), it is possible to derive an optimal frequency of the renegotiation. In this case the renegotiation scheme and the method for defining when to renegotiate are combined and based on the complete knowledge of the input traffic.

If the traffic is known in advance, one approach could be to change the renegotiation period based on the traffic profile. For instance, if the prediction for the next interval of 30 seconds gives a very bursty profile, we could consider to renegotiate the resources more often inside that interval. One factor that can be used to this purpose is the variance of the expected traffic: in general if it is large, we can also foresee a non optimal usage of the resources. This approach, contrary to the one



(a) renegotiation every 10 seconds (b) renegotiation every 30 seconds

Figure 5.12: Percentage of losses in the reset approach

based on the Viterbi-like algorithm is still based on a local approach. However, in both cases, the result is a variable renegotiation period.

5.3.4 "Reset" versus "No Reset" Approach

As described previously, we choose to use the "no reset" approach instead of the simpler "reset" one. In this section we study the losses occurring in the "reset" approach. We want to show that this approach is not valid for traffic with strict loss constraint.

It is trivial that, in terms of costs, the "reset" approach is better because it always restarts from a zero initial condition and considers the lost traffic as sent.

First we point out that the network must use the "no reset" approach because it must ensure to any input traffic, exactly the same service when traffic specification is always renegotiated with the same $\sigma_i = \sigma$ and when the traffic specification is equal to σ and is not renegotiated. This is not possible if the network resets the buckets level at every renegotiation time.

In principle, at the source both approaches are valid. However, when we reset the buckets we must accept to experience some loss due to the fact that the network does not apply any reset. This means that the upper bound to those losses is given by the maximum size of the bucket (b_{max}) times the number of times we apply the

renegotiation. Therefore an upper bound for the percentage of losses is given by

$$\min_i \frac{b_{max} \cdot i}{R(t_i)} \tag{5.17}$$

It is already clear that this upper bound can be not acceptable for many types of traffic. In practice this limit is easily reached, unless b_{max} is very small. Only in this case, where we have that $\frac{b_{max}}{R(t_i)}$ is close to zero for any value of i, the impact of the reset does not affect the system behaviour. Evidently we can assume that this condition should not occur, because it corresponds to a bad network planning.

To evaluate how close we get to this upper bound we simulate the two approaches in the same scenario described in Section 6.2.1, where we use IntServ services with RSVP reservation protocol.

We use again the same MPEG2 4000 frame-long sequence as input. We measure the percentage of losses, that obviously depends on b_{max}. For the renegotiation at every 30 seconds, we experience a percentage of losses from 5%, for b_{max} very small, up to 60%. Obviously, for a fixed b_{max}, the percentage of losses grows with the decrease of the renegotiation period. For very small renegotiation periods can be enormous.

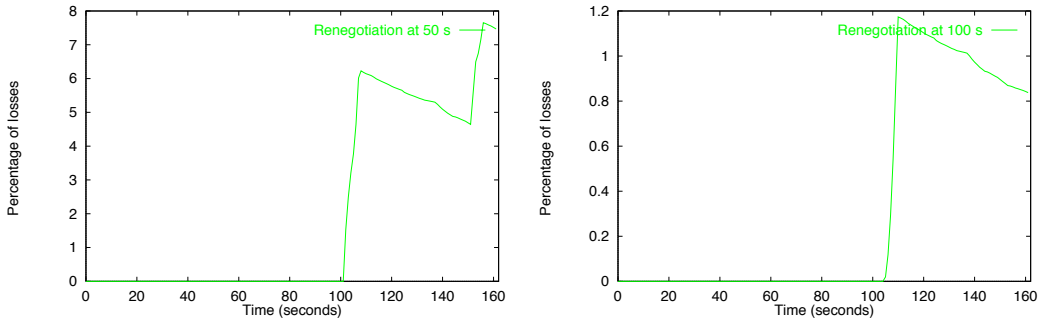
In Figures 5.12 and 5.13 we illustrate the losses for an average b_{max} (compared to the input traffic, $b_{max} = 6$ Mbits) for different renegotiation periods. We observe that for most of these cases the percentage of losses is not acceptable. It is different in the case of renegotiation at 100 seconds because here the renegotiation is quite infrequent.

5.4 Conclusion

In this chapter we characterised the RVBR service in terms of the time varying shaper model introduced in Chapter 4.

Then we used this result to study two aspects of the RVBR service, leaving aside the problem of traffic prediction:

- The first problem (local optimisation) is to find the optimal parameters (rates and bucket sizes) in *one* interval I_i, for two particular cost functions, given



(a) renegotiation every 50 seconds (b) renegotiation every 100 seconds

Figure 5.13: Percentage of losses in the reset approach

that we know the expected traffic in that interval. The solution are Algorithms 1 (Section 5.2.1) and 2 (Section 5.2.1).

- The second problem (global optimisation) is to find the optimal parameters (rates and bucket sizes) over a complete sequence of intervals interval I_i, for one particular cost function, given that we know the expected traffic over the whole sequence. We propose a solution based on a Viterbi-like algorithm (Algorithm 3 in Section 5.2.2).

Apart from the obvious consideration in terms of time and memory and/or computational cost, we showed that it is effective to use the local algorithm. In fact, in the examples we considered, the sequence of optima produced by the local algorithm is very close to the optimal sequence produced by the global algorithm. In particular, when the renegotiation period is not very small (i.e. $\geq$ 30 seconds), in most of the cases the sequence of local optima is equal to the optimal sequence, as illustrated in Figures 5.5-5.7.

In the cases we analysed, we found that the RVBR service is more efficient than the RCBR service in terms of the number of connections that can be accepted on a link with fixed capacity and buffer size. This is illustrated in Figures 5.8-5.10 and discussed in details in Section 5.3.2.

We illustrated that, if some inconsistency exists between network and user sides about the use of the "reset" or "no-reset" approach, then this may result in unac-

ceptable losses (or service degradation) due to policing. We gave an upper bound to the percentage of losses and we noticed that, in general, this upper bound is not acceptable, especially for small renegotiation periods. We also found, in the cases we analysed, that this limit can be easily approached. Some simulation results are given in Figure 5.12 in Section 5.3.4.

We also discussed the impact of the renegotiation period on the renegotiation cost, as one factor that can affect the renegotiation.

Chapter 6

RVBR for RSVP

This work in this chapter appeared in [67], [68], [81], [82] and [88].

6.1 Introduction

In this Chapter we show that RVBR can be efficiently used to renegotiate resource for the Internet traffic that takes the form of IntServ specification with RSVP reservation [2], [85].

In the next section we give a short description of certain aspects of RSVP relevant to this work and focus on the use of RSVP with Controlled-Load and Guaranteed QoS control service. In Section 6.3, we present the simulation of RVBR in a scenario where a sender transmits a MPEG2 video over a network using RSVP reservation protocol with Controlled-Load service. Finally, in Section 6.4, we describe the implementation design of a Video on Demand application, which is the first example of application RVBR-enabled.

6.2 RSVP with the Controlled-Load and Guaranteed QoS

6.2.1 Resource ReSerVation Protocol

Resource ReSerVation Protocol (RSVP) [2] addresses the problem of resource reservation in the Internet. It operates on top of IP and relies on standard Internet routing. Here we deal only with details directly related to the QoS specification. Other details on the RSVP specification can be found in Section E.3.3 in the Appendix.

RSVP allows the reservation of resources for a flow, seen as a sequence of datagrams. The flow descriptor, carried in the resource reservation message, contains the FLOWSPEC. This information is a combination of the traffic characteristic and the available resources in the network, therefore, is the receiver who is responsible for initiating the reservation.

The sender sends the characteristics of the traffic in the *Tspec* traffic descriptor, contained in the PATH message. The receiver establishes a resource reservation by issuing a RESV message upstream following exactly the inverse path of the PATH message. The RESV message creates a reservation state in each RSVP capable router along the path from the receiver to the sender.

The resource reservation request indicated in the RESV message has to pass admission control and policy control modules in all RSVP equipped routers and hosts on its way. The reservation is accepted if it passes these two checks; flow related parameters are set in the packet classifier and packet scheduler. If either of the checks fail, an error notification is returned. The packet scheduler is responsible for negotiation with the link layer in order to reserve the transmission resources. It is here that mapping, from the flow level QoS to the link layer QoS, takes place.

RSVP uses soft state for the reservation. The reservation is periodically refreshed (suggested refresh period is currently 30 seconds), i.e. the PATH and the RESV messages are reissued. The soft state does not imply that resources are renegotiated, because the traffic parameter specification can be reissued without changes. In fact, the original role of the soft state mechanism is to simplify the status management

in the routers, but it can be easily used and without additional costs, for expressing dynamic reservation changes in a straightforward way and thus can be easily used to support resource renegotiation, as we illustrate in Section 6.3.

6.2.2 Controlled-Load Service

Controlled-load service provides the client data flow with a quality of service closely approximating the QoS that the same flow would receive from an unloaded network element, but uses capacity (admission) control to assure that this service is received even when the network element is overloaded [18].

The end-to-end behaviour offered by the controlled-load service to an application, under the assumption of a correct functioning of the network, is expected to provide little or no delay and congestion loss. In this respect, the application may expect to experience a high percentage of successfully delivered packets by the network to the receiving end-nodes, without exceeding a certain minimum transit delay.

The sender provides the information of the data traffic it will generate in the *Tspec*. The parameters specified by the *Tspec* are: a peak rate p and a leaky bucket specification with rate r and bucket size b. In addition, there is a minimum policed unit m and a maximum packet size M. The service offered by the network ensures that adequate network resources will be available for that traffic. In the presence of non-conforming packets arriving (falling outside of the region described by the *Tspec* parameters), the QoS provided by the network to that flow may exhibit characteristics indicative of overload, including large numbers of delayed or dropped packets. In fact, the excess traffic is very probably forwarded as best-effort or dropped.

The controlled-load service is well suited to those applications that can usefully characterise their traffic requirements and are not too sensible to eventual delay or loss.

6.2.3 Guaranteed Service

Guaranteed service provides firm (mathematically provable) bounds on end-to-end datagram queueing delays. This service makes it possible to provide a service

that guarantees both delay and bandwidth [89].

This is obtained by guaranteeing the queueing delay. The queueing delay is a function of the *Tspec* parameters that express the traffic characteristic and of the *Rspec* that describe the desired service. The *Tspec* has the same form as for the controlled-load service, i.e a peak rate p, a leaky bucket specification with rate r and bucket size b, a minimum policed unit m and a maximum packet size M. The *Rspec* is composed by a rate R and a slack term S, where $R \geq r$ and $S \geq 0$. The slack term may be used inside the network to adjust the local reservations.

The sender provides the information of the data traffic it will generate in the *Tspec*. The receiver, on the basis of the target delay and the resources available in the network, starts the reservation adding the *Rspec* information. The service offered by the network guarantees that the requested network resources will be available for that traffic. In the case of non-conforming traffic, the excess traffic is very probably forwarded as best-effort or dropped.

Inside the network, the traffic can be reshaped, in order to restore its conformity to the *Tspec*. This is obtained with the use of the reshaping buffer. The amount of buffering required to reshape any conforming traffic back to its original token bucket shape is $b + Csum + (Dsum * r)$, where Csum and Dsum are the sums of the parameters C and D between the last reshaping point and the current reshaping point. The parameter D is intended to limit the variability in non-rate-based delay. C expresses the data backlog resulting from the deviation from a strict bit-by-bit service.

This service is intended for applications that need a firm guarantee on delay.

6.2.4 RSVP resource reservation protocol with CL and GS control services

RSVP resource reservation protocol does not define the internal format of the QoS object, because it is designed to be used with a variety of QoS control services.

The FLOWSPEC object carries information necessary to make reservation requests from the receiver(s) into the network, i.e. an indication of which QoS control service is being requested and the parameters needed for that service. The *Tspec*

carries traffic information usable by either the Guaranteed or Controlled-Load QoS control services and it carries information about traffic parameters of the desired reservation. The *Rspec* is only specified for Guaranteed service and it carries information to obtain the desired bandwidth and delay guarantees.

The information about the QoS control capabilities and requirements of the sending application and the network elements are carried in the ADSPEC. In the case of Guaranteed service, the ADSPEC carries data needed to compute the C and D terms passed from the network to the application.

6.3 RVBR Simulation

RVBR service uses the knowledge of the past status of the system and the profile of the traffic expected in the near future, which can be either pre-recorded or known by means of exact prediction. This scheme suits perfectly the dynamics of the traffic generated by multimedia applications. Moreover it naturally integrates with the soft state mechanism of RSVP, which allows for renegotiating the resources.

6.3.1 Simulation Scenario

As described in Chapter 5, an RVBR source is a time varying leaky-bucket shapers with two renegotiable leaky buckets ($J = 2$); one with rate r_i and depth b_i and the second with rate p_i and depth always equal to zero, plus a buffer of fixed size X. This suits perfectly with the service requested by a source sending Internet traffic that takes the form of IntServ specification with RSVP reservation [2], [85]. We show that the RVBR service can be used to renegotiate a resource reservation for Internet traffic with RSVP, where the sender sends a PATH message with a *Tspec* object that characterises the traffic it is willing to send. If we consider a network that provides a service as specified for the Controlled Load service (CL) [18], the *Tspec* takes the form of a double bucket specification [19] as given by the RVBR service. There is a peak rate p and a leaky bucket specification with rate r and bucket size b. Additionally there is a minimum policed unit m and a maximum packet size M. In this simulation, we ignore m and M, which are assumed to be

fixed. Following the RSVP specification, where a refreshing period of 30 seconds is suggested, we set the renegotiation interval size to this value. As defined with RVBR, p, r and b are recomputed at each renegotiation time, hence a *new Tspec* is issued. There is no additional signaling cost in applying a *Tspec* renegotiation at this point, even if there is some computational overhead due to the computation of the new parameters, or to the call admission control, etc. It is important to note here that, contrary to the negotiation of a new connection, with the renegotiation the reservation is never interrupted.

If the requested traffic specification cannot be supported by the network, the old traffic specification is restored and the network may not be able to accommodate the next traffic. Mechanisms to prevent this failure from occurring are still under study. Here we assume that the *Tspec* is accepted all over the network as well as at the destination, such that the source can transmit conforming to its desired traffic specification.

To apply the RVBR service in this scenario, we assume that at any time $t_i = 30 \cdot i$ the application knows (because pre-recorded or predicted) the traffic for the next 30 seconds. We further assume to know the cost to the network of the *Tspec*s (indicated by the cost function $u \cdot r + b$) and the upper bound to the bucket size b_{max} and to the bucket rate r_{max}. The backlog $w(t_i)$ and the bucket level $q(t_i)$ can be measured in the system. Then, with the RVBR service, we compute the *Tspec* that the sender will send at the next renegotiation time. The basic architecture of the sender node is described in Figure 6.1. In this context we do not consider delay issues (delay incorporation, as well as the extension to Guaranteed Service [89], is matter of further study).

6.3.2 Simulation results

In this section we describe how we use the local algorithm, defined in Section 5.2.1, to simulate a typical real case: transmission of MPEG2-encoded video using the IntServ Controlled Load service with the RSVP reservation protocol.

The basic architecture of the sender node is described in the introduction and illustrated in Figure 6.1.

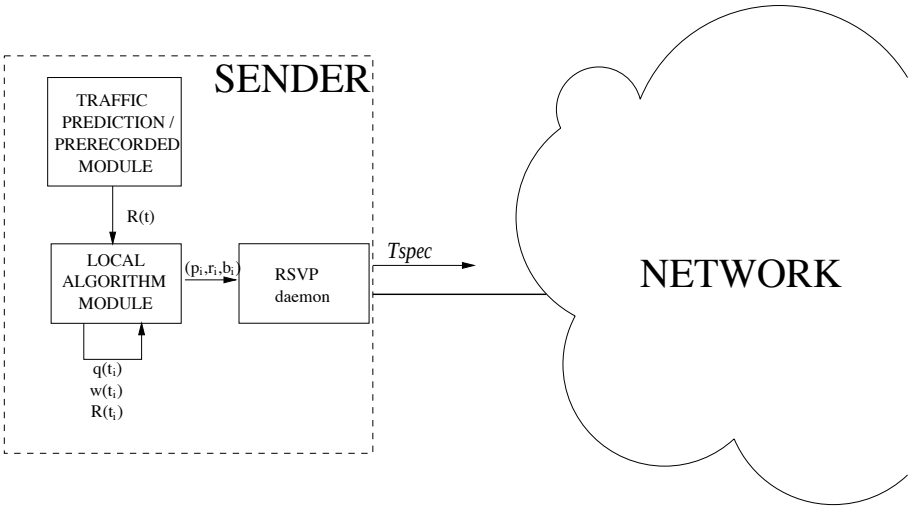


Figure 6.1: A basic architecture to support the usage of the local scheme for RSVP with CL service reservation: each 30 seconds $R(t)$ is predicted and used to compute the optimal p, r and b to generate the new *Tspec*.

In our simulations, we use a 4000 frame-long sequence that conforms to the ITU-R 601 format ($720 * 576$ at 25 fps). The sequence is composed of several video scenes that differ in terms of spatial and temporal complexities. It has been encoded in an open-loop variable bit rate (OL-VBR) mode, as interlaced video, with a structure of 11 images between each pair of I-pictures and 2 B-pictures between every reference picture. For this purpose, the widely accepted TM5 video encoder [90] has been utilised. The evolution of the input traffic is given in Figure 6.2.

The traffic generated by the video is transported by a trunk regulated by a RVBR service (p, r, b) with shaping buffer X. In this context we do not consider any scheduling issues, which is the subject of ongoing work. Therefore we assume that the video, with a total size of 550 Mbits, is transmitted in 163 seconds (25 frames pro second). The cost function is linear with u. We illustrate three scenarios:

Scenario 1: $X = 40$ Mbits, $r_{max} = 5$ Mbps, $b_{max} = 9$ Mbps and $u = 1$

Scenario 2: $X = 30$ Mbps, $r_{max} = 6$ Mbps, $b_{max} = 12$ Mbits and $u = 1$

Scenario 3: $X = 20$ Mbits, $r_{max} = 8$ Mbps, $b_{max} = 10$ Mbps and $u = 6$

The initial conditions are: $q(0) = 0$ and $w(0) = 0$. The file is pre-recorded and, given that we do not enter in scheduling matters, we know $R(t)$ for all t. At time t_i

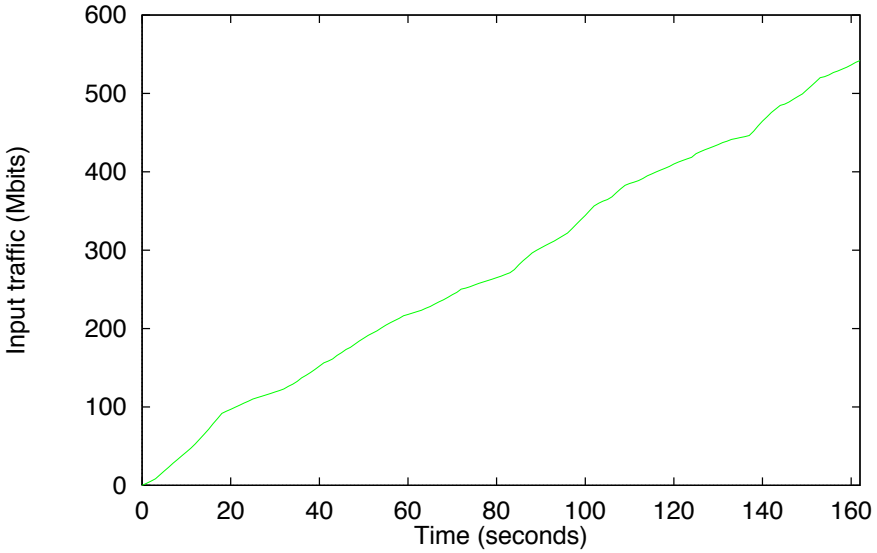


Figure 6.2: Traffic evolution of the sequence used as input in the simulation.

we know $R^*(t)$ for $t \leq t_i$, we measure $w(t_i)$, $q(t_i)$ and compute $\beta_i(t)$, as indicated by Equation 5.9 in Chapter 5. We obtain the optimal shaper parameters by applying the algorithm *localOptimum1* at Section 5.2.1 that we use to generate the *Tspec* that the sender will send at the next renegotiation time.

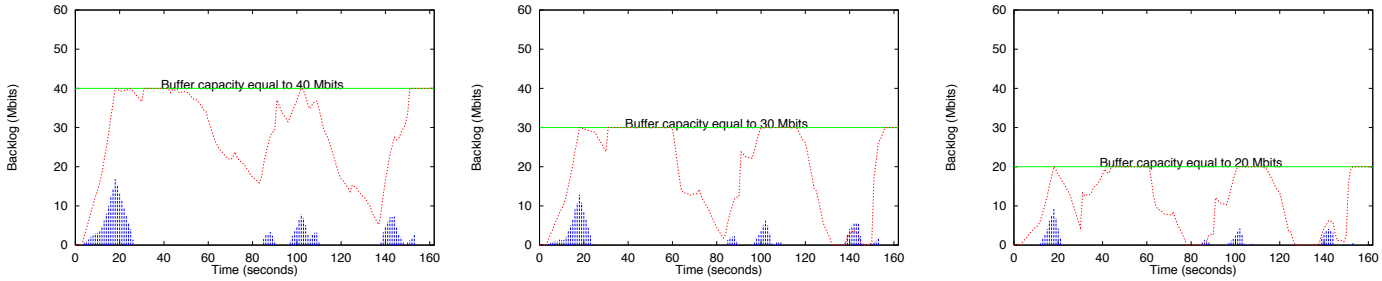
Backlog evolution with and without renegotiation

In Figure 6.3 we plot the backlog for the three scenarios in both cases where we apply the renegotiation and where we do not renegotiate [1]. In order to better distinguish the two approaches, the area of the curve representing the case without renegotiation is coloured.

We observe that in the beginning the curves representing the two approaches do not differ much. This is because the traffic is very heavy in the first 30 seconds and both traffic specifications conform to this traffic.

After that period the traffic rate decreases. The case without renegotiation has to keep the traffic specification negotiated at time $t = 0$, even if it is no longer adequate for the current demand. The resources allocated in the network are so large that it is possible to empty the buffer and thereafter the buffer is rarely used.

[1] Even in this case we compute the optimal traffic specification as introduced in [7].



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 6.3: Comparison of the shaping buffer used with renegotiation (white area) and without renegotiation (black area) for the three scenarios

The curve for the case where we used the RVBR service shows that the buffer is much better utilised, because the traffic specification decreases in the next intervals.

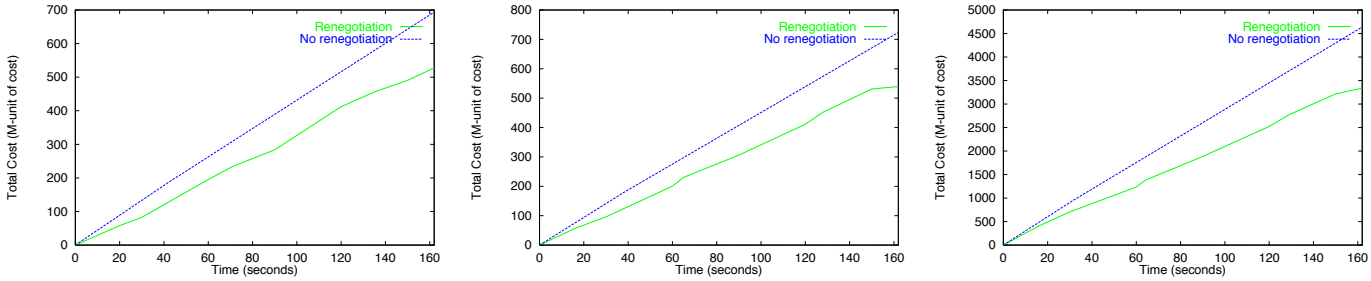
Therefore, in the approach where we apply the renegotiation with the RVBR service, the resources in the network are much better used. In fact, when the buffer is almost always filled the output is conforms to the traffic specification and this means that all the resources in the network are optimally used.

In the first scenario the usage of the buffer with renegotiation is 58%, while without renegotiation it is 13%. In the second scenario the percentages are 59% and 11%; in the last one they are 60% and 11%. In any case we have to remember that the optimisation is done for the worst case, and this explains why, when we do not renegotiate, the buffer never fills completely.

Cost evolution with and without renegotiation

In the graphs in Figure 6.4 we compare the two approaches in terms of the cost of the traffic specification to the network.

The cost of the traffic specification is given in terms of the linear cost function used by the RVBR service in order to compute the optimal traffic parameters. In the previous section we showed, for the case where we renegotiate the traffic specification, a better utilisation of the shaping buffer, which coincides with a better allocation of all the resources into the network. The additional result we derive



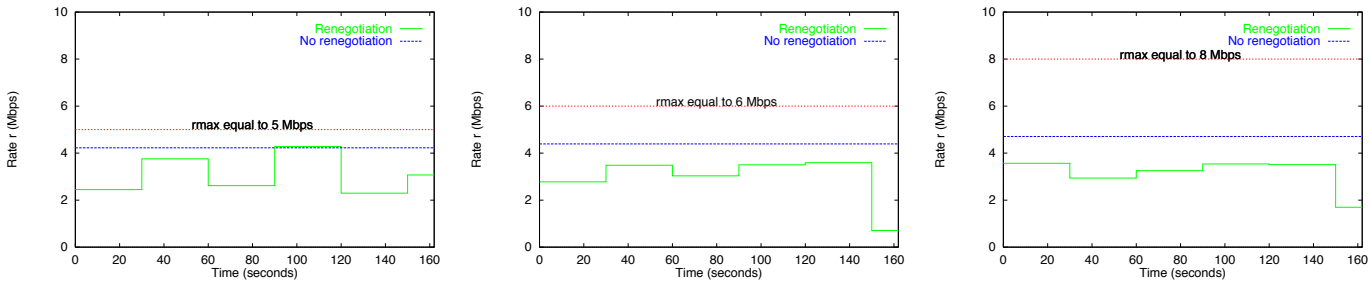
(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 6.4: Comparison of the cost of allocating a renegotiated traffic specification and a traffic specification without renegotiation for different scenarios. The cost of the traffic specification is given in "millions of unit of cost" (M-unit of cost) and computed with the linear cost function used for the optimisation.

from these other figures is that there is also a substantial advantage from the cost point of view in reallocating, because the cost of the traffic specifications is in general smaller.

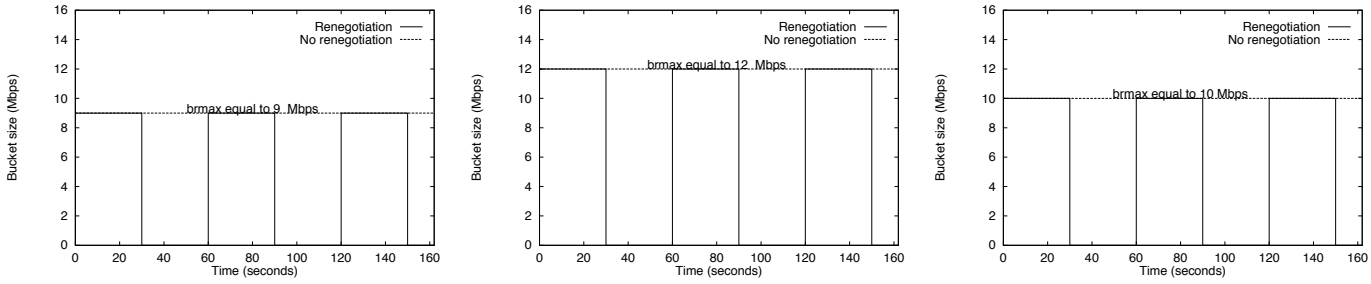
Traffic specification parameters evolution with and without renegotiation

Figures 6.5 and 6.6 illustrate the fact that with renegotiation we can optimise the resources requested to the network and therefore at the end the total r and b allocated in this case are in general smaller. We also notice that inside an interval the RVBR service might allocate a *Tspec* that is larger than the one used when not renegotiating. This occurs when the traffic is very bursty and the buffer is full from the previous interval. For scenario 1 this situation occurs also at the forth interval ($90 - 120$ seconds), as illustrated in Figure 6.5. This happens because the buffer is full and the bucket is not sufficient to absorb the burstiness of the input traffic. It does not take place in scenario 2 and 3, because there is more bucket available and therefore the application can request a larger bucket b.



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 6.5: Comparison of the evolution of the rate r with renegotiation and without renegotiation for different scenarios



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Figure 6.6: Comparison of the evolution of the bucket b with renegotiation and without renegotiation for different scenarios

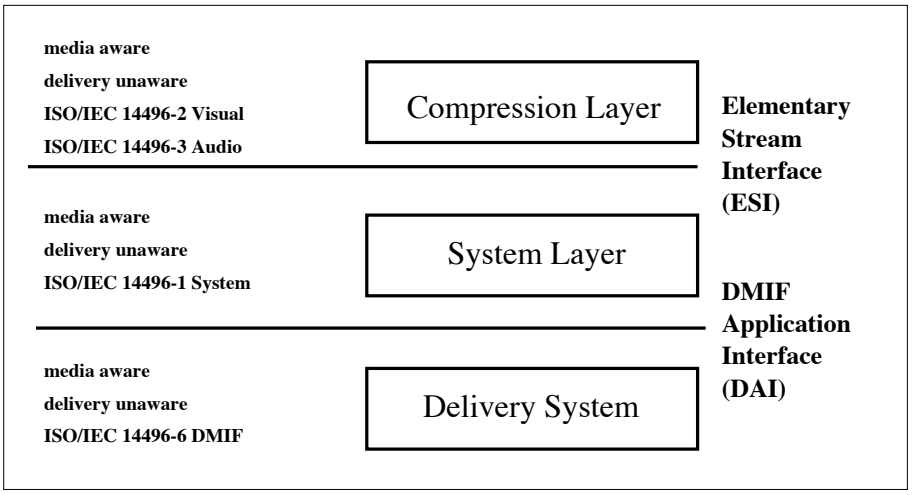


Figure 6.7: Generic MPEG-4 terminal architecture

6.4 An example of RVBR-enabled application

The ACTS-DIANA project [44] is implementing a first prototype of a multimedia application supporting RVBR. The selected application is Applications Retrieving Multimedia Information Distributed over ATM (ARMIDA[2]) [91].

ARMIDA is a video on demand application initially designed (ARMIDA2) [92], according to the DAVIC [93] model, to support the transmission of MPEG2 videos directly over AAL5/ATM as specified by ATM Forum [60].

The current release (ARMIDA4) is the evolution of the previous one and provides features to support MPEG1/MPEG2/MPEG4 over a generic transport network using IP as Internetworking protocol.

This new release has been designed according to the ISO/IEC MPEG architecture defined for MPEG4 [94]. ARMIDA4 provides features to display online several remote multimedia data flows improving the previous release by the MPEG4 add-on. MPEG4 supports different data flows, as video, audio, images 2D or 3D etc. that are multiplexed and managed by the same MPEG4 client displaying all together on the same end-system. A flow of homogeneous data is called Elementary Stream.

The boundary between the Compression Layer and the Systems Layer is named Elementary Stream Interface (ESI) and its minimum semantic is specified in ISO/IEC

[2]supported in the the project by Finsiel partner.

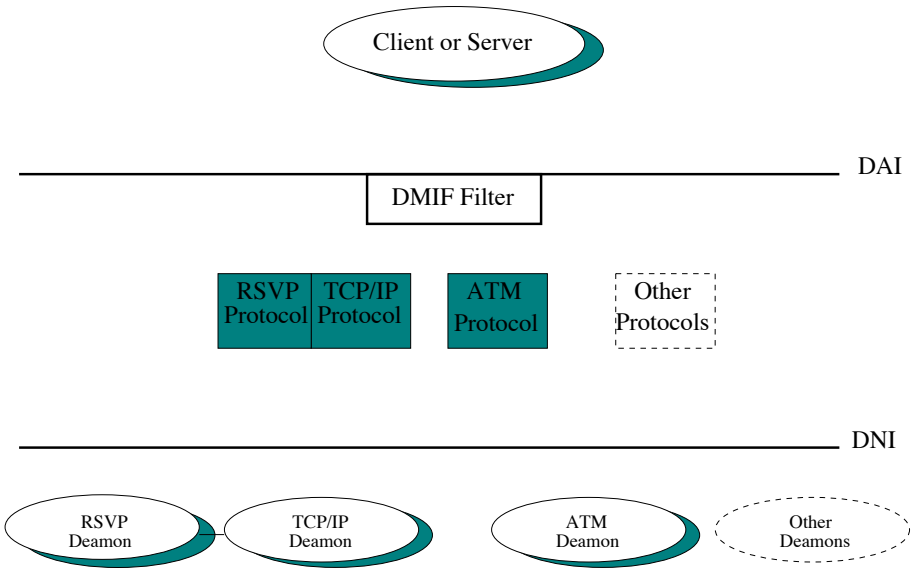


Figure 6.8: ARMIDA4 Architecture

14496-1 (MPEG-4 Systems) [94].

The boundary between the System Layer and the Delivery Layer is named DMIF/Application Interface (DAI) and its minimum semantic is specified in ISO/IEC 14496-6 (Delivery Multimedia Integration Framework) [95].

The Compression Layer is responsible for media encoding and decoding; Audio (MPEG-4 part 3) and Video (MPEG-4 part 2), both Synthetic and Natural, are dealt with at this layer. The Delivery Layer (MPEG-4 part 6) ensures transparent access to MPEG-4 content irrespective of the Delivery technology (Delivery technology is a term used to refer to a transport network technology -e.g. the Internet, or an ATM infrastructure-, as well as to a broadcast technology or local storage technology). The Systems Layer (MPEG-4 part 1) interprets the scene description and manages Elementary Streams and their synchronisation and hierarchical relations, their composition in a scene. It is also meant to deal with user interactivity.

6.4.1 ARMIDA Architecture

A rough idea of ARMIDA4 architecture is given in Figure 6.8. In the ARMIDA4 architecture we can find the same layers defined in the MPEG4 model, confirming

compliance to the standard. The DAI makes the upper layers independent from the specific network providing transport capabilities.

Multiple QoS: DMIF

The main goals of DMIF are:

- To hide the delivery technology details
- To manage real time QoS sensitive channels
- To allow service providers to log resources per session for usage accounting, gathering information on data transfer, etc ...
- To ensure interoperability between end-systems

DMIF defines a communication architecture that hides the details of delivery technologies below an interface that is exposed to the application, called DMIF Application Interface (DAI). Delivery technologies include transport network technologies (e.g. ATM, Internet, etc...) as well as broadcast or multicast technologies. DAI separates the delivery aware and delivery unaware layers of the ISO/IEC 14496 [94] terminal architecture. It is a semantic API they allows the development of application irrespectively of the delivery support. DMIF provides QoS management aspects and mechanisms to gather information about data transfer and resources utilisation opening to the implementation of billing policies. All these features are supported via the DDSP (DMIF Default Signalling Protocol), that makes the intermediate layer between the signalling network protocol (e.g. Q.2931 in ATM or RSVP over general IP networks).

QoS Management

The QoS is managed directly at the application layer and it is bound to the data flow. A client selects a set of data flows (Elementary Stream), e.g. video and image and then contacts the server for receiving data. The related QoS required from selected flow is stored together with data: in this phase the application layer reads the QoS required and gives this information to DMIF via DAI. DMIF, activates

the RSVP sending a PATH containing a *Tspec* compiled with parameters fitting required QoS.

The following parameters are passed to DMIF:

- AVG BitRate
- MAX BitRate
- MAX AU_SIZE
- Priority (for future architectures)
- Max Delay
- Service Constraint (GS or CL)

The negotiation phase corresponds to PATH message sending. The RSVP module, before starting the negotiation phase, builds up the QoS according to IntServ specification and fits the *Tspec* for the PATH message on this basis.

On the server side, depending on the Service Constraint value, the data sending starts only if the negotiation phase succeeds. The application can be considered *signalling aware*, i.e. totally controlled by the related signalling.

For example, if a Guaranteed Service is required [89] the application cannot start if the negotiation does not succeed.

ARMIDA4 behaviour

ARMIDA4 provides MPEG4 data transfer supporting multimedia information according to MPEG4 specification.

The multimedia data, MPEG4 coded, are stored in a Data Server together with information about QoS requirements. The user asks for one or more services (e.g a video service) via an HTML interface. In each file of requested data several Elementary Streams are stored, each one with its QoS descriptor. The server sends all the information about the requested data to the DMIF that attempts to set-up the necessary communications. The DMIF reads the QoS descriptors building-up the QoS needed for these communications. Then it translates them according to

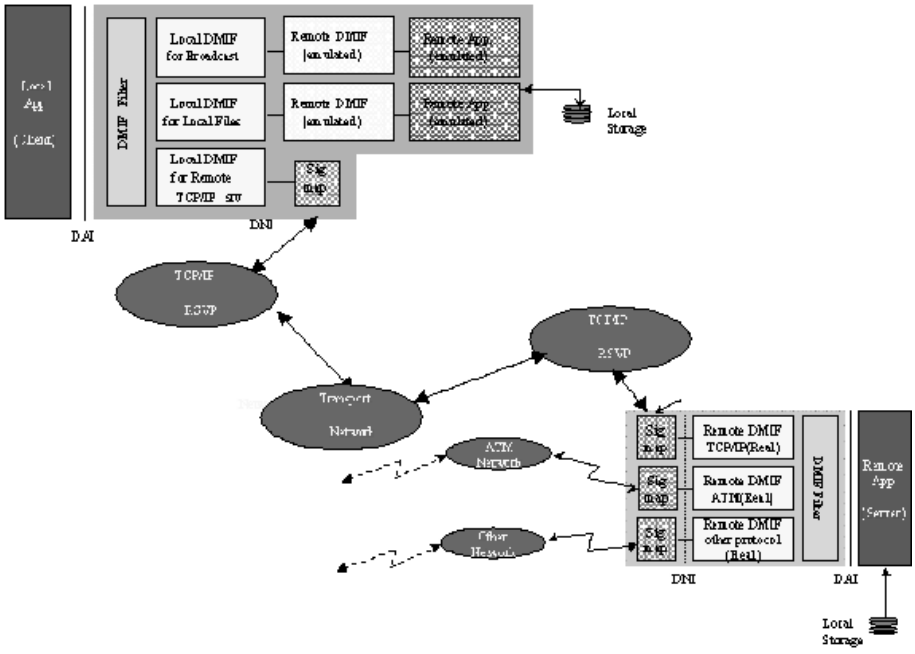


Figure 6.9: ARMIDA4 Client-Server configuration

the available delivering network and starts the negotiation. If the negotiation phase finishes successfully, the data transfer starts and the user can consume the request data. The configuration of ARMIDA4 is illustrated in Figure 6.9.

RVBR enabling

The introduction of RVBR within ARMIDA4 makes an impact on three layers:

1. Application Layer: several QoS descriptors must be handled. The current standards define that only a single QoS descriptor can be associated to an Elementary Stream. In order to introduce the renegotiation, it is necessary to define the association between several QoS descriptors and an Elementary Stream. Additionally, the structure for maintaining and managing several QoS descriptors for the same data flow. It means that a data flow must store more than one QoS descriptor and a reference to the related portion of data.

2. DMIF layer: at each renegotiation, it must provide the new QoS descriptors to RSVP. The DMIF level must be able, first, to identify when an old QoS

expires to start a new QoS and, second, to interact repetitively with the RSVP demon.

3. RSVP layer: several renegotiation phases must be managed The RSVP must modify the Tspec sent with the PATH message asking for a new reservation.

Implementation issues:

- Data stream Fragmentation: input data stream must be segmented in several intervals each one maintaining its QoS descriptor. Problems related to the dimension of each interval must be analysed.

- Synchronisation between the renegotiation phase and the data sending: the reallocation must be performed according to data sending to guarantee that resources are available when needed.

- Renegotiation signalling implementation: the PATH message, in most of existing RSVP packages are automatically generated from the system; in this case the server should be able to send a new PATH message with a different *Tspec* without tearing down the existing connection

- Recovery in case of renegotiation failure

The implementation of ARMIDA4 RSVP and RVBR enabled is the subject of ongoing work.

6.5 Conclusion

We illustrated how the RVBR service can be applied to the RSVP *Path* message generation. This is based on the algorithm proposed for the local optimisation problem in Chapter 5. A numerical example of this is given in Section 6.2.1, where we also compare the performance of transmitting a MPEG2 video trace both with and without renegotiation. The results of our simulation (see Figures 6.3 - 6.6) suggest that renegotiation allows better use of network resources and that with

protocols as RSVP, where there is no additional cost for signaling (or so we mainly assume), it is better to renegotiate.

These results demonstrate that the RVBR service can be easily and efficiently adopted by video applications requiring guaranteed service. In this respect, we introduced the design of a video on demand application RVBR-enabled, which also became the first instance of an application that renegotiates RSVP traffic specification.

Chapter 7

Conclusion

In this work we have shown that a dynamic allocation of the network resources allows us to reach an optimal usage of them and guarantee the QoS requirements of the applications. The thesis has shown through mathematical model, simulation and implementation that a renegotiable VBR service results in a significant benefit in terms of resource optimisation while guaranteeing the QoS requirements. Our work allows us to make the following statements:

- *QoS renegotiation on the customer side can be simple.* We have shown that in areas where end-to-end ATM connectivity already exists and there is support for renegotiation, there is a relatively simple way for IP applications to use the QoS of ATM, specifically, by using *Arequipa* and its simple API. We demonstrated this in the case of *Vic* that became the first instance of an application that is able to tune bandwidth at run time.

- *The static VBR problem can be solved.* We have shown that, for all reasonable cost functions, the static VBR problem can be reduced to a one-dimensional problem. Furthermore, for the specific case of a cost function, it possible to derive algorithms that can easily be implemented for real time computation.

- *VBR is only beneficial in respect to CBR.* We have given a characterisation of the optimal traffic descriptor of a VBR trunk under buffer constraint, assuming to know the input traffic. In particular, the optimal peak rate for the

VBR trunk results equal to the peak rate of the CBR (the deterministic equivalent capacity e_X). In this sense, the VBR, with the additional parameters (sustainable rate and burst size), is only beneficial.

- *The renegotiation can be analytically modelled.* Existing models are not suitable for modelling a situation where the traffic changes dynamically and consequently the network resources changes. This is because they do not take into account the traffic that is present in the shaper at the transient moments. We have modelled such a situation with a class of time varying shapers that we have called *time varying leaky-bucket shapers*. The time varying leaky bucket shaper class is characterised by network calculus. To our knowledge, this is the first model, which takes into account non-zero conditions at the transient time. This innovative result forms the basis of the RVBR service.

- *The time varying leaky-bucket shaper model can be applied to characterise the RVBR service and the mathematical model proposed for the RVBR service is suitable to solve the dynamic VBR problem.* We have derived the input-output characterisation of the RVBR service as a special case of the time varying leaky bucket shaper. An RVBR source is a time varying leaky-bucket shaper with two renegotiable leaky buckets ($J = 2$); one with rate r_i and depth b_i and the second with rate p_i and depth always equal to zero, plus a buffer of fixed size X. For the RVBR service, the dynamic VBR problem is equivalent to the problem of computing the RVBR parameters for the next interval. Therefore, as well as for the static VBR problem, assuming an objective function, the dynamic VBR problem can be solved.

- *The application of our mathematical model to the dynamic VBR problem results in simple, efficient algorithms.* Using the RVBR mathematical model, we have provided explicit algorithms that solve the dynamic VBR problem, when the knowledge of the input traffic is limited to the next interval (*local optimisation problem*) and when we dispose of the complete input traffic description (*global optimisation problem*). For the local problem we have proposed two versions: one, when the cost function is represented by a linear cost

function and the other, when we compare two solutions in terms of the number of connections (with those parameters) that would be accepted on a link with capacity C and physical buffer B. This second cost function was also used for defining an algorithm for the global problem.

- *The renegotiation is effective, it is valid to use the local algorithm and the "no reset" approach is essential for a lossless service* We have simulated the RVBR service versus a VBR service and illustrated that there are indisputable advantages with the RVBR service.

- *It is valid to use the local algorithm* By simulation, we found that the sequence of optima produced by the local algorithm is very close to the optimal sequence produced by the global algorithm and, in several cases, is even equal to it.

- *The "no reset" approach is essential for a lossless service* The "reset" approach would be easier to implement. However, simulation has shown that if the renegotiation does not take into account the bucket conditions at the transient moment the source is very likely to experience losses due to policing.

- *RVBR performs better than RCBR.* We have simulated the RVBR service versus the renegotiable constant bit rate (RCBR) service and illustrated that the RVBR approach can provide substantial benefits.

- *The size of the renegotiation interval affects on the efficiency of the RVBR service.* In general, we expect that the larger the renegotiation period is, the higher the cost of the traffic specification. We have shown that with a local approach this is not always true. We have found that the local optimum of a larger period can be less expansive than the sum of the cost for a smaller renegotiation period on the same interval. We have also shown that this is highly related to the input traffic.

- *The algorithms provided for solving the dynamic VBR problem with the RVBR service can easily be implemented in real applications.* RVBR service uses the knowledge of the past status of the system and the profile of the traffic expected in the near future, which can be either pre-recorded or known by means of exact

prediction. This scheme suits perfectly the dynamics of the traffic generated by multimedia applications that handle pre-recorded or classified video traffic. Moreover it naturally integrates with the soft state mechanism of RSVP, which allows for renegotiating the resources. We have presented the implementation design of ARMIDA, as a first example of application that renegotiates RSVP traffic specification with RVBR.

Future work on RVBR service includes trials with real application RVBR enabled and study on the renegotiation period, as well as the integration of the network delay and the application to Guaranteed Service.

We argue that, if the experimentation with real application RVBR enabled are positive, this service should be proposed as a possible service for applications and networks that use RSVP as reservation protocol.

In this respect, the development of mechanisms to prevent the failure occurring because the requested change to the traffic specification cannot be supported by the network is essential.

In further work our results for the class of time varying leaky-bucket shapers will be used to model network resources renegotiation in other scenarios, as, for instance, in the video smoothing case [96].

Appendix A

Proofs of propositions in Chapter 3

Proposition 4 (Continuity of RequiredBuf) *RequiredBuf(y, z) is continuous with respect to y.*

Proof: *Inside each case the function is continue. Thus, we must demonstrate that there are no discontinuity point passing from one case to another. It is sufficient to compute the limiting values at each end point, which reduces the problem to taking limits with respect to single variables.*

CASE 2 *When m_0 becomes smaller than NR, then we pass in* CASE 3 *or in* CASE 4. *In both cases,* $\lim_{m_0 \to NR} \text{requiredBuf}(y_0, z_0) = 0$.

CASE 3
- *When R_0 becomes larger than NR, then we pass in* CASE 5. *In this case,* $\lim_{R_0 \to NR} t_0(NR - R_0) + (t_c - t_0)(NR - m_0) = (NR - m_0)(t_c - t_0)$.
- *When t_c becomes smaller than t_0, then we pass in* CASE 4. *In this case,* $\lim_{t_0 \to t_c}(NR - m_0)(t_c - t_0) = 0$.

CASE 4 *When R_0 becomes larger than NR, then we pass in* CASE 6. *In this case,* $\lim_{R_0 \to NR} t_c(NR - R_0) = 0$.

CASE 5 *When t_0 becomes larger than t_c, then we pass in* CASE 6. *In this case,* $\lim_{t_0 \to t_c} t_0(NR - R_0) + (t_c - t_0)(NR - m_0) = t_c(NR - R_0)$.

Thus, given that requiredBuf is continue in all limits, it is continue.

Proposition 5 (Upper bound of Peak Cell Rate of $y_0 \in \mathcal{S}(z)$) *The Peak Cell Rate of the elements of $\mathcal{S}(z)$ is smaller than or as large as NR.*

Proof:

If $y_0 \in$ CASE 5, or CASE 6 *it is evident.*

Assume $y_0 \in$ CASE 2.
Let $R_0 > NR$.
$\Rightarrow y_0' = (NR, \tau_0, NR) \preceq y_0$, $y_0' \in$ CASE 2.
$\Rightarrow \text{requiredBuf}(y_0', z_0) = \text{requiredBuf}(y_0, z_0) = 0$, $\Rightarrow y_0 \notin \mathcal{S}(z)$, *because* $y_0' \preceq y_0$.

Assume $y \in$ CASE 3.
Let $R_0 > NR$.
$\Rightarrow y_0' = (m_0, \tau_0, NR) \preceq y$, $y_0' \in$ CASE 3.
$\Rightarrow \text{requiredBuf}(y_0', z_0) = (NR - m_0)(t_c - t_0) = (NR - m_0)(t_c - t_0) = \text{requiredBuf}(y_0, z_0)$, *not possible because $y \in \mathcal{S}(z)$.*

Assume $y \in$ CASE 4.
Let $R_0 > NR$.
$\Rightarrow y_0' = (m_0, \tau_0, NR) \preceq y_0$, $y_0' \in$ CASE 4.
$\Rightarrow \text{requiredBuf}(y_0', z_0) = \text{requiredBuf}(y_0, z_0) = 0$, *not possible because $y_0 \in \mathcal{S}(z)$.*

Proposition 6 (Lower bound of Peak Cell Rate of $y_0 \in \mathcal{S}(z)$) *The Peak Cell Rate of the elements of $\mathcal{S}(z)$ is larger than or as large as $NR - X/t_c$.*

Proof: *If $m_0 > NR - X/t_c$, then is evident. Otherwise:*

If $R_0 < NR - X/t_c$, thus we are in CASE 5, or in CASE 6.

Assume we are in CASE 5. *$m_0 \leq R_0 \Rightarrow m_0 = R_0 - a$. From the value of requiredBuf in* CASE 5, *we derive that $\tau_0 = \frac{NRt_c - X - (R_0 - a)t_c}{R_0 - a}$. Thus:*
$t_0 = \tau_0 m_0/(R_0 - m_0) = \frac{NRt_c - X - (R_0 - a)t_c}{a} = t_c + (NR - X/t_c)(t_c/a) > t_c$
Thus this value of y is not in CASE 5, *but in* CASE 6. *This implies that , in* CASE 5, *there are not solution in $\mathcal{S}(z)$ for $R_0 < NR - X/t_c$.*

If we are in CASE 6*, to belong to* $\mathcal{S}(z)$*,* R_0 *must be equal to* $NR - X/t_c$*. This implies that , even in* CASE 6*, there are not solution in* $\mathcal{S}(z)$ *for* $R_0 < NR - X/t_c$*.*

Proposition 7 (Optimal value for peak cell rate) *The optimal value for peak cell rate is the lower bound:*
$NR - X/t_c$*.*

Proof:

$t_c \leq t_0$ $X = (NR - R_0)t_c \Rightarrow R_0 = NR - X/t_c$*.*

$t_c > t_0$ $X = (NR - R_0)t_0 + (NR - m_0)(t_c - t_0) = NRt_c - m_0(\tau_0 + t_0)$ *that is independent by* R_0*. For this reason we must set it to its lower bound* $\Rightarrow R_0 = NR - X/t_c$*.*

Proposition 8 (Solution Space for RequiredBuf) *The Solution Space for requiredBuf is given by:*
$\mathcal{S} = \{y_0 = (m_0, \frac{NRt_c - X - m_0 t_c}{m_0}, NR - X/t_c)\}$,
$Nm \leq m_0 \leq NR - X/t_c$*.*

Proof:
Assume that exists $y_0' \in \text{requiredBuf}^{-1}$ *and* $y_0' \preceq y_0$

Let $y_0' \in$ CASE 2*.*
We define $\mathcal{S}_2 = \mathcal{S}(z) \bigcap$ CASE2*.*
$NR \leq m_0 \leq R_0$*. For Proposition 5* $R_0 \leq NR$*.*
$\overset{y_0' \in \mathcal{S}_2}{\Rightarrow}$ $NR = m_0 = R_0; \text{requiredBuf}(y_0', z_0)$ *does not depend on* τ_0*, that is thus equal to 0.*
$y_0' = (1/NR, 0, NR)$
Assuming $y_0' \in$ CASE 2*, we assumed that* $X = 0$*, thus* $y_0' = y_{(R_0 = NR, m_0 = NR, X = 0)}$*.*

Let $y_0' \in$ CASE 3*.*
$NR \leq R_0$*. For Proposition 5* $R_0 \leq NR$*.*

$\overset{y'_0 \in \mathcal{S}_3}{\Rightarrow} NR = R_0$ & requiredBuf$(y'_0, z_0) = X \Rightarrow \tau_0 = \frac{NRt_c - X - m_0 t_c}{m_0}$
thus $y'_0 = y_{(R_0 = NR)}$.

Let $y'_0 \in$ CASE 4.
$NR \leq R_0$ & $t_0 \geq t_c$. *For Proposition 5* $R_0 \leq NR$.
$\overset{y'_0 \in \mathcal{S}_4}{\Rightarrow} NR = R_0$ & $t_0 = t_c \Rightarrow \tau_0 = \frac{NRt_c - X - m_0 t_c}{m_0}$.
Assuming $y'_0 \in$ CASE 4, *we assumed that* $X = 0$, *thus* $y'_0 = y_{(R_0 = NR, X = 0)}$.

Let $y'_0 \in$ CASE 5.
$R_0 \leq NR$ & $t_c \geq t_0$.
$\overset{y'_0 \in \mathcal{S}_5}{\Rightarrow}$ requiredBuf$(y'_0, z_0) = X \Rightarrow \tau_0 = \frac{NRt_c - X - m_0 t_c}{m_0}$ *thus* $y'_0 = y_0$.

Let $y'_0 \in$ CASE 6.
$R_0 \leq NR$&$t_c < t_0$.
$\overset{y'_0 \in \mathcal{S}_6}{\Rightarrow} R_0 = NR - X/t_c$ & $t_c = t_0$
$\Rightarrow \tau_0 = \frac{NRt_c - X - m_0 t_c}{m_0}$ *thus* $y'_0 = y_{(R_0 = NR - X/t_c)}$.

Corollary 5 (Optimal burst time of VT) *The optimal burst time of VT is given by* $t_0 = t_c$.

Proof:
If $R_0 = NR - X/t_c$, *then* $t_0 = \tau_0 m_0/(R_0 - m_0) = \frac{(NR - m_0)t_c - X}{NR - X/t_c - m_0} = t_c$.

Appendix B

RM&R Architecture

This work in this chapter appeared in [46], [47], [48] and [49].

In this appendix we describe the architecture design of the Resource Management and Routing (RM&R) architecture build upon the VT solution and a dynamic resources management scheme, which estimates the changes in the traffic.

B.1 Resource Management Scheme used in combination with the VT solution

The purpose of this architecture is to allocate bandwidth to the VPCs following the changes in the traffic. This architecture is built upon the VT solution and assumed to work in a ATM network.

A VT is a virtual path connection (VPC) setup by the network in order to reduce connection awareness at the transit nodes. A virtual trunk is therefore considered as a connection by the network supporting it (the VP network), and as a logical trunk by the connections supported. In this context, VTs are considered to be VBR connections.

The VT solution is combined with a dynamic resources management scheme [11], [12], [13] and [14], which estimates the changes in the traffic. The integration of the VT solution and the dynamic resources management scheme results in a virtual trunk that changes its own connection descriptor dynamically (by negotiation with

the network supporting the virtual trunk).

Here, for each VPC, we have a tunnelling scenario where a number of VBR Virtual Channel Connections (VCCs) are multiplexed onto a VBR VPC at a node that acts as a general shaper and the VBR VPCs are multiplexed on the network. This is called the *VBR-over-VBR* approach. In the more traditional VBR-over-CBR approach, VBR VCCs are multiplexed onto a CBR VPC. In this case, it is no longer possible to multiplex the VPCs on the network.

We note that these approaches assume that at each period the resources (shaping buffer, maximum burst tolerance) are completely available, like at the initial time. This is, of course, not true in a real scenario, but we believe that, in the scenario under consideration, we can make this assumption. In fact, we introduce several factors of overestimation: the input flows are described by means of their arrival curves (VBR traffic descriptors), which are upper bounds to the generated input traffic; the VT is described by means of its service curve (a CBR or a VBR traffic descriptor), which is a lower bound to the service offered; and, finally, we use a worst case approach to optimise the VT parameters. All these steps add some approximation to the algorithms.

B.1.1 Dynamic, periodic bandwidth allocation scheme

In this centralised resource management (RM) method, VPC bandwidths are reallocated at periodic intervals. In the sequel, the updating interval is denoted by t_u. The objective is to allocate for each VPC only as much bandwidth as needed to satisfy the stationary blocking probability target, which may be class-specific. Thus, it may happen that not all the capacity of the physical links is allocated to VPCs. The original references are [11], [12], [13] and [14]. The method is based on the knowledge of the number of active connections per VPC. In addition, the average call arrival intensities and the mean holding times for all traffic streams are needed. We use this method in conjunction with the static VBR optimisation scheme in order to achieve an optimal renegotiation of the VPC.

The RM method includes the following three phases.

1st phase

Allocation is first made for each VPC separately. Consider a VPC. Let n denote the number of active VCC connections conveyed by the VPC. Denote by λ and T the arrival rate and the mean holding time of such VCC connection requests, respectively. In the simulations, the two statistical parameters λ and T are assumed to be known. Let $\rho = \lambda T$.

First is calculated the maximum number N of connections the VPC should support during the next updating interval in order that the blocking probability during the interval be less than $\varepsilon/2$,[1] where ε is the target blocking probability for the class. For this we need the following function:

$$N = \text{transientErlangRequirement}(n, \rho, \epsilon/2, t_u/T) \tag{B.1}$$

In principle, there are precise numerical methods to implement this function [97]. However, these methods are far too time-consuming for our purposes. Thus a simple approximation is needed. In the simulations, the following (rather crude) approximation is used:[2]

$$N = np(t_u/T) + N_\infty(1 - p(t_u/T)) \tag{B.2}$$

where

$$p(t) = \exp(-t)$$

and

$$N_\infty = \text{stationaryErlangRequirement}(\rho, \epsilon/2)$$

The function stationaryErlangRequirement utilises the ordinary Erlang blocking formula,

$$\text{stationaryErlangRequirement}(\rho, \epsilon/2) = \min\{N\text{—Erlang}(N, \rho) \leq \epsilon\}$$

[1] The (vague) heuristic behind this is as follows: the upper limit N_∞ for N is chosen so that the proportion of time when there are N_∞ active connections is (approximately) $\varepsilon/2$. In this state, all the incoming connection requests are rejected. On the other hand, when there are less than N_∞ active connections, the proportion of time for which is $1 - \varepsilon/2$, the dimensioning is made so that the proportion of rejected connection requests would be $\varepsilon/2$. Thus, the overall blocking probability becomes (approximately).

[2] According to the simulations made, this dimensioning formula seems to function when t_u/T is great enough, say 1. However, with smaller values, the allocations seem to be too small.

2nd phase

In the second phase, an allocation is made for each physical link separately. Consider a physical link. Let C^l denote its capacity (bandwidth)and denote by K the number of VPCs conveyed. The VPCs are indexed by k.

As the result of the first phase we have the maximum number N_k of connections to be supported by each individual VPC k. This is converted into a bandwidth requirement C_k. For this we need the CAC function called requiredBandwidth (see Section B.1.2),

$$C_k = \text{requiredBandwidth}_k(N_k)$$

After the bandwidth requirements are calculated we have to check whether the link capacity is sufficient, i.e.

$$\sum_k C_k \leq C^l$$

If this is true, we can step into the final phase. Otherwise the allocations must be *adjusted* not to exceed the capacity available. In the latter case, we first calculate the bandwidth requirements c_k of the existing connections. For this we need the number n_k of active connections and (again) the CAC function requiredBandwidth (see Section B.1.2),

$$c_k = \text{requiredBandwidth}_k(n_k)$$

The remaining capacity is denoted by R,

$$R = C^l - \sum_k c_k$$

It is shared as fairly as possible, the fair share for VPC k defined by

$$\frac{C_k - c_k}{\sum_i C_i - c_i}$$

Thus, we have the following adjusted capacities:

$$\tilde{C}_k = c_k + R\frac{C_k - c_k}{\sum_i C_i - c_i}$$

Note that

$$\sum_k \tilde{C}_k = C^l$$

By using the (inverse) CAC function called allowedNrCalls (see Section B.1.2), we may calculate the adjusted maximum number of connections to be supported by VPC k,

$$\tilde{N}_k = \text{allowedNrCalls}_k(\tilde{C}_k)$$

After this, the (real) capacity allocations are as follows:

$$\tilde{\tilde{C}}_k = \text{requiredBandwidth}_k(\tilde{N}_k)$$

which is less than or equal to $\tilde{C}_k$. Thus, there may still remain some capacity left over, namely

$$R = C^l - \sum_k \tilde{\tilde{C}}_k$$

This remaining capacity may still be utilised by traffic classes with lower bandwidth demands.

3rd phase

Finally, VPC bandwidths are adjusted at the network level. However, since in our setting each VPC traverses exactly one physical link, no more adjustments are needed.

B.1.2 CAC functions: use of the VT solution

The purpose of the CAC functions is to calculate the required bandwidth given the number of homogeneous connections and their class or to calculate the allowed number of homogeneous connections given the bandwidth available and the traffic class. The former function is called requiredBandwidth and the latter one allowedNrCalls. To this purpose we use the results obtained in Sections 3.3.1 and 3.4 for the VT solution. Below we describe the two approaches to the connection admission control used. We first present the traditional VBR-over-CBR approach and then the VBR-over-VBR approach, resulting from the solution given in Section 3.4.

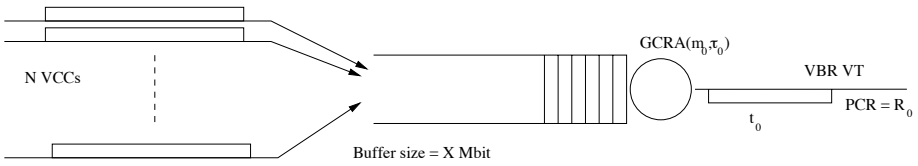


Figure B.1: CBR Node Reference Configuration

VBR-over-CBR approach

This is a modified peak rate allocation method that takes into account the shaping buffers of VPCs when available. The node reference configuration used in the simulation is shown in B.1. A multiplexer, fed with a number of input connections of VBR type, multiplexes them into one CBR virtual trunk, using a shaping buffer of the size B. The shaper guarantees that the buffer output conforms to GCRA$(1/R0, 0)$.

Denote by N the number of homogeneous VBR VCC connections (with peak rate R, sustainable cell rate m and maximum burst length t) sharing the VPC. The connection between the maximum burst length t and the burst tolerance τ is, as defined above:

$$t = \tau m/(R - m)$$

Denote further by C the bandwidth available for the VPCand by X the size of the shaping buffer connected to the VPC. The VT attributes are defined by:

- Trunk state $z = (N, m, \tau, R)$
- Connection descriptor: $y = (R_0)$

To obtain the the trunk state z, we assume that all the VBR sources are of the deterministic on-off type with active and idle periods of length t and τ, respectively. The worst case is that the active periods of the sources start at the same time. As a consequence, we have at time s

$$\text{WorstCase}(s) = \begin{cases} NR(s - k\tau) & s \in [k(t+\tau), k(t+\tau)+t) \\ NR(k+1)t & s \in [k(t+\tau)+t, (k+1)(t+\tau)) \end{cases}$$

Note that this is smaller than

$$\min\{NRs, Nm(\tau + s)\}$$

Otherwise, the service offered by the simple shaper at time s to the input flow is as follows:

Now it follows, from Equation 3.4, that

$$R_0 = \max\{\frac{NR - X}{t}, Nm\}$$

This is the minimum value of R_0 that guarantees no losses with a shaping buffer of size X. Finally we conclude that the two CAC functions needed in the bandwidth allocation are as follows:

$$\text{requiredBandwidth}(N, R, m, t, X) = \max\{\frac{NR - X}{t}, Nm\}$$

$$allowedNrCalls(C, R, m, t, X) = \max\{n \text{ such that } \max\{\frac{NR - X}{t}, Nm\} \leq C\}$$

Note that by omitting the shaping buffer ($X = 0$) we obtain the ordinary peak rate allocation method.

VBR-over-VBR approach

This is an advanced allocation method that takes into account both the shaping buffers of VPCs and the link buffers of physical links when available. In addition, the target cell loss probability is needed. The node reference configuration used in the simulation is shown in Figure 3.1. A multiplexer, fed with a number of input connections of the VBR type, multiplexes them into one VBR connection (the VBR trunk) using a buffer of size X. The shaper used in the simulations is not a buffered leaky bucket regulator but a simple shaper guaranteeing that the buffer output conforms to GCRA$(1/R_0, 0)$. However, due to the regulated nature of the input flow, it is possible to find parameters m0 and τ_0 such that the output conforms also to GCRA$(1/m_0, \tau_0)$.

Denote by δ the target cell loss probability. In addition, let B^l denote the size of the link buffer. In this case the VT attributes are defined by:

- Trunk state $z = (N, m, \tau, R)$
- Connection descriptor: $y = (m_0, \tau_0, R_0)$.

As above, to get the aggregate arrival curve α corresponding to the trunk state z, we assume that the VCC connections are of the same deterministic on-off type. Thus,

$$\text{WorstCase}(s) = \begin{cases} NR(s - k\tau) & s \in [k(t+\tau), k(t+\tau)+t) \\ NR(k+1)t & s \in [k(t+\tau)+t, (k+1)(t+\tau)) \end{cases}$$

Since we used a simple shaper plus a buffered leaky bucket regulator in the simulations, the service offered to the input traffic is as follows:

$$\min\{R_0 s, m_0(\tau_0 + s)\}$$

So, again, from Equation 3.4, we have that

$$R_0 = \max\{\frac{NR - X}{t}, Nm\}$$

Since we assumed that the service rate of the shaping buffer is R_0 and all the bursts of the underlying VCC connections start at the same time, lasting the maximum time t, the output from the shaper looks like another deterministic on-off source with sustainable rate m_0 and burst length t_0.

The triple (R_0, m_0, t_0) is further mapped to an equivalent capacity needed for the bandwidth allocation by using the function equivalentCapacity originally defined in [10] as in Equation 3.9

$$\text{equivalentCapacity}(R_0, m_0, t_0, X^l, \delta) = R_0 \frac{Y - X^l + \sqrt{(Y - X^l)^2 + 4XYm_0/R_0}}{2Y} \tag{B.3}$$

where

$$Y = \ln(\delta) t_0 (R_0 - m_0)$$

This is the rate necessary for achieving a desired buffer overflow probability δ on a given physical link, given a physical link buffer of size B^l and the traffic descriptor (R_0, m_0, t_0). From that we derive:

$$m_0 = Nm$$

$$t_0 = NRt/R_0$$

And

$$\tau_0 = t_0(R_0 - m_0)/m_0$$

Thus, the two CAC functions are in this case as follows:

$$\text{requiredBandwidth}(N, R, m, t, X, X^l, \delta) = \text{equivalentCapacity}(R_0(N), m_0(N), t_0(N), X^l, \delta)$$

$$\text{allowedNrCalls}(C, R, m, t, X) = \max\{n \text{ such that equivalentCapacity}(R_0(n), m_0(n), t_0(n), X^l, \delta) \leq C\}$$

Here $R_0(N)$ and $R_0(n)$ correspond to peak rates calculated from the previous formula by assuming that the number of active connections is N and n, respectively. The same is true also for the functions m_0 and t_0. Note that by omitting the link buffer ($B^l = 0$), we obtain the same CAC functions as in the CBR-over-VBR approach described above.

Appendix C

RM&R Simulation

This work in this chapter appeared in [46], [47] and [48].

Here we report the simulation performed for the RM&R based upon the VT solution and the dynamic resource management scheme presented in [11], [12], [13] and [14].

C.0.3 Simulation scenario

The simulation scenario consists of two overlaid networks, the physical network and the logical (VP) network. The underlying physical network is assumed to consist of nodes (ATM-switches), which are completely connected by identical physical links. The physical links are characterised by giving the capacity (bandwidth) of the link and the size of the link buffer. In the simplest case, there are three nodes connected together as a triangle, see Figure C.1.

All the traffic is modelled to arrive from regulated VBR sources. Each connection is assumed to be symmetric (with identical traffic parameters for forward and backward streams) belonging to one of the traffic classes. Each traffic class is characterised by its traffic descriptor (including PCR, SCR and BT) and the statistics characteristics (the mean holding time, the arrival rate of connection requests). The latter are used for the generation of traffic. The traffic sources are regulated so that they conform to GCRA(1/PCR,0) and GCRA(1/SCR,BT). The VCC connection requests are assumed to arrive according to a stationary Poisson process. Thus, no

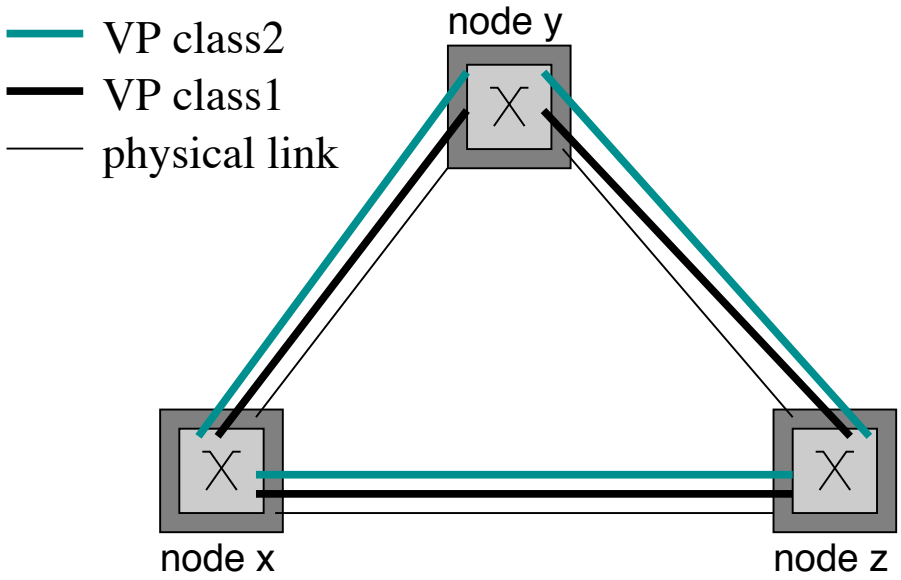


Figure C.1: Three nodes connected together as a triangle.

transient effects due to variations in the traffic load are taken into account. The holding times are sampled independently from an exponential distribution. In addition, the traffic pattern is thought to be even, i.e. the source and the destination of a VCC connection request are sampled from a uniform distribution.

Each traffic class is assumed to be served by its own logical network consisting of VPC links. Thus, we have taken the traffic separation approach. Also the logical networks are modelled to be completely connected. So, each VPC traverses through exactly one physical link, and each physical link conveys as many VPCs as there are different traffic classes. In Figure C.1, there are two traffic classes and thus, two logical triangle networks. The structure of the logical networks is assumed to be stable. Thus, no new VPCs are established nor any of the existing VPCs are torn down during the simulation.

The following two simulation trials were performed:

- Simulation Trial 1: the novel VBR-over-VBR approach was compared to the traditional VBR-over-CBR approach.

- Simulation Trial 2: the length of the updating interval was varied. Two different (albeit rather artificial) traffic classes were considered, one with a high

Parameter	*class 1*	*class 2*
peakCellRate	10.0	1.0
sustainableCellRate	2.0	0.2
burstTolerance	80	80
maxBurstLength	20	20
blockingThreshold	0.01	0.01
cellLossThreshold	0.00001	0.00001
meanHoldingTime	1	1

Figure C.2: **Table 1**: The two traffic classes used in the simulations.

bandwidth demand and the other with a low bandwidth demand. The constant parameters of the two classes are given in Table 1.

In particular, we see from the definitions below that the maximum burst length (with full cell rate) is 20 cell level time units[1] for both classes. During a burst of a connection belonging to class 1, cells may arrive at maximum rate 10 cells per cell level time unit, implying that the maximum burst size is 200 cells. For class 2 the corresponding values are 1 cell per cell level time unit and 20 cells. Note further that the mean holding time, which is the average length of a connection, is chosen to be 1 call level time unit [2] for both classes.

The network considered consists of three nodes connected together with identical physical links as a triangle. In fact, the network configuration is as already presented in Figure C.1. The capacity (bandwidth) of physical links is assumed to be 100 cells per cell level time unit in every case.

In the simulations we used the dynamic, periodic bandwidth allocation scheme by Mocci et. al. described earlier. In addition, all connection requests accepted were routed along the direct paths, which implies that, in fact, the results of the simulations are independent of the size of the network.

In each simulation trial, multiple simulation runs were performed with varying offered traffic loads. The traffic load of each class was taken to be equal. The following parameters were considered as a result of each simulation run:

[1]The cell level time unit can be chosen freely, e.g. a millisecond.

[2]Also, the call level time unit can be chosen freely, e.g. a minute. In particular, it does not need to be the same as the cell level time unit.

Parameter	*VBR-over-CBR*		*VBR-over-VBR*	
	no shaping	shaping	no shaping	shaping
shapingBuffer	0	200	0	200
linkBuffer	0	0	1000	1000

Figure C.3: **Table 2**: Parameters for the four alternatives in simulation trial 1.

- the percentage of average free capacity (i.e. the part of the capacity of physical links not allocated to VPCs) in the physical network,
- the percentage of rejected calls (from all calls offered) for each traffic class.

In next Section, where the results of the simulation runs are given, these parameters are presented as a function of the offered traffic load. By the traffic load we mean the ratio of the traffic offered (from all classes together) to a physical link and the capacity of a physical link (expressed in percents). Thus, if the offered traffic load is said to be 50, it means that, on the average, the traffic offered requires half of the capacity in each physical link. In these figures, the percentage of average free capacity is plotted in a normal linear scale, whereas the percentage of rejected calls is presented in a log-linear scale.

C.0.4 Simulation Trial 1: VBR-over-VBR vs. VBR-over-CBR

In this simulation trial the novel VBR-over-VBR approach was compared to the traditional VBR-over-CBR approach. In both approaches we further studied the effect of a shaping buffer. Thus we had four alternatives to compare. The parameters of these alternatives are given in Table 2.

Shaping buffers are assumed to be identical for all VPCs. Correspondingly, link buffers are assumed to be identical for all physical links. The buffer sizes are given in number of cells. Note that a shaping buffer of 200 cells can include 1 burst of class 1 or 10 bursts of class 2. Similarly, a link buffer of 1000 cells can include 5 bursts of class 1 or 50 bursts of class 2. In the simulations we used the dynamic,

periodic bandwidth allocation scheme by Mocci et. al. described earlier in section 4.1 with updating interval 1 call level time unit.

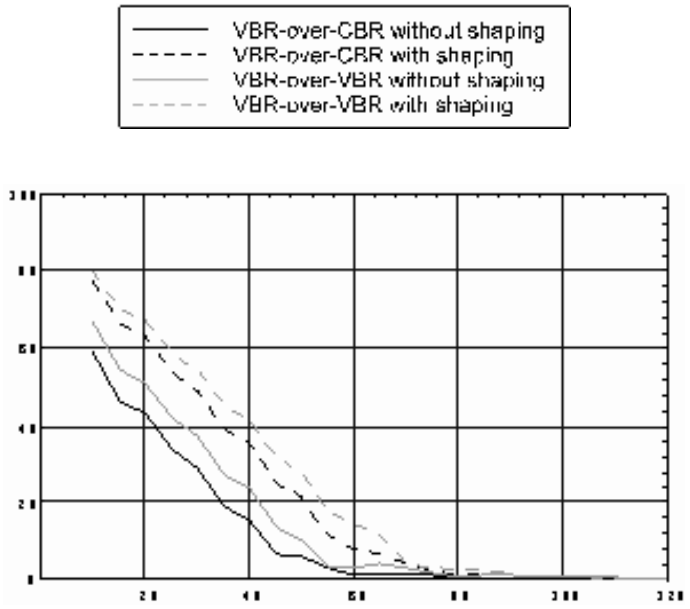
The results of the simulations are presented in Figure C.0.4. As expected, the VBR-over-VBR approach gives a better performance. However, the difference between the two approaches does not seem to be very significant. This is partly due to the rather inefficient method for calculating the effective bandwidth. By introducing more advanced methods, better results may be achieved by the VBR-over-VBR approach.

Whereas, by introducing shaping buffers it is possible to increase remarkably the performance of both approaches. However, this requires that the traffic shaped not be critical for delays.

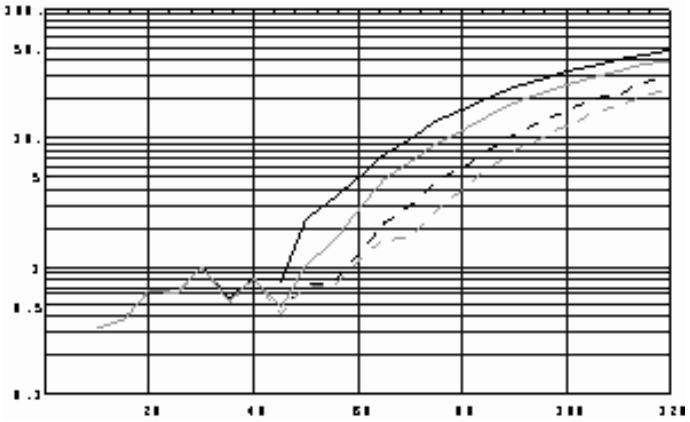
In addition, the simulations show that the dynamic bandwidth allocation method functions as expected. With a light or medium traffic load, the blocking probability is in the target area varying from 0.5% to 2%. The deviation from the exact target of 1 % is partly due to random variations, which could be diminished by having longer simulation runs. Note that the stability in the blocking probability is achieved by an increasing use of network resources: the percentage of the average free capacity falls from 100% down to 0% when the traffic load is increased. With a heavy traffic load, the blocking probability naturally grows because of the lack of network resources.

C.0.5 Simulation Trial 2: Varying updating interval of VPC capacities

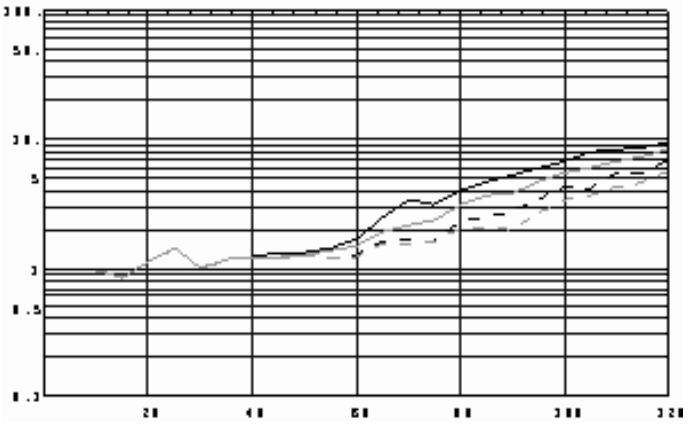
In this simulation trial, the length of the updating interval of VPC capacities, which relates to the dynamic, periodic bandwidth allocation scheme by Mocci et. al., was varied. The comparison was made between three different values of the length parameter (updatingInterval): 1.0, 0.5 and 0.1 call level time units. All connection requests accepted were routed along the direct paths. The results of the simulations are presented in Figure C.0.5. The results show clearly that the approximative method for the bandwidth allocation used in the simulations functions only if the updating interval is great enough (1 call level time unit or greater). With smaller values, the allocations are too small.



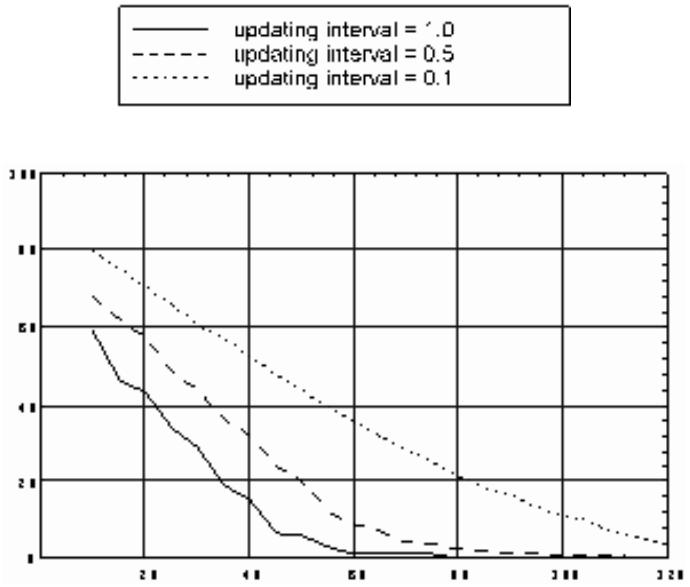
(b) Percentage of average free capacity vs. traffic load (both classes).



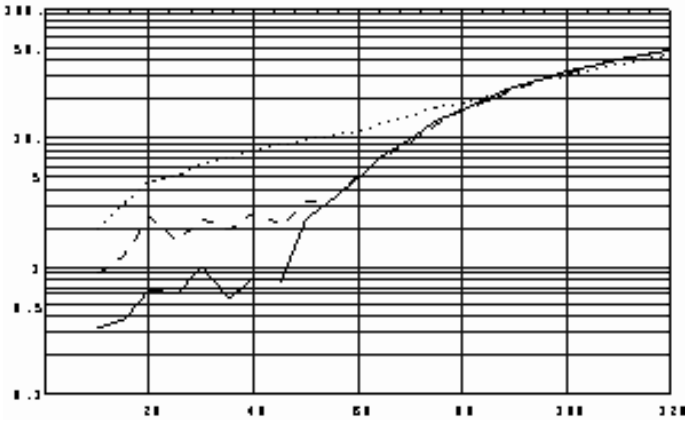
(c) Percentage of rejected calls vs. traffic load (class 1).



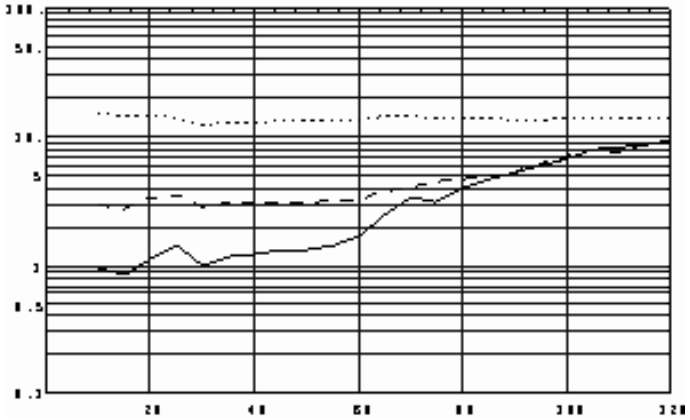
(d) Percentage of rejected calls vs. traffic load (class 2)



(f) Percentage of average free capacity vs. traffic load (both classes).



(g) Percentage of rejected calls vs. traffic load (class 1).



(h) Percentage of rejected calls vs. traffic load (class 2).

Appendix D

RM&R Trials

This work in this chapter appeared in [48] and [49].

Here we report the trials performed for the RM&R based upon the VT solution and the dynamic resource management scheme presented in [11], [12], [13] and [14].

This was integrated in the the EXPERT testbed [98], in order to support the trials. A major problem that was encountered was the lack of "open" and standardised control programming interfaces in the switches of the experimental testbed. Therefore, in order to establish and release the connections, several switch-specific modules had to be implemented. The heterogeneity of the testbed's equipment prevented the operation of the platform on a great number of actual switches. However, experiments with many switches were conducted in "simulation" mode in order to evaluate the scalability of the platform.

To our knowledge this is a unique example of an advanced resource management and routing architecture that was simulated and tested in a real ATM environment.

D.1 Trial Platform

The trial network configuration consists of 3 ATM switches and 6 links interconnecting them. Three different classes of service are assumed that the network supports. Different logic networks, constructed by the Network Elements (NEs) interconnected by VPCs specific to a class of service have been defined.

The platform architecture is shown in Figure D.1. The system is distributed and

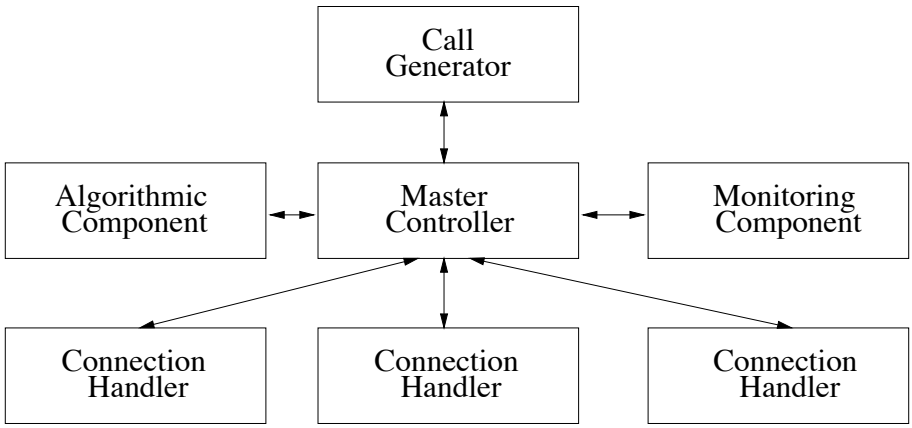


Figure D.1: The Trial Platform Architecture

consists of several components that may reside on the same or different machines. The communication between the modules is performed by means of TCP/IP sockets.

The system consists of the following modules:

- The Call Generator module (CG) emulates calls of a particular Class of Service (CoS) from a Terminal connected to an end-user side to the another Terminal with the same CoS that is connected on another end-user side. It is assumed that the end-user sides are connected directly to the switches of the network. The Call Generator is not aware of the architecture of the Network. The Call Generator has knowledge only for the end-user side and the number of Terminals that the end-user may have, as well as the CoSs that each terminal supports and also the call characteristics of these CoS (inter-arrival and holding times). The Call Generator generates calls for all terminals of all end-users simultaneously. The Master Controller (MC), is the centre of the system. In addition, the Master Controller has a better view of the network topology itself, concerning links, VPCs and CoS supported by every VPC.

- The Algorithmic Component is the algorithm that the platform tests (such as Resource Management (RM) and Routing).

- The Connection Handlers (CH) are responsible for establishing and releasing connections. The Connection Handlers reside on the sub-system controllers that control the NEs.

- The Monitoring Component polls the Master Controller and gets statistical information during the experiment. The Monitoring component can use this information for either representing at run time the results in a graphical way, or by saving the results to a log

The centre of the platform is the Master Controller Module. As the communication takes place at different machines and the components communicate to each other by means of TCP/IP sockets, every other component has to register with the Master Controller. By registering, every component reports its location (the IP address of the machine where it resides as well as the port where it will listen) to facilitate communications with the MC module. The Master Controller has all the knowledge of the network and can been seen, in this sense, as a configuration manager. Furthermore, the CAC components and the Route Selection Algorithm operate within this component. The other components of the platform communicate with the Master Controller (a) to test if the Master Controller is up and running, and (b) to permit to the Master Controller the co-ordination of the trial by providing the necessary information.

In Figure D.2, an example of how this communication takes place is shown. We used a single connection handler in order to simplify the example.

As evidenced, the CH, the CG and the RM register with the Master Controller. After that, the Call Generator generates a connection request. The MC generates a route for the call and then it contacts CH for a call-setup request. In this example, there is only one CH, but typically a number of CH's are involved. The CH makes the connection and notifies the MC of the connection id. When the CG generates a call-release for that call (each call is uniquely identified by a call ID, and this is used in the call release phase), it requests the MC to release this call, which in turn contacts the CH for the Call release. The CH replies back with an ACK or NACK message, which is propagated back to the CG. At any time throughout the experiment the monitoring component may request from the MC statistical information such as the number of Active calls, rejected calls, allocated bandwidth etc. All the above messages are implemented by means of well-defined TCP/IP messages between each module.

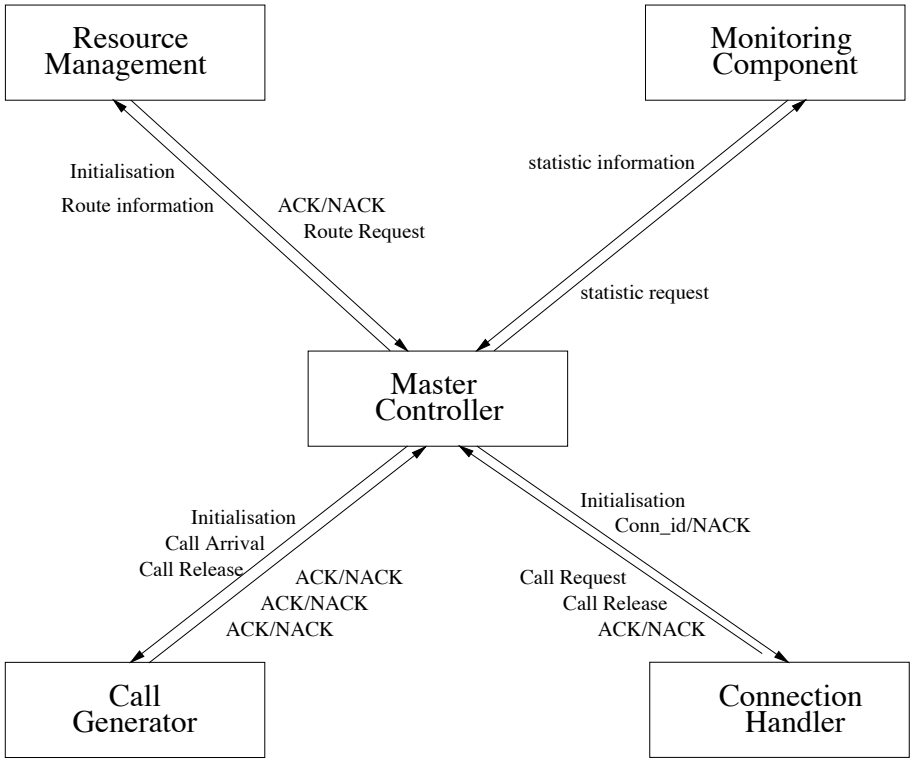


Figure D.2: Example of communication between the modules: the messages sent between the modules are numbered. The messages are numbered in the order in which they are generated.

We do not report here the complete internal architecture, but only the part of the Algorithmic Component relevant to the resource management, which is called the Resource Management Module.

D.2 The Resource Management Module

The Resource Management (RM) module covers the logical management of the VP network, following the scheme defined in Sections B.1.1 and B.1.2.

D.2.1 Network Configuration

The considered configuration consists of two networks:

- the physical network,
- the logical network

The physical network consists of nodes (ATM-switches) connected by unidirectional physical links. The physical links are characterised by link capacity (bandwidth) indicated by C and link buffer size, indicated by X. The logical network consists of logical nodes connected by unidirectional Virtual Path Connections. VPs are characterised by VP traffic parameters (traffic descriptor: PCR0, SCR0, BT0, and the VP shaping buffer B0), topology parameters (the underlying links), and parameters that describes the traffic multiplexed on this VP. Traffic is classified in a traffic class. Each traffic class is characterised by call traffic parameters (traffic descriptor: PCR, SCR and BT) and call statistics parameters (mean holding time and arrival rate). A separate logical network consisting of VPs serves each traffic class, thus each VP transports only homogeneous traffic. The structure of the logical networks is assumed to be stable and only the traffic parameters of VPs changes.

D.2.2 Bandwidth Reallocation

VP traffic parameters are reallocated at periodic intervals. We assume that a VP can be either a VBR or a CBR connection, even if the former is a CBR connection

with a logical VBR traffic descriptor parameters. The objective is to allocate for each VP sufficient bandwidth as needed to satisfy the stationary blocking probability target, which are class-specific. This is done by using the method, described in Section B.1, already used in the simulation that predicts the number of calls (N) in the next interval $T1$ (updatinginterval). This result is used to compute the corresponding bandwidth and, thus, the traffic parameters for each VP in the logical network. Finally, this information is sent to the MC module, which, in conjunction with the CHs, takes care of and physically manages the logical network.

D.2.3 RM Phases

The conceptual behaviour of the Resource Management module can be described by the following phases:

- Initialisation phase:

  1.1) receive external target parameters;

  1.2) contact the MC and receive the initial physical and logical configurations;

  1.3) build the operating configuration.

- Operational phase:

  2.1) wait for timer expires;

  2.2) contact the MC and receive the current logical configurations;

  2.3) compute the logical network reallocation and compare it with the old logical network configuration;

  2.4) send to the MC the new logical network configuration;

  2.5) go back to 2.1.

D.2.4 RM Pseudo-code

Initialisation phase: init()

PHASE 1.1) The `init()` process receives from external the socket information to communicate with the MC and the statistical target information: the updating time

($T1$) and the target blocking probability (ϵ).

PHASE 1.2) `init()` contacts MC (by using the `SendMessageToServerRM()`) to get the physical and logical configurations and receives them.

PHASE 1.3) These two configurations are kept in memory and accessed by two pointers `pt_PNCS` and `pt_LNCS`, that always point to the current configuration. Also the updating time and the target blocking probability are put into global variables: $T1$ and ϵ of type `integer` and `float`, respectively.

Thus, `init()` has the following structure:

```
int
init(fd, net, conf, s, name) /* phase 1.1 */
{
  SendMessageToServerRM (mc, msg); /* phase 1.2 */
    pt_PNCS=msg->physicalnet; /* phase 1.3 */
    pt_LNCS=msg->logicalnet;
}
```

After `init()` the RM has the current configuration:

```
struct PhysicalNetworkConfigStruct *pt_PNCS;
/* the pointer to the physical configuration (links) */
struct LogicalNetworkConfigStruct *pt_LNCS;
/* the pointer to the logical configuration (VPs) */
int T1; /* the updating time (secs) */
float epsilon /* the target blocking probability */
char *host /* the host name where the CM is running */
int portnum /* the port number used to communicate with CM */
```

Thereafter the code of the subroutine `Realloc()`:

```
int
Realloc(net, conf, VParr)
{
```

```
        for (i==0; i < nrVPs; i++)
        {
  for (j==0; j < linksofVP; j++)
  {
tmp_bw[k]=Max(tmp_bw[k],ComputeBW(tmp_N[nrVPs],R,m,t,X,X^1,losspb));
   };
};
        for (i==0; i < nrlinks; i++)
{
  for (j==0; j < VPsonlink; j++)
      {
reqbw[i]=reqbw[i]+tmp_bw[VPid(j)];
      }
  if (reqbw[i] > linkcap)
  nok[i]=1;
}
if (nok!=0)
   for (i==0; i < nrlinks; i++)
       {
       if (nok[i]==1)
  for (j==0; j < VPsonlink; j++)
      {
k=VPid(j);
availbw[k]=min(FairlyShareCapacity(reqbw[k],tmp_bw[k],
linkcap), availbw[k]);
      }
       }
    for (i==0; i < nrVPs; i++)
{
   k=VPid(i);
   R0=PCR(min(availbw[k],tmp_bw[k]));
   m0=SCR(min(availbw[k],tmp_bw[k]));
   t0=BurstLenght(min(availbw[k],tmp_bw[k]));
}
}
```

Operational phase

PHASE 2.1) After the initialisation phase, RM enters in the operational phase, where it reallocates the logical network (if needed) every $T1$ seconds. RM is in an infinite loop where it waits that the timer T1 (which lives for $T1$ seconds,

expires and then restarts) expires.

PHASE 2.2) Then sends a message to the MC (again by using the `SendMessageToServerRM()` and the `CreateUpdateRMRequestMessage()`)processes) to get the current network configuration. The current configuration substitutes in memory the old one, thus is accessed by pointer `pt_LNCS`, that always point to the current configuration.

PHASE 2.3) The current configuration is used to predict the next logical network configuration. The next logical configuration substitutes the current configuration.

PHASE 2.4) If current and next configuration differ, RM sends the next configuration to MC.

PHASE 2.5) It returns to the beginning of the loop. Thus, `db_update()` has the following structure (phase 2.1, 2.3, 2.4 and 2.5 are in `main()`):

```
db_update(s, mc, net, conf, name) /* phase 2.2 */
{
  msg=SendMessageToServerRM (mc, CreateUpdateRMRequestMessage());
  pt_LNCS=msg->logicalnet;
}
```

Thus, `rm()` has the following structure:

```
int
rm()
{
  fd=fopen("rm.conf","r");
  mc_addr=init(fd, net, conf, s, name);
  while(err==0)  /* phase 2.1 */
{
sleep(conf.timer);
err=db_update(s, mc, net, conf, name);
} /* phase 2.5 */
}
```

The communication between MC and RM, after the initialisation phase, requires always the transmission, from MC, of the logical network configuration at the time when the MC receives the request from RM. The remaining part of this protocol is acknowledge messages.

Realloc()

`realloc()` is used in phase 2.3 to predict the next logical configuration and to check whether the physical configuration (i.e the links capacity) is sufficient to support it. It is divided in two steps:

Step1) given the current logical network, for each VP is computed the function `transientErlangRequirement`$(n, \rho, \epsilon/2, T1/T)$, where n=(`active_calls` + `rejected_calls_in_last_interval`) on this VP, T is the holding time of the class of call supported by this VP and $\rho = \lambda * T$, with λ being the arrival rate of the class of call supported by this VP. This predicts the maximum number N of connections that the VP should support in the next interval.

Step2) From this value , by using the same algorithm used by CAC, it is computed the requiredBandwidth C_vp needed to support N connections. The sum of the C_vp for all VPs sharing a same link must be smaller than or equal to the capacity of the link itself. If this is true, next logical configuration is updated with the values of C_vp for all VPs, otherwise the available capacity on the link is fairly shared by all the VPs.

D.2.5 RM Structures

RM uses 4 structures for the data:

Struct1) The Physical Network Configuration (PhysicalNetworkConfigStruct) is implemented as an array of links LinkLineStruct.

```
typedef struct LinkLineStruct
       {
int linkid; /* unique link identifier */
```

```
int capacity; /* link capacity (fixed) */
int physicalbuffer; /* link physical buffer capacity */
float losspb; /* wished loss probability on that link */
int numberofVPonlink; dimension of next structure */
int *listofVPonlink;  /* list of VPs ids on this link */
int SourcePort; /* The link is attached to a Source Switch Port */
int DestPort; /* The link is attached to a Dest Switch Port */
 } LinkLine;
```

Struct2) The Logical Network Configuration (LogicalNetworkConfigStruct) is seen as an array of virtual paths VPLineStruct.

```
typedef struct VPLineStruct
       {
int VPid; /* unique VP identifier */
int VPtype; /* 0 for CBR, 1 for VBR */
int classid; /* Call class supported by that VP */
int activecall; /* number of call active on that VP */
int rejectedcall; /* number of call active on that VP */
int PCR0; /* VP Peak Cell Rate */
int SCR0; /* VP Sustanaible Cell Rate (0 if CBR)*/
float bursttolerance0; /* VP Burst Tolerance (0 if CBR)*/
float burstlength0; /* VP Burst Length */
int VPcapacity; /* VP allocated capacity (requiredBandwidth)*/
int VPbuffeR; /* VP shaping buffer */
int VPListDim;  /* Number of element of the following list */
int *listoflinkusedbyVP;/* list of links used by that VP */
int i_SourceNode  /* number of the source node */
int i_SourcePort; /* number of the source port */
int i_DestNode; /* number of the dest node */
int i_DestPort; /* number of the destination port */
int i_LastAllocatedVCI; /* number of the last allocated VCI */
float arrivalrate; /* call arrival rate */
} VPLine;
```

Struct3) The set of Call Class of Service (CallClassStruct) is an array of class of service ClassLineStruct.

```
typedef struct ClassLineStruct
       {
int classid; /* Call class identifier */
int PCR; /* Call Peak Cell Rate */
int SCR; /* Call Sustanaible Cell Rate */
float bursttolerance; /* Call Burst Tolerance */
float arrivalrate; /* Statistical Call Arrival Rate */
int holdingtime; /* Statistical Call Holding Time */
       }
```

Struct4) The Network Configuration (NetConf) is a record of all of them.

```
typedef struct NetConf{
    NetDim *dim;
    LinkLine **PhysNet;
    VPLine **LogNet;
    ClassLine **Class;
}
```

D.3 Trial Results

The trial configuration used for the purpose of collecting the presented results, is a simple one. This was a deliberate choice motivated from the major objectives of the trials. Specifically, these trials aimed at:

- demonstrating the use and the benefits of the Resource Management scheme,
- evaluating known effects of the scheme, in a realistic environment,
- confirming the architecture design and the simulation results,
- comparing bandwidth allocation with static bandwidth allocation and network planning, regarding their overall effect on the network utilisation.

The trial configuration, consists of two switching nodes connected with a 100Mbit/s cell based link. There are two distinct logical configuration instances.

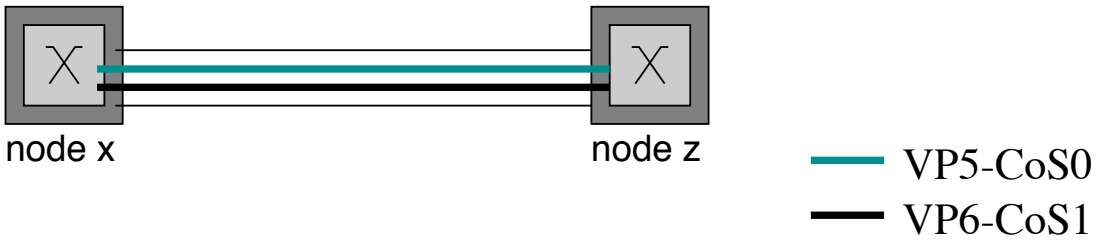


Figure D.3: The Logical Configuration with two classes

Traffic Class	*PCR (Mbits/sec)*	*SCR (Mbits/sec)*	*Burst Tolerance*	*Holding Time (sec)*
CoS0	10	5	2.0	30
CoS1	2	1	2.0	30

Figure D.4: **Table 3**: Traffic Classes initially defined for the resource management Trial.

During the first instance, which is depicted in Figure D.3 two traffic classes were considered between the two switches, namely CoS0 and CoS1. The characteristics of these two classes are shown in Table 3. CoS0 has a peak cell rate (PCR) of 10Mbits/s, whereas CoS1 features a 2Mbits/s PCR. These parameters are crucial for the VBR-over-CBR CAC. Two VPs were set up between the switches. VP5 was dedicated to serving CoS0 calls, while VP6 was exploited by CoS1. Having established this trial context, the trial ran several times, with different inter-arrival times for each one of the two traffic classes. Thus, the inter-arrival times for each one of the two traffic classes constituted variables that imposed different loads on the network during different repetitions of the trial. This allowed the evaluation of the scheme under different loads, through corresponding measurements.

D.3.1 Effectiveness of the Dynamic RM

The initial objective of the trials was to verify the appropriate function of the periodic bandwidth allocation scheme by proving that the trial results are in accordance to the analytical description of the scheme. As already stated, this algorithmic scheme attempts to allocate for each VPC sufficient bandwidth as is needed to satisfy the stationary blocking probability target. This target probability may be different

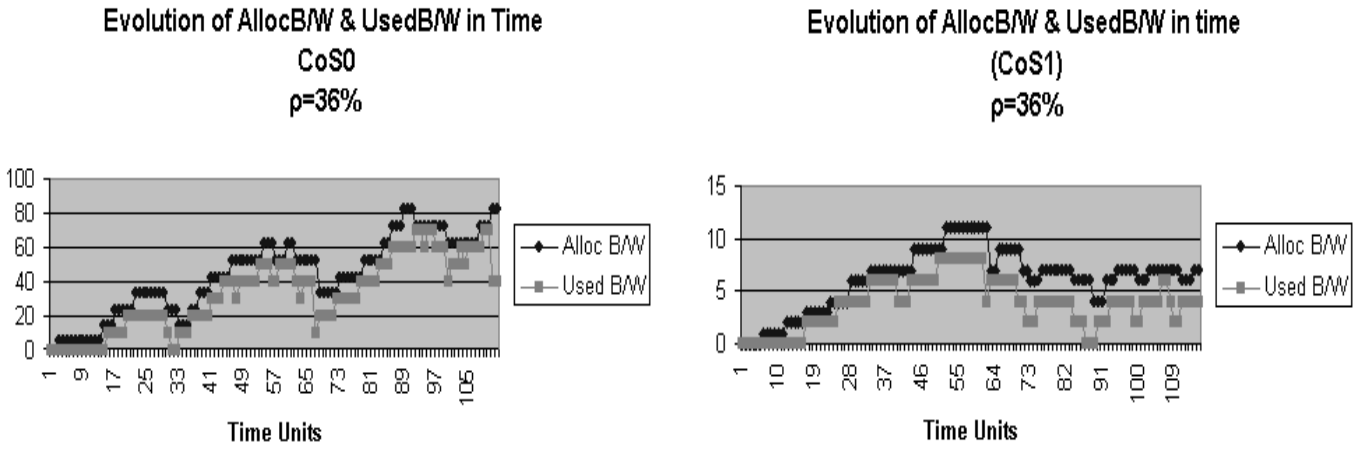


Figure D.5: Network load 36%.

for each defined traffic class. In order to indicate that this basic concept is met in the implementation of the algorithm, the previously described trial configuration was exploited. The resource management algorithm was activated and the bandwidth of each VPC was updated periodically (with a period equal to 8 seconds). The call blocking probability figure, which was set as a target was 5%. With these parameters and the characteristics of the two traffic classes (as described in Table 3), the trial ran 6 times. Each run featured different inter-arrival times for each class. Thus, each trial repetition imposed a different load on the network. The parameters were selected with a view to attaining loads ranging from a light 36% to a heavy 120% load that rendered the system non-ergodic. The theoretic function of the algorithm was verified as the trials exposed the algorithm's effort to conform to the demand of every class with the overall goal of minimising the probability for call rejections. This fact is depicted in the following diagrams that show the bandwidth demand (Used B/W) and the corresponding bandwidth (AllocB/W) assigned by the bandwidth management algorithm. Each of the following diagrams corresponds to a specific traffic class. Furthermore, the fact that the algorithm tries to keep up with the classes' demand is verified for the whole range of traffic loads. Figures D.5, D.6 and D.7 illustrate the trials results for different network load values.

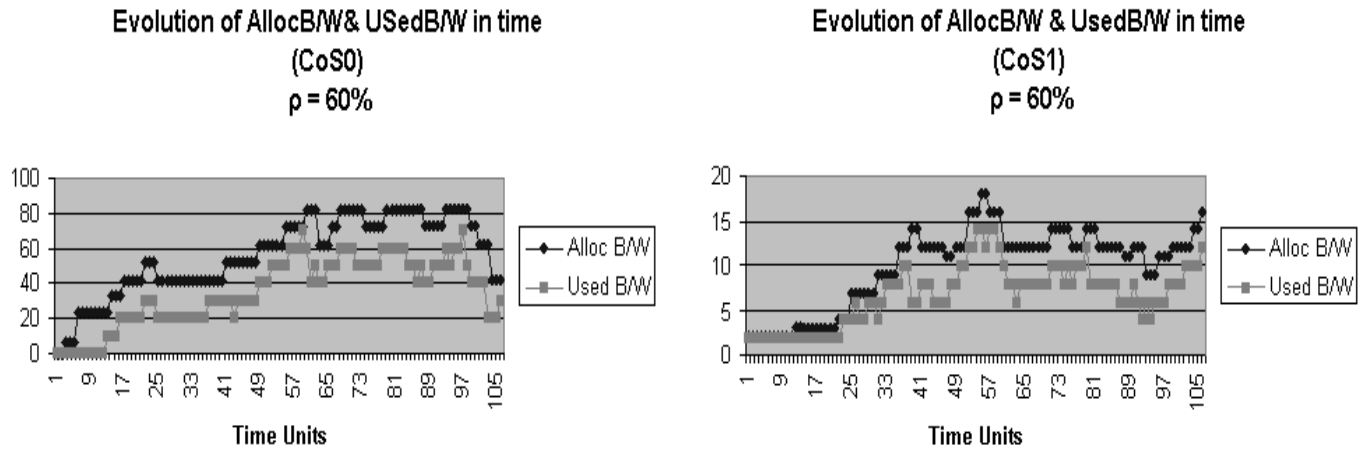


Figure D.6: Network load 60%.

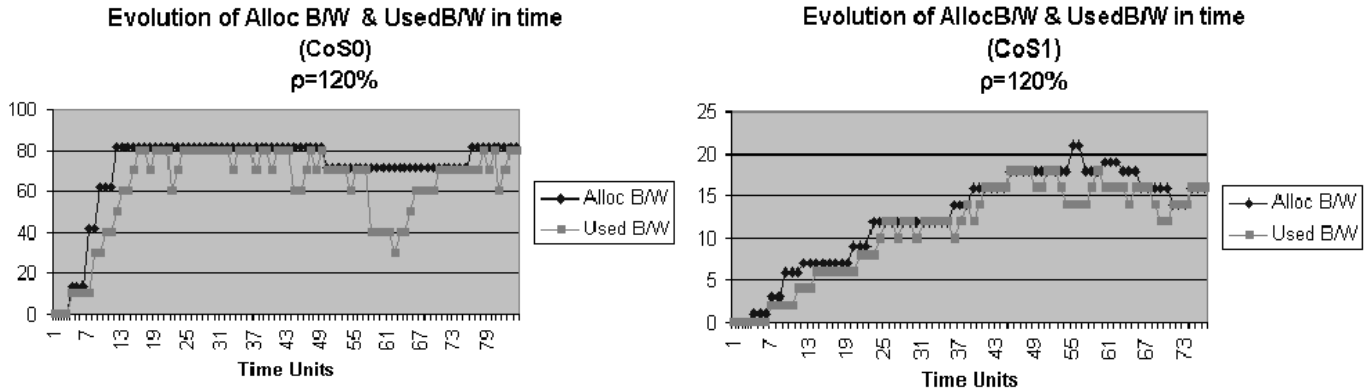


Figure D.7: Network load 120%.

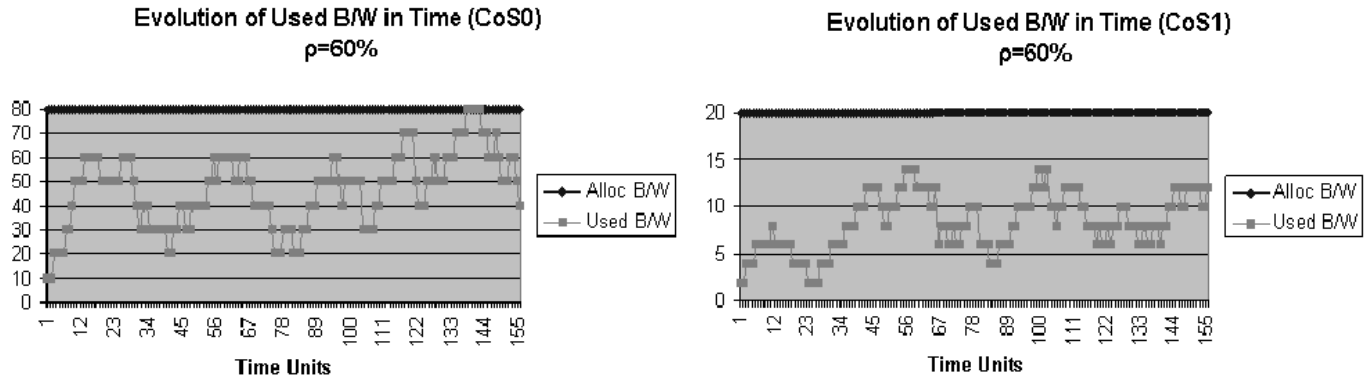


Figure D.8: Network load 60%.

D.3.2 Benefits of the Dynamic RM

Then we focused on evaluating the benefits of the resource management strategy. Therefore, the same trials were performed, with the bandwidth allocation algorithm deactivated. The deactivation of the bandwidth allocation algorithm was based on the update interval. Specifically, an interval that exceeded the duration of all trials was defined. As a result, no bandwidth reallocation occurred during any of these experiments.

We deactivated the allocation of bandwidth to each class. In particular, traffic assigned an amount of bandwidth on the link, which was in proportion to its predicted needs. The predictions were based on the call characteristics of each class (i.e inter-arrival times and holding times), as well as on the PCR for the class. This is the parameter that is used by the CAC algorithm in the CBR-over-VBR approach. Moreover, the static allocation assigned the full capacity of the link exclusively to the two classes. The rationale behind this approach was an attempt to achieve the lowest possible call blocking ratio. However, it eliminates any free capacity on the link. The following diagrams illustrate results obtained during these trials, for network load equal to 60%:

D.4 Results analysis

The trials on Resource Management, have demonstrated the advantages of bandwidth reallocation over approaches that involve static allocation. A direct conclusion, is that the selected resource management scheme features an acceptable responsiveness. Specifically, it was clearly seen that the algorithm maintains a low call blocking ratio, even in cases where bursts of call arrivals occur. It is noteworthy that the algorithm managed to respond to the demand of the various CoSs in conditions of high load. However, the call acceptance drops dramatically once the system is overloaded (load $> 100\%$), but this is an expected result. The efficiency of the resource manager in allocating the required resources was also demonstrated. Results showed that the resource management algorithm produced similar call blocking figures with a static assignment scheme with prediction, while at the same time making more efficient use of the network resources.

Similar trials were held with three traffic classes and demonstrated that the introduction of a third class does not cause any serious deviation from the two classes example, as far as the function of the bandwidth allocation algorithm and its effect on network utilisation is concerned. The static allocation of bandwidth, which is based on the prediction of the load imposed by each class, still produces satisfactory results. Furthermore, the dynamic bandwidth allocation algorithm does not have any difficulty responding to the fluctuations in the overall link utilisation imposed by the third class. Moreover, the needs for bandwidth are satisfied through bandwidth reallocation that "saves" link capacity, to be used for other demands.

Appendix E

IP and ATM - current evolution for integrated services

This work in this chapter appeared in [34], [99] and [100].

This appendix gives a technical overview on the competing integrated services network solutions, such as IP, ATM and the different available and emerging technologies on how to run IP over ATM, and attempts to identify their potential and shortcomings.

E.1 Introduction

For many years, ATM based Broadband-ISDN has generally been regarded as the ultimate networking technology that could integrate voice, data, and video services and was suitable for LANs and WANs, both private and public.

ATM has been around for several years and, contrary to the expectations, it was not the foreseen success in in the public WAN area. Although ATM products are broadly available today, public network operators hesitate with the deployment of public ATM based networks. In contrast to this, ATM had quite an impact in the private LAN area, where ATM is mainly deployed as a high-speed backbone network interconnecting legacy LAN equipment, driven by the need to increase transmission speed.

With the recent tremendous growth of the Internet, the future role of ATM seems

to be less clear than it used to be. WWW based use of multimedia applications on the Internet is widespread. By offering not only typical data services but even real time voice and video applications (though with poor quality), the Internet is entering the typical target market of ATM at service level. Furthermore the Internet Society is quite drastically loosening their policy of shared resources and free usage and deeply investigating on how to introduce resource reservation and charging support in the Internet to provide better support for multimedia applications and service providers.

The dominance of IP based networks in the WAN and LAN area has also led to proposals for ATM deployment that considerably differ from the traditional view of public telecom operators, such as using ATM only as a high speed transmission system.

The discussions about whether IP or ATM is the better technology for an integrated services network are ongoing and reached almost the state of a 'war' between advocates of the two technologies.

This appendix gives a technical overview of the different technologies today and makes a neutral assessment of their feasibility for an integrated services network. In order to be able to compare different solutions, we establish the requirements for an integrated services network in section 2. These requirements depend on the perspectives of different actors and contain more than just the requirement for quality of service (QoS). Then we focus on the main competitors, the Internet protocols (section 3) and the ATM technology (section 4). In section 5 we discuss several of today's and tomorrow's solutions for IP support on top of ATM networks. Of each presented technology the advantages and disadvantages are assessed in the corresponding section. Section 6 compares the technologies and tries to guess about their applicability and the role they are going to play in the future.

E.2 Integrated Services Networking Requirements

In this section, some of the most important requirements are listed for the comparison of integrated services networking technologies. Criteria for a broad accep-

tance of such a technology are established. The requirements can be categorised as follows:

- Requirements from the user's perspective
- Requirements from the service provider's perspective
- Requirements from the network provider's perspective

Note that we do not list them in a priority order, even if users satisfaction was often the key of the success of most of the winner technologies.

E.2.1 User's Perspective

Because the acceptance and success of a technology is dependent on user requirements, it is essential that these requirements are met by the technology. A user's major concerns are performance, ease of use, cost, universal availability and security of services:

- Guaranteed support of an appropriate minimal performance: Each service should be offered with the appropriate minimal performance. It is important to note here that from the user's perspective not all services have to be offered with very high performance. However there are considerable differences in performance requirements depending on the kind of service, and on its pricing. An audio phone service with today's POTS performance imposes high quality requirements on a networking technology, whereas an e-mail service could be acceptable with as basic a quality requirement as reliable delivery. In order to be generally acceptable to the user, services using a new networking technology should be offered with equal or better performance and at the same or even lower price than those commonly available today. A Cost-Performance compromise has to be taken into account and it can be argued that the final decision for such a compromise ideally should be delegated to the user. This requires some degree of Cost-Performance transparency.

- Ease of Use (and configuration) is important and the availability of simple and cheap terminal equipment is a must. This naturally calls for Integration of

services: An acceptable networking technology should integrate the full range of services to satisfy all of today's and tomorrow's communication needs.

- Universal connectivity is a growing requirement. Users would like to have the possibility of reaching any other user over the same access technology. Flexibility features such as personal mobility should be supported as users wish to have access to their subscribed services from any terminal equipment.

- Security: An acceptable networking technology should support security features such as authentication and privacy. Authentication limits service access to authorised users only, e.g. eliminating the risk of users accessing a service without paying for it. Privacy means encryption of data so those eavesdroppers are not able to interpret the received data, allowing for example credit card numbers to be exchanged over the network.

E.2.2 Service Provider's Perspective

With deregulation in the public telecommunication market and the transition towards integrated services networks, it is expected that there will be a clear functional separation between network providers, who operate network infrastructure and provide network connectivity, and service providers, who provide services on top of that network infrastructure. In today's public networks, both network provisioning and service provisioning is typically under the control of the same legal entity (e.g. PNO), but in the future we expect a high number of independent service providers to enter the market, who will run their business separately from network providers [101]. The requirements for an integrated services networking technology are not identical for service providers and network providers. This section lists the requirements from the service provider's perspective. Service provider's requirements also include requirements imposed by the provided services themselves. It has to be noted here that a "service" in this context is not restricted to new multimedia services.

Service provider requirements are mainly in the area of universal connectivity and traffic support, network and service separation support, service management

and security:

- Universal connectivity and universal traffic support are essential to maximise the market size for a service. An integrated services networking technology should be able to cope with any traffic type, as traffic characteristics produced from different services vary considerably. This calls for high bandwidth availability so that service providers are able to offer any kind of service. It also requires end-to-end transmission quality guarantee in order to be able to provide services with the user-requested performance. Addressing flexibility is needed, as many services require more than just normal unicast (point-to-point) addressing. The networking technology should offer the flexibility to support the full range of addressing types such as unicast, multicast, broadcast and anycast.

- Service charging support: In an integrated services network there should be support for service charging as service providers are expected to prefer to charge for their services independent from network providers.

- Security: In addition to the security related requirements from the user's perspective (i.e. authentication and privacy), service providers require support of non-repudiation. Non-repudiation means that once a user has committed to pay for a service, the payment can not be refused.

- Network and service separation support and ease of service management are not taken into account in this chapter, as these issues are not primarily dependent on the underlying network technology.

E.2.3 Network Provider's Perspective

Even though deregulation in the telecommunication market will allow for new network providers, the group of network providers is the smallest group of actors in an integrated services network. Nevertheless their requirements must not be neglected because they build, operate and own the networks. The main focus of network providers is on manageability, network availability, scalability, chargeability and (of course) low costs:

- Scalability: A future-proof networking technology should be able to scale to an unlimited number of endpoints and to ever increasing resource demands.

- Support of network charging: Charging of network usage is an essential requirement of network providers. In integrated services networks it will not be sufficient to have a flat-rate usage charging but rather a usage charging based on traffic size and quality. Without this traffic based charging, network overload situations will become the norm. Traffic based charging can only be achieved if a networking technology provides the functionality to monitor the traffic.

- Low cost: The infrastructure as well as the operating costs for a networking technology should be low. Protection of investment is an important factor. Given the enormous investments in existing networking infrastructure (e.g. POTS, Cable TV), the ability to be run on top of parts of this existing infrastructure is very essential for a new networking technology in the opinion of network operators. This holds especially true in the customer access area where investments for physical connections are huge. Furthermore a new networking technology should allow for a smooth migration, making use of large parts of existing telecommunication infrastructure for the short term and allow for successive replacement with new infrastructure for the medium and long term.

- Network management support and network availability are not taken into account in this chapter, as these issues are not primarily dependent on the underlying network technology.

E.3 Internet Technology

E.3.1 IPv4

General Overview

The Internet Protocol (IP, or IPv4) is the central part of the Internet protocol suite. IP (RFC 791 [102], RFC 1122 [103]) offers a connectionless packet delivery service on top of which the transport level protocols i.e. TCP and UDP build their functionality. IP is a datagram oriented protocol that treats each packet independently. Therefore each packet must contain complete addressing information. It neither guarantees delivery nor integrity, because the protocol does not use checksums to protect the content of the packet and there is no acknowledgement mechanism to determine whether the packet has reached its destination or not.

The IP protocol together with a set of supporting protocols (ARP, RARP or BootP, ICMP) defines the format of the Internet datagram, addressing, address resolution, packet processing, routing, and error reporting mechanisms. As described in RFC 1122 any host running the IP protocol suite typically also supports the following protocols: Address Resolution Protocol (ARP, RFC 826 [104]) and Internet Control Message Protocol (ICMP, RFC 1122 [103]). The following figure summarises how the IP protocol is related to the other protocols in the Internet stack. As shown in the figure, IP can be run over a variety of data link layers, because IP hides the underlying technologies from its users.

The IP datagram structure

The IPv4 datagram is variable in length with a theoretical maximum of 65'535 octets. However, in practice the size of a datagram is limited by the size of the data link layer or the physical layer as a whole datagram has to fit into a single frame of the underlying layer. For example, Ethernet limits the datagram sizes to 1500 octets. This limitation to the datagram size imposed by the underlying technology is called the "maximum transfer unit", MTU. However, in a heterogeneous environment with varying MTUs, the datagram may need to be fragmented into smaller pieces, IP therefore supports fragmentation.

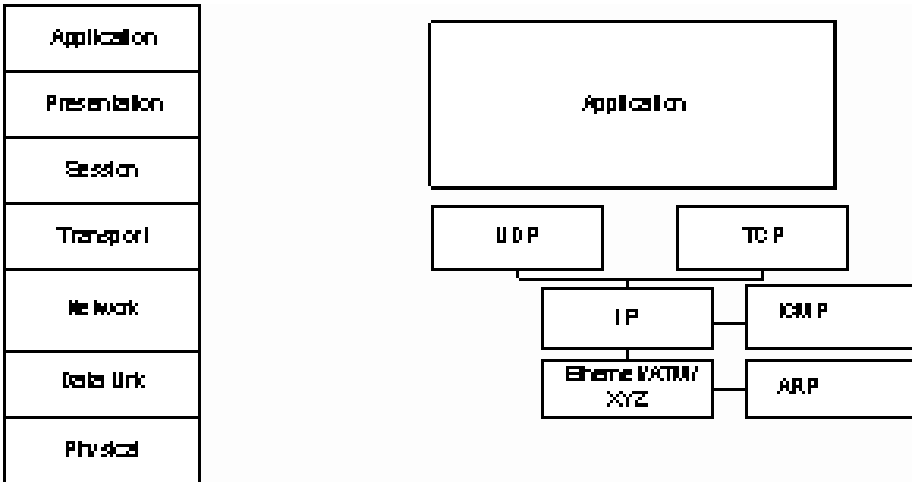


Figure E.1: IP in the Protocol Stack.

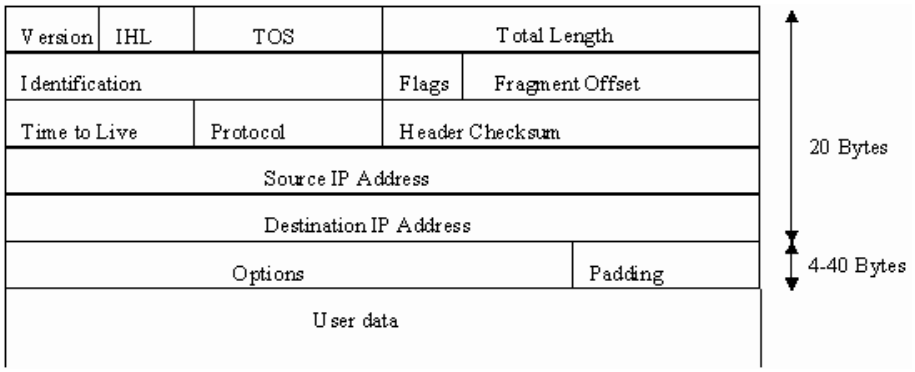


Figure E.2: IP datagram format.

The coding of an IP datagram format is shown in the following figure.

- Version: identifies the protocol version (i.e. 4 for IPv4)
- Internet Header Length (IHL): the length of the header in 32 bit words
- Type of Service (TOS): indicates possible priority and the type of transport the datagram desires (options are low delay, throughput, reliability)
- Total Length: the length of the datagram measured in octets up to 65'535 octets

- Identification: a value assigned by the sender to aid in assembling the fragments of a datagram
- Flag bits: control the fragmentation (e.g. don't fragment, may fragment...)
- Fragment offset: indicates the place in the original datagram where this fragment belongs
- Time to Live: indicates the maximum hop number that the datagram is allowed to pass in the network
- Protocol: indicates the next layer protocol that was used to create the user data (e.g. TCP, UDP)
- Header Checksum: a 16 bit checksum over the header
- Source/Destination address: 32 bit IP addresses to identify the sender and receiver
- Options field: carry information for network control, debugging, routing and measurements.

IP Addressing and Routing

In IPv4 the IP address space is limited to 32 bits. An address begins with a network number used for routing, followed by a local, network internal address. IP addresses are classified in four classes according to the size of the network portion of the address:

- Class A, where the high order bit is zero, the next 7 bits are for the network, and the rest for the local address
- Class B, the high order two bits are one-zero, the next 14 bits are for the network and the rest for the local address
- Class C, the high order three bits are one-one-zero, the next 21 bits are for the network and the last 8 for the local address.

- Class D, this is for multicasting, the high order four bits are one-one-one-zero followed by multicasting address

As can be seen from the above address classes IP supports multicasting (in subnetworks). Broadcasting is also supported (RFC 919 [105]).

The IP addressing builds upon the notion of "network", which is fundamental for routing in the Internet. There are two types of equipment at the IP level, hosts and routers. Hosts are any end-user computer system that connect to a network. Hosts know (or learn during the boot phase) their network address and local address. This forms the host address. A physical host may have several local addresses and a single network address. A multi-homed host is a host that is attached to two different networks as a host, however this is a special case. A router is a (dedicated) computer that attaches to two or more networks and forwards packets from one to the other based on the network portion of the destination IP address. Routers exchange network addresses as reachability information between them using various routing protocols (e.g. EGP, OSPF) depending on where in the network hierarchy the routers are located.

IP traffic within the same network can be delivered directly from host to host, whereas IP traffic to another network always passes one or several routers.

Assessment

The roots of IP are in the early '80s. Since then processing power and memory size of computers and the nature of applications have changed considerably. IPv4 has the following restrictions, some of which have led to the recent redesign of the IP protocol (IPv6, see section 3.2):

- The fixed size address space of 32 bits is a limiting factor for the predicted Internet growth (B class addresses exhausted, supernetting of C class addresses is only a short term solution).

- New types of address hierarchy are needed to make the protocol more flexible.

- No support of an anycast addressing concept.

- Per packet computational load is not optimal and can be improved, resulting in a more efficient datagram delivery.
- No support of multimedia type of streaming (flows).
- No support of guaranteed QoS: just plain, best effort, connectionless packet delivery.
- Potentially inefficient routing (all IP packets of a persistent data flow are routed independently)
- Potentially inefficient transmission (IP header is too big, especially for short packages)

If only the mandatory set of the IPv4 protocol suite (as described in [103]) is supported, the following restrictions also apply:

- No plug and play type of address autoconfiguration and re-numbering.
- No network layer security support.
- No mobility support.

However, there are additional RFCs, which cover these features.

Despite these restrictions IP is widely deployed today and with the current boom of the Internet it will become even more important among the network layer protocols. Apart from its wide deployment offering almost universal connectivity, there are some other advantages of IP:

- There is an unmatched variety of services and applications available that build on IP
- IP can be run over a big variety of physical layers
- IP is a working solution and its performance has been well tuned over the years
- IP equipment is cheap for network providers as well as for users

- IP based applications do not have to know their bandwidth demand in advance and can easily adapt to the encountered traffic level along the traversed network path.
- Separating network and service provisioning is a reality in the Internet architecture

E.3.2 IPv6

Why IPv6

Mainly triggered by the fear of the approaching address space exhaustion and to solve some of the shortcomings of IPv4 (see section 3.1), the IETF started working on IPv6 (or IPng) in 1992. By 1996 version 6 of the Internet Protocol was specified.

IPv6 is not a radical change to IPv4, it is rather an evolutionary step and coexistence between Ipv4 and Ipv6 is possible for a transition phase [106]. Except for the larger address space and some autoconfiguration features, all new functionality could also have been fitted into IPv4. Nevertheless, after over 10 years of building and enhancing the Internet protocol stack, it is necessary to clean and consolidate the functionality of the very central IP layer and make it a ready platform which will be able to cope with new Internet functionality required in the near future.

The IPv6 protocol suite

The IPv6 protocol suite is not defined in a single specification but comes in a whole collection of RFCs, the most important of which are listed below:

- IPv6 (RFC1883 [107]), IPv6 Addressing (RFC 1884 [108])
- ICMPv6 (RFC 1885 [109]) Internet Control Message Protocol, including address resolution
- Authentication Extension (RFC 1826 [110]), ESP Extension (RFC 1827 [111])

All higher layer protocols in hosts (UDP, TCP, Web, DNS...) need to be enhanced to be able to use the new functionality of IPv6. There are Internet drafts

available on how to enhance the IPv4 API (socket interface) in order to bring the IPv6 functionality to the application layer.

New addressing and routing

IPv6 uses 16 byte addresses and improves addressing flexibility through the definition of unicast, multicast and anycast addresses. Furthermore IPv6 supports plug and play features such as automatic IP address configuration and re-numbering.

Scalability was introduced to IPv6 multicast routing by using address scopes. In general, IPv6 routing is almost identical to CIDR of IPv4, based on the route selection of longest matching address prefix. With very little modification, all of IPv4's routing mechanisms can be used to route IPv6.

Source routing is used in IPv6 to ease future implementation of new functionality such as terminal mobility and provider selection.

IPv6 packet structure

The IPv6 packet's base header is a streamlined IPv4 header, reducing the processing cost of packet handling and limiting the packet size by removing some of the fields and options. Some of the options removed from the old header and some new options of IPv6 are now supported through an arbitrary number of extension headers following the base header, each of them indicating in its Next Header field the type of the next following extension header. Examples for such extension headers are the source routing extension header, the fragmentation header and the authentication header. Extension headers are normally only examined or processed by the destination node. The use of extension headers introduces high modularity in the IP packets and easily allows future options or extensions to be integrated. The data follows the last extension header. The structure of the IPv6 base header and of an IPv6 packet is given in Figure E.3.

New features of IPv6

IPv6 introduces flow labelling capabilities (Flow Label field in base header), which allows packets belonging to the same flow to be labelled. The sender can

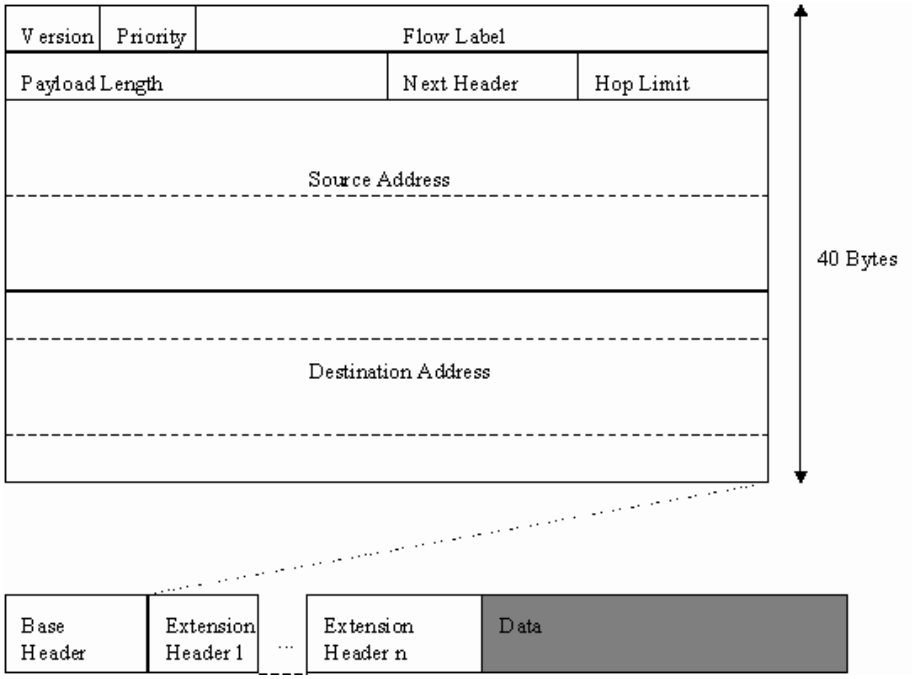


Figure E.3: the IPv6 packet structure.

request special handling of a flow by the routers, such as non-default quality of service or "real-time" service. Routers can cache flow information obtained from processing the first packet of a flow and thus speed up processing for following packets of the same flow.

The Priority field in the base header allows the desired delivery priority of a packet to be specified, but this is only relative to the priority of other packets from the same source.

IPv6 introduces network layer security defined in the authentication and the ESP extension header. Authentication is used to guarantee the packet sender's identity and ESP means the encapsulation of security payload so that a third party can not read it.

IPv6 supports source routing capabilities defined in another extension header.

IPv6 allows only restricted fragmentation, i.e. fragmentation is only allowed at the packet source but NOT at intermediate IPv6 routers. This is achieved by either using the minimum MTU guaranteed by all delivery systems (576 octets) or by using ICMPv6 messages for path MTU discovery.

Transition from IPv4 to IPv6

IPv6 and IPv4 are similar but all the same are distinct protocols. To allow for an incremental upgrade of IPv4 equipment to IPv6, during which both protocols can coexist, it is crucial that there is both a way to interwork between the two protocols and to tunnel IPv6 through a cloud of IPv4. Given today's vast installed base of IPv4 equipment this was a key issue during IPv6 protocol design.

To address the transition problem, there is RFC 1933[106], which defines a set of mechanisms that IPv6 hosts and routers should implement in order to be compatible with IPv4 hosts and routers. The proposed solution is based on dual IP layer nodes, tunnelling, and DNS support.

- Dual IP layer node is a host or router that implements both IPv6 and IPv4 All such nodes need an IPv6 and an IPv4 address. For this purpose the "IPv4 compatible IPv6 address" was defined, which uses the IPv4 address in its lower 4 bytes and all zeros for the 12 higher bytes.

- Tunnelling defines how IPv6 packets have to be encapsulated within IPv4 datagrams, so that they can be carried over IPv4 routing infrastructure, and addresses the tricky issues of fragmentation and ICMP error message mapping.

- DNS is used to provide IPv4 and IPv6 addresses for a machine name.

It is very important to note here that this is only a short-term solution, which will work as long as the IPv4 address space is not exhausted, as all dual layer nodes need IPv6 and IPv4. Other, much more complex solutions, such as real address translations, will have to be defined after the address space exhaustion.

State and availability of IPv6

Work on the IPv6 protocol has been finished though there is still some remaining work on higher and lower layers, such as to support IPv6 in all routing protocols over all different media and enhance the API for IPv6.

There is already a variety of router and host implementations available from different vendors and for different architectures. A global IPv6 based backbone (6bone) is operational and growing.

Assessment

IPv6 is a neat new version of IP, cleaning up old functionality and adding some new functionality to make it a stable and future proof network layer. It comes as a SW upgrade to current IPv4 equipment and offers an incremental transition phase, minimising costs for new equipment and protecting past investments.

A very important new feature of IPv6 is its basic support of QoS at the network layer through flow labels and priority indication. This does not mean however, that IPv6 alone can guarantee real end-to-end quality of service as there is no way to make network resource reservations. But IPv6 provides the network layer functionality which allows end-to-end quality of service to be provided when used together with protocols for network resource reservation like RSVP (see section 3.3). IPv6 basic QoS support also provides the basic functionality for a future, traffic based charging.

The larger address space overcomes the current limits of the Internet growth and has the potential to provide worldwide, universal connectivity. The big challenge for IPv6 was thought for its transition to be completed before IPv4 routing and addressing break. If this can not be achieved, very complex address translation solutions would have to be used to be able to keep the Internet paradigm of universal connectivity alive. However, on the basis of the NAT protocol (Network Addressing Translation [112]) that introduces the concept of globally non-unique addresses, the limitation of the current addressing scheme is no more an urgent problem. In fact many organisations do need any more a large number of unique addresses, because they can use a limited set for the accessible hosts, and give to the hosts are on their internal network, globally non-unique addresses, because they do not necessitate to be known outside. Therefore the pressure on finishing standardisation work is reduced and the deployment of IPv6 is slowing down.

The security features of IPv6 support authentication and privacy. They also provide the basic functionality for service charging.

With its new plug and play features, IPv6 networks are much easier to configure and maintain.

All higher layer protocols and applications need to be ported to be able to make use of the new functionality provided by IPv6.

E.3.3 Resource ReSerVation Protocol (RSVP)

Motivation

IP provides best effort datagram delivery that is sufficient for most of the conventional applications such as e-mail, WWW and file transfer. However, a new class of application (e.g. multimedia) is emerging that requires guaranteed resources from the network in order to function properly. Typically such requirements for resource guarantees are related to stringent real time requirements.

To address the problem of resource reservation in the Internet, the IETF formed the Integrated Services working group. This working group with the goal of "efficient Internet support for applications that require service guarantees" is defining an Integrated Services framework, of which RSVP is an integral part.

It has to be noted here that RSVP is enhancing IP based networks to support end-to-end quality of service, it is however not related to ATM.

RSVP is a signalling protocol for the Internet.

Protocol overview

Resource ReSerVation Protocol, RSVP [2][113], has been proposed to be the protocol that allows applications to reserve network resources in an IP network such as the Internet. RSVP operates on top of IP (either IPv4 or IPv6) and it relies on standard Internet routing. It is used both in hosts and routers to reserve resources for a simplex (uni-directional) data stream, called a flow. A flow is a sequence of datagrams identified either by the IP destination address (either multicast or unicast address), or by the IP protocol ID and optionally by a destination port. The requested QoS for the flow is described by a flowspec together with a filter spec. These two form a flow descriptor that is carried in the resource reservation message.

RSVP is designed for both unicast and multicast communication in a heterogeneous network, where receivers may have different characteristics and multicast membership is dynamic. These requirements lead to a solution, where the receiver is responsible for initiating the resource reservation.

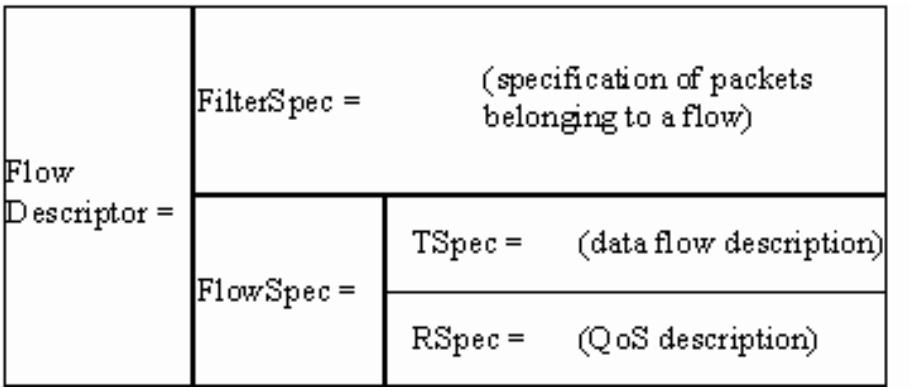


Figure E.4: RSVP flow descriptor.

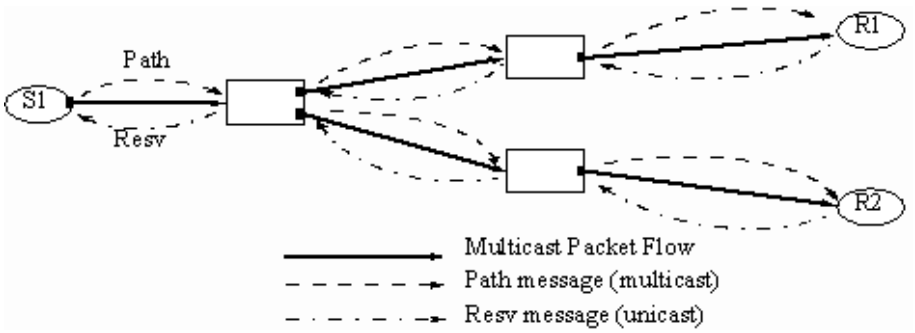


Figure E.5: RSVP message exchange in the multicast tree.

The message flow for the establishment of a network reservation for a multicast communication is shown in Figure 5.

It is assumed that a multicast group already exists (created by Internet Group Management Protocol, IGMP [114]). The sender S1 sends a Path message to a multicast group announcing the characteristics of the flow it is going to send. The Path message contains a Tspec, describing the maximum traffic characteristics of its data flow, and a Filter Spec, describing the packet format of the flow. When the receivers, R1 and R2, want to make a resource reservation, they will send a Resv message upstream following exactly the inverse path of the Path message. The Resv message contains the desired reservation style (see Figure E.8) and flow descriptor. The Resv message creates reservation state in each RSVP capable router along the path from the receiver to the sender. In a multicast situation as the one shown in Figure E.5 there are nodes that will receive two or more Resv messages from

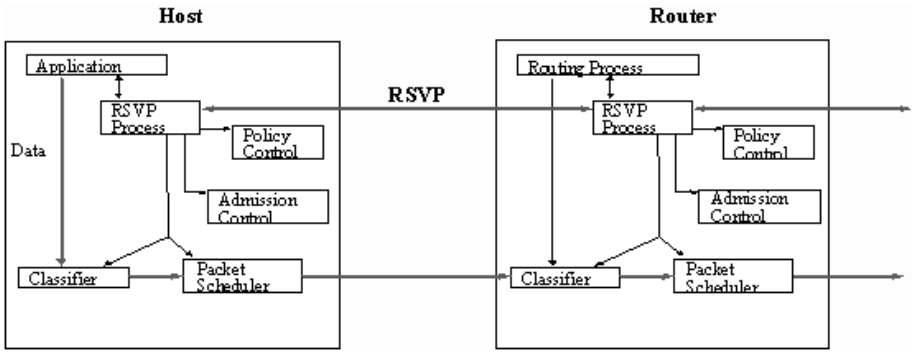


Figure E.6: RSVP in hosts and routers.

different branches of a multipoint tree. These nodes merge the received reservations and forward only one merged reservation request upstream, containing the most demanding (maximum) flowspec.

The resource reservation request indicated in the Resv message has to pass admission control and policy control modules in all RSVP equipped routers and hosts on its way. These check if the reservation can be accepted. Admission control determines whether the node has sufficient resources and policy control [115] deals with administrative issues such as accounting and access rights. If the reservation passes these two checks, flow related parameters are set in the packet classifier and packet scheduler. If either of the checks fail, an error notification is returned. The packet scheduler is responsible for negotiation with the link layer to reserve the transmission resources. It is here that mapping from the flow level QoS to the link layer QoS takes place.

The treatment of RSVP reservations in routers depends on the reservation style indicated by the Resv message. The different styles and attributes are listed in the following figure.

The reservation styles indicate either if there should be a separate reservation for each sender of a session (Fixed-Filter), or if the reservation can be shared among the named senders of the session (Shared-Explicit), or if the reservation can be shared by all the senders (Wildcard-Filter). Fixed-Filter, Shared-Explicit and Wildcard-Filter style are mutually incompatible. This results in rules for merging the reservations. For example, merging of shared reservations with distinct reservations is prohibited.

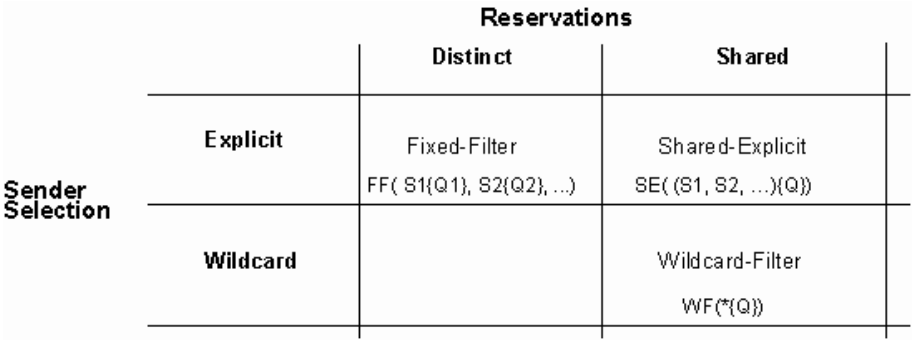


Figure E.7: RSVP reservation attributes and styles.

RSVP uses soft state for the reservation. This means when a reservation is made, it must be periodically refreshed (suggested refresh period is currently 30 seconds). Refreshing is accomplished by sending Path and Resv messages. The advantage of using soft state for the reservation is that the route of the connection can be changed dynamically inside the network and the reservation will be re-established when the new Path and Resv messages has passed through the new route. Soft states also help to allow for dynamic multicast group membership.

In addition to Resv and Path messages RSVP has messages for tearing down the reservation state. The PathTear message is sent from the sender to tear down the path and thus the reservation state and ResvTear is sent from the receiver. A sender can request reservation confirmation to its Resv message; the sender or a router that is merging the reservation to another reservation sends a ResvConf message to confirm the reservation.

Assessment

RSVP defines an efficient, flexible and robust solution for setting up resource reservations in IP based networks, but it does not scale well for large number of flows. RSVP is especially tailored for the need of multicast connections in heterogeneous networks.

With the support of resource reservation in the network application requested end-to-end QoS becomes possible. However it remains unclear, how routers are going to map the resource reservations to internal settings for the packet classifier

and scheduler and how reliably they are going to support the requested reservation. Furthermore end-to-end QoS can only be guaranteed if all the routers and hosts along the routed path are running RSVP software, because tunnelling through non-RSVP clouds destroys all end-to-end QoS.

The fact that RSVP sets up reservations in the upstream direction of a pre-established multicast tree makes it impossible that QoS information is used for routing decisions.

The use of RSVP in the Internet may provide input for traffic based charging.

There is an IETF RSVP Working Group that is in charge of evolving the RSVP specification. The RSVP-WG also coordinates its work with the parallel IETF working group that is considering the service model for integrated service, in order to have RSVP compliant with the overall integrated service architecture and the requirements of real-time applications. The RSVP-WG also coordinates its work with the IPng-related working groups.

E.3.4 Current research directions in IETF

Current resource reservation architectures for multimedia networks in IETF (e.g. RSVP) don't scale well for a large number of flows. This complicates the use of those architectures in network backbone areas, and the utilisation of RSVP is strongly limited by this scarce scalability. The effort to find a solution to this problem converged to methods that do not depend (at least completely) on a per flow resource reservation. There are two interesting approaches recently presented in IETF: Differentiated Services and SRP.

Differentiated Services

In the Internet world is commonly defined a Differentiated Services mechanism any simple mechanism that does not depend entirely on a per flow resource reservation and allows providers to differentiate the service pro user base. However differential services is still far to be stabilised; even the exact coverage of the definition differentiated services is still font of debate, because of the several proposal under investigation.

Two of those possible approaches are described in [116] and [117]. Both methods distinguish traditional best-effort traffic from traffic that requires/wants a better service. The global network is therefore seen as divided in two (independent) virtual networks: the current Internet and a "contracted network". The contracted network is monitored and the traffic that exceeds its profile is dropped or not assured. Note that this is different from priority schemes where, under congestion, lowest priority traffic is discarded even if it respects its contract; here traffic is discarded only if it does not respect the contract. In [116] the profile is statistically provisioned, while in [117] it is given on the base of the expected capacity.

The contracted network is transparent to best-effort traffic, which is transmitted as in the current Internet.

SRP

SRP (Scalable Reservation Protocol) [118] proposes a new architecture that aggregates flows on each link in the network. Therefore, the network has no knowledge of individual flows, and scalability even for very large numbers of flows results. Admission decisions are performed by routers independently and on a per-packet basis. Routers estimate the current level of reservation by continuously measuring reserved traffic and accepted requests.

Contrary to traditional approaches, routers learn about (aggregate) flow behaviour by monitoring and therefore only need to know the type of each packet (reserved, request, or best effort) and no explicit signalling or globally known classification of possible traffic patterns is needed. Furthermore, a feedback protocol is used by the destination to report end-to-end reservation status to the source. This protocol is transparent to routers.

Resource management functions traditionally implemented in the network (such as flow acceptance control) are delegated to hosts. Conformance of sources and intermediate systems can be controlled in a hierarchical structure by monitoring aggregated flows at routers.

E.4 ATM Technology

E.4.1 Introduction

Asynchronous Transfer Mode (ATM) is a cell-based, connection-oriented switching technology that is designed to support a wide variety of services, including cell relay, frame relay, SMDS, and circuit emulation. ATM transmits all information using small (53 byte) fixed length cells over broadband or narrowband transmission facilities. It is asynchronous because the cells carrying user data are not required to be periodic. The asynchronous and multimedia characteristics of ATM are what makes it possible for ATM networks to carry both circuit and packet types of traffic simultaneously, with complete transparency to the applications. ATM was designed to provide large amounts of bandwidth economically and on-demand. When a user does not need access to a network connection, the bandwidth is available for use by another connection that does need it.

The ATM technology was defined by the ITU-T, mainly formed by the representatives of public network operators. The rather slow development of standards in ITU-T was sped up by the ATM Forum (founded in 1991), a growing group of companies focusing on private network and data communication.

The term ATM can be interpreted in a variety of ways. In fact, it is true to say that there is no single definition. It takes on many forms, encompasses both hardware and software, and can run on several types of digital transmission facilities. ATM can refer to a physical interface (the 53-byte cell), a switching technology, or a unifying network technology that provides integrated access to multiple services.

There is a broad consensus that ATM will first be implemented within wide area networks primarily as a switching technology to support existing services in private WANs and in public service networks. ATM excels primarily as a backbone technology, because it is in this context that most of the benefits of cell relay are realised.

E.4.2 Virtual Paths and Virtual Channels

Each ATM cell contains a two-part address, a Virtual Path Identifier (VPI) and a Virtual Channel Identifier (VCI), in the cell header. This address uniquely identifies an ATM virtual connection on a physical interface. The physical transmission path (such as DS1 or DS3) contains one or more virtual paths, and each virtual path can contain one or more virtual channels. The VPI and VCI are tied to an individual link on a specific transmission path, and have local significance only to each switch. The VPI and VCI addresses are translated at each ATM switch in the network connection route – each switch maps an incoming VPI and VCI to an outgoing VPI and VCI. Therefore, these addresses can be reused in other parts of the network as long as care is taken to avoid conflicts. ATM can perform switching on a transmission path, a virtual path, or a virtual channel.

E.4.3 Permanent Virtual Circuits and Switched Virtual Circuits

ATM provides two virtual circuit communications services: Switched Virtual Circuits (SVCs) and Permanent Virtual Circuits (PVCs). SVCs establish short-term connections that require call setup and teardown, while PVCs are similar to dedicated private lines because the connection is set up on a permanent basis. Users establish PVCs either by requesting them from a public carrier providing the frame relay or ATM service, or from the WAN administrator of the private network. ATM virtual connections can operate at a constant bit rate (CBR) for voice and video traffic, at a variable bit rate (VBR) for bursty traffic and at available bit rate (ABR) or unspecified bit rate (UBR) for best effort traffic. Each virtual connection has its own set of parameters (Minimum Cell Rate (MCR), Sustained Cell Rate (SCR), Peak Cell Rate (PCR)), that determine the amount of bandwidth, priority, Quality of Service (QoS), etc.

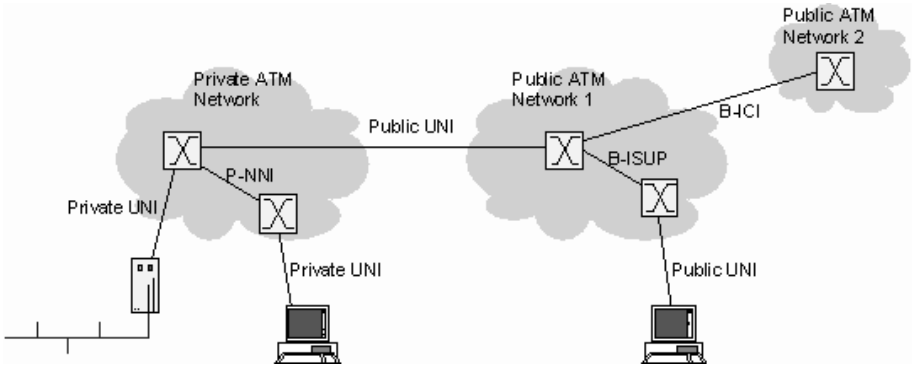


Figure E.8: ATM network architecture and interfaces.

E.4.4 ATM Signalling, Routing and Addressing

ATM Signalling Protocols vary by the type of ATM link - ATM UNI signalling is used between an ATM endsystem and an ATM switch across an ATM UNI; ATM NNI signalling is used across NNI links.

The current standard for UNI signalling is described in the ATM Forum UNI4.0 specification [25], which is an enhancement to the earlier UNI3.1 and provides some kind of alignment to the recommendations for public UNIs specified by ITU-T. Besides the basic call set up and tear down functionality UNI4.0 defines point-to-multipoint operation, address registration and extended QoS support. An overview of UNI signalling capabilities can be found in [119].

ATM switches are interconnected via one of three NNIs: P-NNI in private ATM networks, B-ISUP in public networks and B-ICI between different public networks. NNI interfaces define not only signalling procedures but also routing. P-NNI defined by the ATM Forum [120] supports not only signalling procedures similar to UNI4.0 but also topology discovery via the distribution of reachability information, hierarchical routing and addressing and QoS.

Whereas ITU-T has long settled upon the use of telephone number-like E.164 addresses for public ATM networks, the ATM Forum defined a private address format based on the syntax of an OSI Network Service Access Point (NSAP) address to be used in private networks.

Application Programmer Interfaces (APIs) for ATM are still under definition.

The industrial standard WinSock2 (for Windows based applications) is becoming available now.

E.4.5 Assessment

The most significant advantages of pure ATM solutions are:

- ATM supports end-to-end QoS guarantees on a per virtual connection basis. ATM virtual connections allow users to expect a guaranteed minimum amount of bandwidth for each connection. ATM supports several Quality of Service (QoS) classes to accommodate the differing delay and loss requirements for each type of traffic.
- ATM uses statistical multiplexing, which allows bandwidth to be shared among many users. Bandwidth is only provided when it is needed "on demand", thus reducing the cost of network resources.
- ATM supports multiple services. ATM can be used to transport literally any kind of information and can simultaneously support a broad range of user interfaces. Only ATM WANs can provision frame relay, SMDS, native ATM, voice, video, and existing leased circuit services (circuit emulation) over the same wide area circuits.
- ATM provides high performance.
- ATM enables traffic based charging.

On the other hand, ATM has got some disadvantages that may interfere with widespread deployment, ease of high-speed implementation or present architectural concerns.

- ATM technology is very complicated and the control software is extensive and complex.
- Connection establishment times may be prohibitive for short duration data flows.

- Applications have to know their QoS demands in advance and can not easily adapt to changing network load.
- No multipoint-to-multipoint support
- Currently there are hardly any applications available that run directly on top of ATM and can exploit its benefits. ATM APIs are only emerging now and today's TCP/IP based applications will have to be changed considerably to be adapted to ATM and make subtle use of resources.
- ATM does not provide security. This will have to be handled in higher layers.
- As public ATM network deployment is still very slow, the connectivity in the public WAN area is bad today.
- ATM generates quite a lot of overhead (20 %).

E.5 IP/ATM Co-Existence

Given the vast installed base of LANs today, the variety of LAN based applications and the network layer protocols operating on these networks, the key to the success of ATM in the short and medium term will be its ability to allow for interoperability between itself and these technologies. To enable the connectivity between ATM and existing LANs it is essential to use the same network layer protocols (such as IP, IPX) to provide a uniform network view to higher level protocols and applications.

Today, there are two standards available to run the predominant IP protocol over ATM: ATM Forum's LAN-Emulation (LANE) and IETF's "Classical IP over ATM". Both use the so-called overlay model, where IP addresses are mapped to ATM addresses. ATM is only used as a very fast packet transmission system and neither LANE nor Classical IP over ATM can therefore exploit ATM's QoS support as the IPv4 layer hides all the good features of ATM from higher layers. Moreover both technologies can establish ATM end-to-end VC connections only inside a 'subnet' (LIS or VLAN) and require IP routers for traffic across 'subnets', with the routers

becoming potential performance bottlenecks. LANE and Classical IP over ATM are presented in Section 5.1.

There are also several ongoing activities in the ATM Forum and the IETF to enhance their overlay protocols to make better use of ATM. NHRP and MPOA are discussed in Section 5.2.

However there are solutions available already today, which can bring QoS to IP based applications by supporting end-to-end ATM VC connections on a per flow basis and across subnet borders. Section 5.3 introduces Arequipa, providing application requested end-to-end ATM connections with QoS for IP based applications, and some of the approaches that combine the label switching technology of ATM with network layer routing (IP Switching, Tag Switching) while avoiding the usage of ATM addressing, routing and signalling altogether.

E.5.1 Co-Existence without QoS Support

There are two fundamentally different ways of running network protocols over ATM networks. One method is the native mode operation, where network layer addresses are mapped directly to ATM addresses and network layer packets are sent directly across the ATM network, the other method is LAN Emulation.

LAN Emulation (LANE)

General overview

The ATM Forum specified LAN Emulation (LANE) in order to accelerate the deployment of ATM in the local area while native mode operation is still under definition. LANE offers a solution to the problem of running predominant local area protocols like Ethernet and Token Ring transparently over an ATM network. The version of LANE that is implemented in most of the products available on the market can be found in [121], and is the one we refer thereafter.

LANE emulates a bridged LAN on top of an ATM network by offering a service interface to network layer which is identical to that of existing LANs (e.g. IEEE 802.3 Ethernet or 802.5 Token Ring) and it sends data across the ATM network using

appropriate LAN MAC encapsulation. In brief, LANE makes an ATM network look and behave like an Ethernet or Token Ring, albeit a fast one. The big advantage of emulating a LAN is, that all network layer protocols and applications can be used without any modifications.

Today, LANE protocol software is widely available on ATM hosts (either implemented in the operating systems or on ATM network interface cards (NIC)) and on LAN Switching Equipment. ATM switches are transparent for the operation of the LANE protocol. They do not need to be modified for the use of LANE, although some of the LANE server components could be implemented on them.

The main issues on emulating a LAN technology like Ethernet on an ATM network is address resolution, broadcast and data encapsulation. Address resolution from MAC to ATM addresses is solved by using a special protocol called LE_ARP between hosts and a special LANE entity known as the LES (LAN Emulation Server). Broadcasting is emulated by sending packets to another LANE entity known as the BUS (Broadcast and Unknown Server) which distributes the packets to all hosts. The LAN packets (e.g. Ethernet frames) are encapsulated in AAL5.

Architecture

In the upper part of Figure E.9, the architecture of a standard bridged LAN environment is shown. On a shared medium LAN (such as Ethernet) all packets sent by one station travel to all other stations on the medium. Bridges are intelligent repeaters (layer 2) which try to avoid unnecessary forwarding of packets. The functionality of a LAN segment can be emulated by an ATM network running LANE (lower part of Figure E.9). This emulated LAN segment is called an Emulated LAN (ELAN). Together with the remaining old LAN infrastructure it forms a virtual LAN (VLAN).

For the operation of LANE the following entities are needed:

- LEC (LAN Emulation Client) A LEC runs on every host. It provides a standard LAN interface to upper layers. The LEC issues address resolution requests and performs data encapsulation and forwarding.

- LES (LAN Emulation Server) There is a single LES per ELAN. It registers

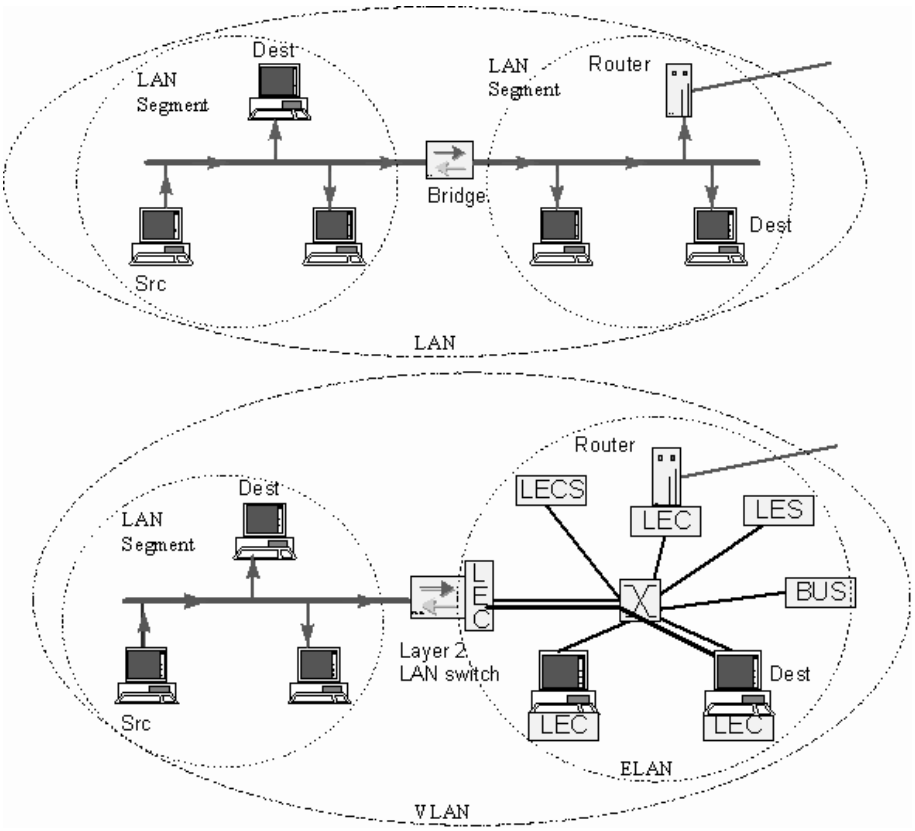


Figure E.9: Classical LAN and Emulated LAN architecture.

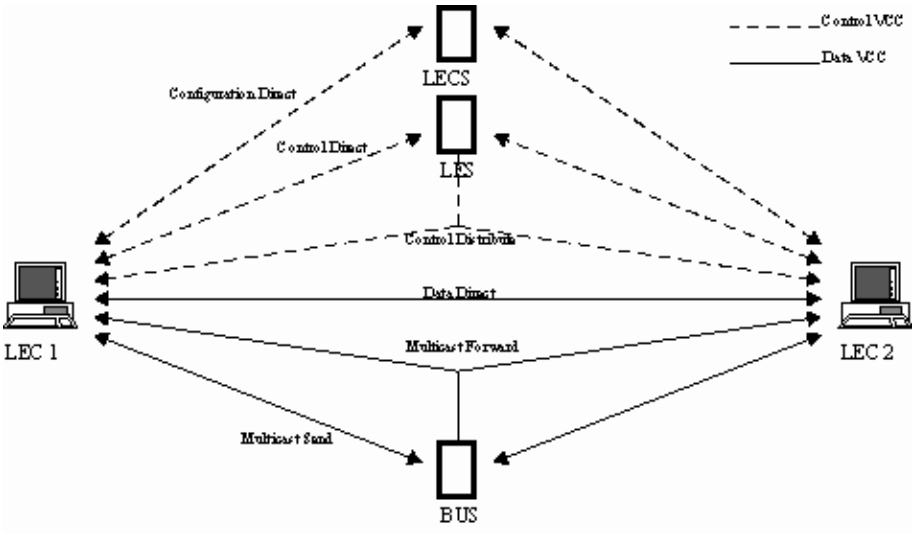


Figure E.10: ATM connections for LANE.

the mapping of MAC to ATM addresses and replies to or forwards address resolution requests.

- BUS (Broadcast and Unknown Server) There is a single BUS per ELAN. It emulates broadcasting by forwarding packets to all known ATM addresses on the ELAN.

- LECS (LAN Emulation Configuration Server) There is a single LECS per domain, used for the configuration of several ELANs.

Several Virtual Channel Connections are needed between these entities. Figure E.9 shows the VCCs in a 2 host ELAN.

All these VCCs are established by signalling (SVCs). The VCCs are either UBR or ABR.

LANE procedures

LANE Configuration:

- LEC establishes the Configuration Direct VCC to LECS.

- LEC learns from LECS the ATM address of LES over the Configuration Direct VCC.

- LEC sets up the Control Direct VCC to LES and registers its ATM and MAC address in LES.
- LES adds LEC as a leaf to its point-to-multipoint Control Distribute VCC.
- LEC learns ATM address of BUS by using LE_ARP to LES for the MAC broadcast address.
- LEC sets up the Multicast Send VCC to BUS.
- BUS adds LEC as a leaf to its point-to-multipoint Multicast Forward VCC.

LANE Operation:

- LEC1 wants to send to LEC2, but only knows its MAC address.
- LEC1 uses LE_ARP request to LES to map LEC2's MAC address to its ATM address.
- While waiting for the reply, LEC1 sends packets to BUS, which floods it to all connected LECs.
- After receiving the LE_ARP response LEC1 sets up the Data Direct VCC to LEC2.
- Before sending on the Data Direct, LEC1 has to send a flush to BUS to make sure that all packets previously sent to LEC2 over BUS were delivered (to preserve frame ordering).

Assessment

LANE is a good solution to interconnect legacy LAN equipment in a private network, exploiting ATM's fast transmission speed with minimal changes to LAN equipment and no changes at all to higher layer protocols and applications. Its a working solution for today and allows for a smooth integration from LAN to ATM in a corporate network.

LANE even introduces enhanced configuration flexibility and improved management compared to standard LANs with its concept of virtual LANs.

However LANE can not be the ultimate solution for a modern integrated services network because of the following limitations:

- LANE totally hides the QoS support of ATM with its emulation of a connectionless shared media technology.
- LANE is unable to run protocols in native mode.
- LANE is limited to a logical subnet (VLAN).
- All inter-VLAN traffic has to pass through routers even if direct ATM connectivity would be possible. These routers are likely to become bottlenecks.
- LANE address translation is very inefficient because addresses are translated from Layer 3 addresses to MAC addresses to ATM addresses, using two different address resolution mechanisms.
- LANE operation needs a lot of connections, limiting the number of stations that can be attached to an emulated LAN
- LANE has not recovery mechanisms for the server, thus it does not foresee the possibility to define backup LES and BUS to manage the VLAN in emergency conditions.
- LANE has limit on the MTU size. Currently LANE has been split in LUNI (LANE User Network Interface) and LNNI (LANE Network to Network Interface). For LUNI it exists the version 2.0 [122].

CLassical IP over ATM (CLIP)

General description

A solution to overlaying IP networks on ATM networks is the so-called Classical IP over ATM, specified by the IP-Over-ATM working group of the IETF and described in detail in RFC 1577 [123].

The Classical IP model refers to a network where hosts are organised in subnetworks sharing a common IP address prefix, where the ARP is used for IP address to

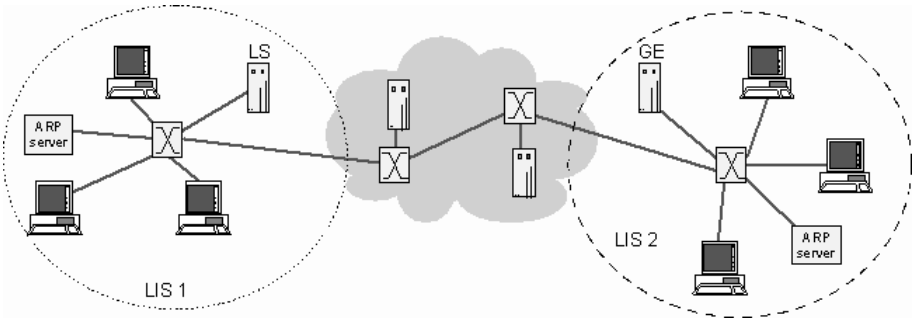


Figure E.11: Classical IP over ATM network architecture.

MAC address resolution and where communication across subnetworks goes through routers. Preserving the classical IP model on ATM means that ATM is used as a direct replacement for the "wires" and local LAN segments connecting IP end-stations ("members") and routers operating in the "classical" LAN-based paradigm.

The Classical IP over ATM specification defines classical IP and ARP in an ATM network environment configured as a Logical IP Subnetwork (LIS) as illustrated in Figure E.11. It does not describe the operation of ATM Networks in general.

It is the goal of RFC 1577 to allow compatible and interoperable implementations for transmitting IP datagrams and ATM Address Resolution Protocol (ATMARP) requests and replies over the ATM Adaptation Layer 5 (AAL5).

Data transmission in Classical IP over ATM is based on the virtual connection (VC) switched environment provided by ATM networks. The VCs can either be established by management (PVCs) or by signalling (SVCs). Each VC is directly connecting two IP members within the same LIS and carries all IP data flow between them.

The ATM connections could in principle be of any type (CBR, UBR, VBR, ABR), but only CBR and UBR is used in today's implementations.

Encapsulation of IP Datagrams

IP packets are transmitted using AAL5 with a maximum packet size (MTU) of 9180 bytes. Additionally, when using SVCs, IP packets must be encapsulated with LLC/SNAP [31] and the SETUP signalling messages to establish these SVC must carry Lower Layer Information (BLLI) indicating that the packets should be

delivered to the LLC entity [124].

Address Resolution Mechanisms

When SVCs are used for transmission, special address resolution mechanisms are needed to map IP addresses to ATM addresses and vice versa. Similar to classical IP networks where ARP [104] and InARP [125] are used to map between IP and MAC addresses, Classical IP over ATM defines ATMARP and InATMARP services to map between IP and ATM addresses. For example, if Host A wishes to send IP datagrams to Host B it needs to have the ATM address of Host B to be able to establish a switched VC using signalling. For this IP to ATM address resolution, the ATMARP service is used. The originating host sends an ATMARP request to a special network entity, the dedicated ATMARP server of the LIS. The ATMARP server, knowing the IP and ATM addresses of all hosts and routers in its LIS (see below), maps the provided IP address of Host B to the corresponding ATM address and sends it back to Host A. Host A can then establish an SVC to Host B using normal signalling procedures. Host B then uses InATMARP procedures on this newly established connection to learn the IP address of Host A.

When hosts are connected by PVCs, they may use a preconfigured table to map IP addresses to VCs but they have a mechanism for resolving VCs to IP addresses via InATMARP for new VCs.

Each host must know its own IP and ATM address(es) and must respond to address resolution requests appropriately. It must also be configured with the ATM address of an ATMARP server (for SVCs only) located within the LIS (there is only one server per LIS). At power-up a host establishes a connection to the server. On each new incoming connection the ARMARP server send an InATMARP request and registers the reply. The reply contains the information for the ATMARP server to build its ATMARP table cache. This information is used to generate replies to the ATMARP requests it receives.

Because ATM does not support broadcast addressing, there is no mapping from IP broadcast addresses to ATM broadcast services. This is currently also true of multicast address services, although an Internet draft for multicast support already

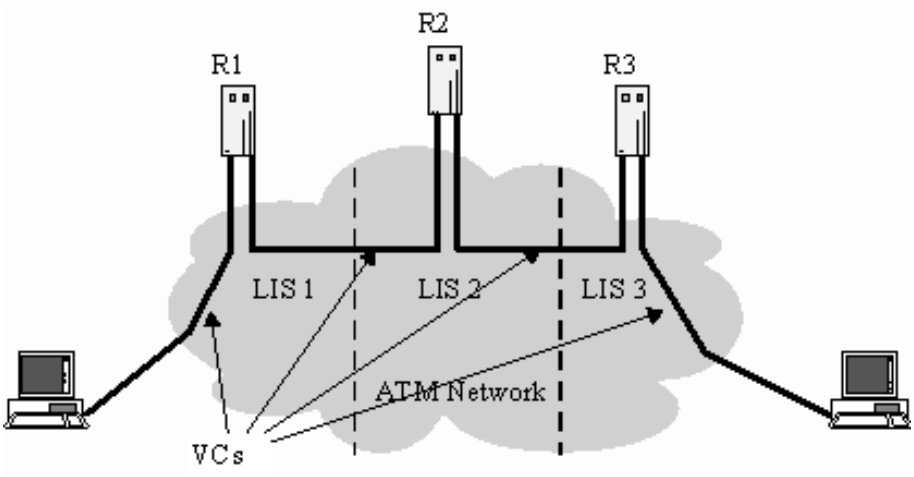


Figure E.12: routing for traffic across LIS borders.

exists [126].

All hosts as well as the server must maintain an ATMARP table. A table entry contains the IP address, ATM address and VCI/VPI of a connection together with encapsulation information and a timestamp. Hosts must refresh the entries at least every 15 minutes and the server must refresh the entries at least every 20 min. Connections are released after a certain idle period.

It is important to stress the fact that the address resolution mechanisms of Classical IP over ATM can only be used inside a single LIS and not across LIS borders.

Routing

Classical IP over ATM uses exactly the same end-to-end routing architecture as the classical IP network. As the classical IP network uses ARP protocols and tables for the routing inside of the subnetwork, so does Classical IP over ATM use ATMARP protocols and tables for the routing inside of a LIS. For communication across LIS borders routers are needed in the same way as when crossing subnet borders in classical IP networks. This leads to the necessity of using several hops over IP routers across an ATM networks for traffic between hosts in different LIS (see Figure E.12).

Assessment

The main advantage of using Classical IP over ATM is its full compatibility with normal IP, enabling the vast set of higher layer protocols and applications to run transparently over ATM while making use of ATM's high bandwidth availability. Another advantage is that Classical IP over ATM allows easy integration of IP based services with other ATM services (e.g. voice).

The major shortcoming of Classical IP over ATM is that it can not benefit from ATM's inherent end-to-end QoS guarantees for the following reasons:

- Direct ATM connections can only be established inside a LIS but not across LIS borders. Because of using the classical IP routing mode (address resolution is limited to a LIS) IP traffic between hosts on differing LISs always flows via one or more intermediate IP routers who can only provide best effort delivery on IP level. This results in a concatenation of ATM connections even though it may be possible to open a direct ATM connection between the two hosts, thus pre-empting end-to-end QoS. In other words, IP packets across LIS borders hop several times through the ATM network instead of using one single hop.

- All IP data flowing between two hosts shares the bandwidth of a single VC. Having only one shared VC between two hosts makes it impossible for individual applications to get a QoS guarantee for their specific data flow.

Other shortcomings of Classical IP over ATM are that neither multicast nor anycast is supported, that IP layer implementations need to be adapted to interface with ATM directly and that it is necessary to deploy routers with ATM interfaces in every LIS. Furthermore there is no default path to forward IP datagrams before a connection is established, resulting in a high delay for the passing of the first datagram.

Unlike LANE Classical IP over ATM does not allow to use much of the old legacy LAN equipment, but it offers a more appropriate MTU size (larger).

E.5.2 QoS Support by Emerging Standards

Both the IETF and the ATM Forum are aware of the shortcoming of their respective solutions of running IP over ATM (CLIP, LANE) and try to solve them by defining new additional standards. NHRP, under definition in IETF, tackles the extra-hop problem (router hops are required for traffic across LIS instead of direct ATM connections) to provide end-to-end ATM connectivity and bring the QoS features closer to IP based applications while generalising on Layer 2 (IP over any layer 2). MPOA, under definition in the ATM Forum, defines a way to emulate a routed protocol over ATM and also addresses the extra-hop problem but generalises on Layer 3 (any layer 3 over ATM).

Next Hop Resolution Protocol (NHRP)

General description

The IETF is generalising its approach to support IP (and other internetworking protocols) not only over ATM but over all kinds of Non-Broadcast Multiple-Access (NBMA) networks, such as ATM, Frame Relay or X.25. For this purpose, the IP over NBMA (ION) working group was formed as a successor of the Routing on Large Clouds (ROLC) and the IP over ATM (ipatm) working groups. The Next Hop Resolution Protocol (NHRP) was defined by ION as a key element of supporting IP over NBMA. NHRP is currently only an Internet Draft [127].

NHRP addresses one of the key problems in NBMA networks, namely the problem of stations communicating over a Non-Broadcast Multiple-Access (NBMA) subnetwork, that are not on the same LIS. The NHRP protocol allows the internetworking layer addresses and NBMA addresses of suitable "NBMA next hops" toward a destination station to be determined.

As we already pointed out in the description of Classical IP over ATM (see section 5.1.2), the address solving problem arises when the stations are in the same NBMA network, but not in the same LIS. In fact, in this scenario, classical address resolution as described in RFC1577 [121] and RFC1209 [128] does not work, because it can only discover a router that is a member of multiple LISs, and packets can hop

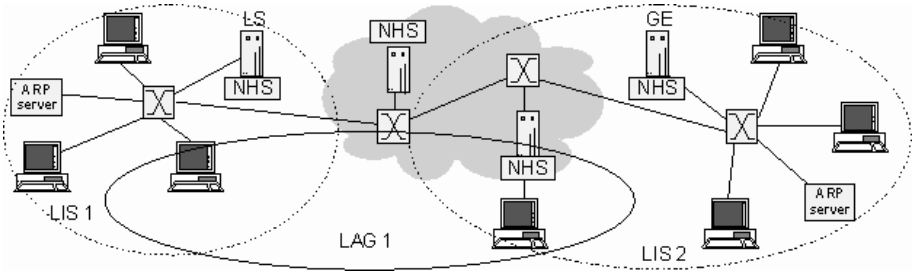


Figure E.13: the Local Address Group (LAG) concept.

several times through the NBMA network instead of using one single hop. NHRP solves this problem with the definition of an inter-LIS address resolution mechanism, providing the source station with a "short-cut" routing, that allows to communicate through the NBMA network without having to involve intermediate routers.

In this sense NHRP is not a routing protocol, but just an inter-LIS address resolution mechanism that makes use of network layer routing in resolving the NBMA address of the destination. Therefore NHRP does not replace existing routing protocols, that are still used to determine the source path (other means than routing can be used to do it, for example, static configurations) .

NHRP replaces the concept of LIS with the concept of Local Address Groups (LAGs). LAGs were introduced in [129] to extend IP architecture that limits direct communication between hosts with the same subnet, to large data network. LAGs are identified by an IP address prefix, and group hosts and routers with different subnet. As described in [127], for NHRP the essential difference between using the LIS or the LAG models is that while with the LIS model the outcome of the "local/remote" forwarding decision is driven purely by addressing information, with the LAG model the outcome of this decision is decoupled from the addressing information and is coupled with the Quality of Service and/or traffic characteristics. This implies that two stations that are on the same NBMA, but that are not necessary on the same LIS, can directly communicate being part of the same LAG, as illustrated in Figure E.13.

Protocol overview

For NHRP operation there has to be one Next Hop Server (NHS) in every LIS. All hosts on a LIS register their NBMA and internetwork layer (e.g. IP) address with their NHS when booting.

Assume a Host S wants to send an internetwork layer packet (e.g. IP) to Host D which lies outside its LIS. To resolve the NBMA address of D, S sends a next hop Resolution Request to its NHS. The NHS checks whether Host D lies in the same LIS (is served by the same NHS). If the NHS does not serve Host D, the NHS forwards the request to the next NHS along the routed path. Using this algorithm the request is passed on from NHS to NHS and eventually arrives at the NHS that serves Host D. This NHS can resolve Host D's NBMA address and sends it back to Host S in a next hop Resolution Reply either along the routed path or directly. If it is sent back along the routed path, intermediate NHSs can optionally store the address mapping information for Host D contained in the Resolution Reply to answer subsequent Resolution Requests. Using this mechanism NHRP provides S with the NBMA address of D, if D is directly attached to the NBMA, or in the other case the address of an egress router at the edge of the NBMA which has connectivity to D. Host S and Host D may choose to cache the address mapping.

Host S can choose to either drop the packet triggering NHRP, retain it until the arrival of the Resolution Reply or forward the packet along the routed path towards Host D.

Use of NHRP

Issuing an NHRP request would be an application dependent action [129], in particular because NHRP allows the special features provided by the NBMA to be used. Thus, when a "cost" is associated with NBMA connections, there is an evident advantage in using NHRP short cuts, i.e. only one connection across the NBMA. For example, when the NBMA network is ATM and the application requests QoS guarantees, the short-cut routing of NHRP helps to establish a direct VC in the ATM domain across several IP subnets, allowing the application to benefit from the QoS features of ATM.

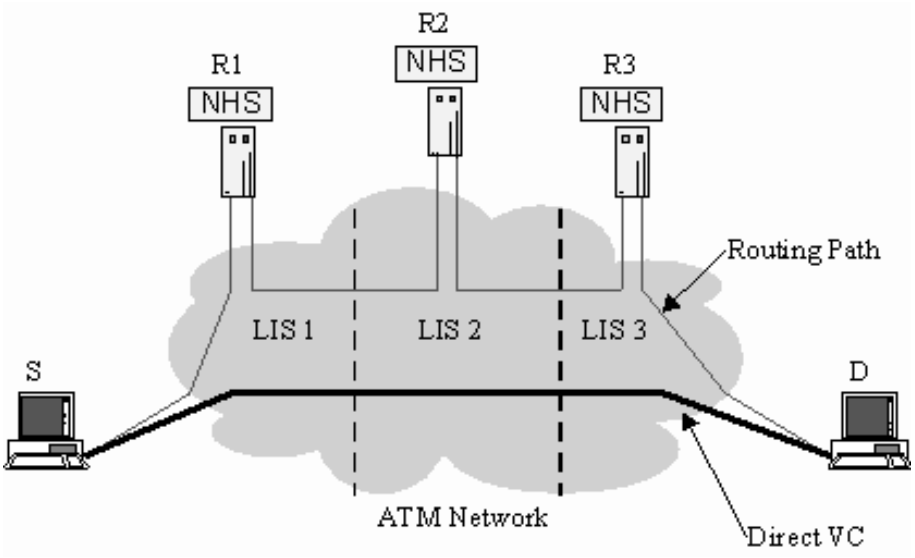


Figure E.14: NHRP established direct ATM connection across LIS borders.

For this reason, the Multiprotocol Over ATM (MPOA) Working Group of the ATM Forum has decided to use NHRP for resolving the ATM addresses of MPOA communications where the destination does not belong to the same Internetwork Address Sub-Group of the source [130], as illustrated in Figure E.14.

Assessment

The main advantage of NHRP is that it can solve the multiple-hop problem through NBMA networks by offering inter-LIS address resolution, thus enabling the establishment of a single-hop connection through the NBMA network. If the NBMA network is ATM, this means that by using NHRP a single direct VC can be established across several LIS, bringing QoS to the IP data flow between the VCs endpoints. But NHRP can only achieve this if the routed path lies entirely within the NBMA network and only under the conditions that NHRP is supported on all routers along the routed path. Furthermore it is also important to note, that even if a direct connection can be established through the NBMA network, it will be shared by all IP traffic between the two endpoints, which means that it does not bring QoS to an individual application.

Another problem with NHRP is that stable routing loops may occur, if NHRP

initiating and responding stations are routers, which are additionally connected over another network. Avoiding these routing loops imposes restrictions on the network configuration. But there is already work in progress [131] to augment NHRP to solve this problem.

Another negative effect that could arise with NHRP Resolution Request is the domino effect. This occurs when a router originates a NHRP Resolution Request for a transit packet (a packet arriving over one of its NBMA attached interfaces). If the router forwards this data packet without waiting for an NHRP transit path to be established, then the next transit router receiving the packet can originate its own NHRP Resolution Request and forward the packet, and so on. One solution proposed to solve this problem is that a router does not generate NHRP Resolution Request for transit packets, but only for packets on its non NBMA interfaces.

Deployment of seamless NHRP functionality require s additional software on all hosts and routers connected to the NBMA network.

The current NHRP specification works only for unicast communication, it does not suit a broadcast or multicast setting.

NHRP is only a draft and is far from being generally deployed.

Multiprotocol over ATM (MPOA)

Motivation

The ATM Forum's Multiprotocol Over ATM (MPOA) subworking group is defining an approach to support seamless transport of layer 3 protocols across ATM networks. Multiple layer 3 protocols are to be supported, such as IP, IPX, Appletalk, etc.

MPOA is extending the VLAN beyond what was defined in LANE based VLANs, addressing the well known shortcomings of LANE that router hops are required for VLAN interconnection and its inability to run protocols in native mode, which could exploit ATM's QoS features. In other words, MPOA tries to offer transparent emulation of routed protocols over ATM network, much the same as LANE offers transparent emulation of a LAN protocol over ATM network. MPOA provides end-to-end Layer 3 connectivity between hosts attached to the ATM fabric and hosts

attached to legacy subnetwork technology. MPOA operates at layer 2 and 3, but uses LANE for layer 2 forwarding.

MPOA was built with the following design goals in mind:

- Allow MPOA devices to Establish Direct ATM connections
- No significant changes to installed Bridges, Routers and Hubs
- Integrate with LAN emulation
- Support Network Layer Multicast and Broadcast
- Support Auto Configuration at ATM hosts
- Separate Switching from Routing

Much as the IP oriented IETF is trying to run only IP over all underlying technologies (ATM being only one of them), the ATM Forum tries to run all kind of Layer 3 (IP only one of them) protocols over only ATM. Where "IP over ATM" is concerned the two standardisation bodies converge and the IP version of MPOA can be considered the unification of Classical IP over ATM (together with MARS and NHRP extensions) and LANE.

So far the ATM Forum produced a MPOA Baseline document [130].

The MPOA reference model

The basic unit of organisation within MPOA is the Internetwork Address Sub-Group (IASG). It is defined as a range of internetwork layer addresses summarised into an internetwork layer routing protocol. In the case of IP this is essentially a subnet.

An IASG will contain a number of devices acting as MPOA servers and clients as described in the MPOA reference model (Figure E.15). Servers are those devices providing layer 3 co-ordination, address resolution, route distribution and broadcast/multicast forwarding. Clients are users of the MPOA services.

MPOA Clients are:

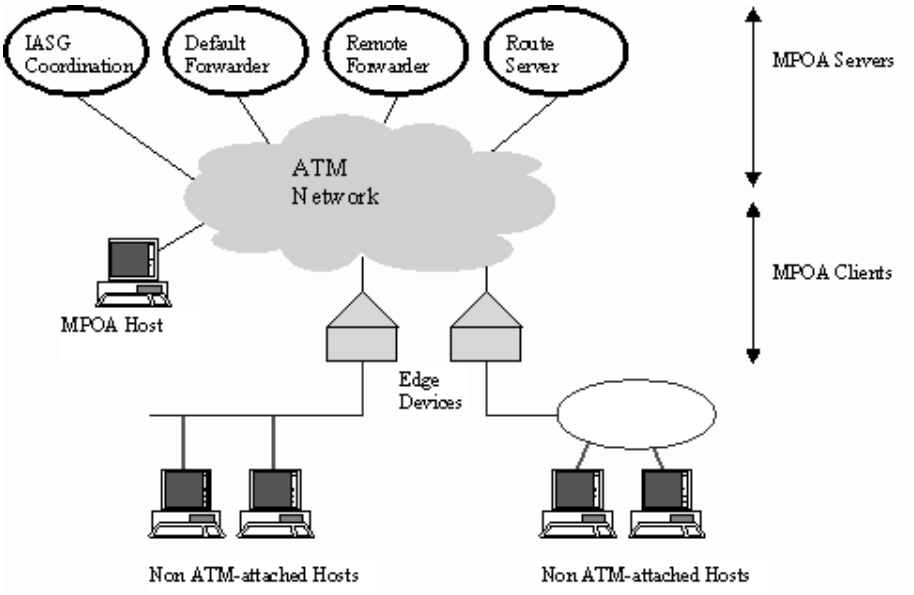


Figure E.15: MPOA reference model.

- MPOA Hosts: hosts that are directly attached to ATM, running MPOA protocol stack

- Edge Devices: physical devices that are capable of forwarding packets between legacy LAN interfaces and ATM interfaces at both Layer 2 and Layer 3. However they do not run layer 3 routing protocols to get the information for the Layer 3 packet forwarding, but they query the Route Server for this information.

  The services offered by MPOA Servers can be classified in the following functional groups:

- ICFG (IASG Coordination Functional Group): coordinates the distribution of an IASG across multiple traditional LAN ports and/or ATM connected hosts, it is responsible for the configuration of the IASG

- RSFG (Route Server Functional Group): runs layer 3 routing protocols, provides address resolution and route distribution

- DFFG (Default Forwarder Functional Group): forwards traffic within an IASG

if no direct client to client connection exists and performs the Multicast Server Function (MSF) within the IASG

- RFFG (Remote Forwarder Functional Group): forwards traffic between IASGs

MPOA architecture

Typically the MPOA server functionality is split among two physical entities, the Route Server and the IASG Coordinator. The IASG Coordinator provides ICFG and DFFG functionality. The Route Server provides RSFG and RFFG functionality.

MPOA Hosts have direct VCs to the IASG Coordinator and to the Route Server. Edge Devices, Bridges and LANE Hosts connect to the server entit ies over LANE, implying that the MPOA servers and these devices all run a LEC.

MPOA procedures

Procedures in MPOA are highly complex. Nevertheless a simplified description is given which relates to Figure E.16.

When initialising, all MPOA Hosts and Edge Devices announce their own Layer 3 and ATM addresses and the layer 3 addresses reachable through them to the IASG Coordinator and the Route Server. In parallel normal LANE initialisation takes place.

When a MPOA host desires to know how to contact another host over ATM it issues an address resolution query to ICFG. If the destination host is a MPOA host within the same IASG, ICFG can reply with its ATM address. If the destination host is in another IASG, the request will be passed among the RSFGs across IASG borders. In the destination host's IASG a RSFG/ICFG knows the ATM address of the destination host and can reply to the address resolution query. In either cases the source host can then establish a direct ATM connection to the destination host. Note that this functionality is identical to NHRP and indeed MPOA relies on this protocol. But in addition to the functionality of establishing a direct ATM connection, MPOA offers the passing of packets before the ATM connection is established by sending it from the source host over DFFG and several RFFG to the destination host along the routed path.

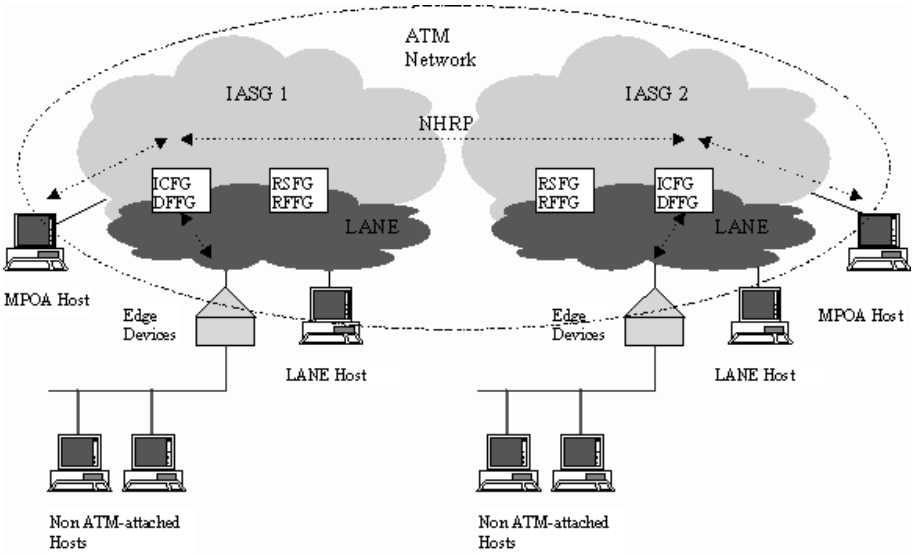


Figure E.16: MPOA architecture.

Now consider an Edge Device trying to send packets to another host over ATM. The edge device first looks at the MAC destination address. If it is not the MAC address of a router within the IASG, it has to remain inside the IASG and the Edge Device uses LANE either to send it directly if it knows the MAC to ATM address mapping (e.g. destination is a LANE host) or to send it to ICFG for forwarding. If the MAC address is the MAC address of a router, the Edge Device looks at the internetwork address contained in the packet. If it knows the internetwork to ATM address mapping (e.g. destination is a MPOA host) the Edge Device can forward it directly. If the internetwork address is unknown, the Edge Device asks the Route Server for an internetwork to ATM address resolution. In the latter case the Edge Device has the same behaviour like the MPOA host described in the previous paragraph.

Assessment

MPOA is a very complex technology and the work in the ATM Forum has only started and is far from being completed. IP might be worth the complexity because it is so widely used, but it can be doubted if this holds true for other layer 3 protocols as well .

Nevertheless, the MPOA model is a very promising technology providing the following benefits:

- MPOA provides the connectivity of a fully routed environment, supporting even multicast and broadcast at layer 3.
- MPOA takes maximum advantage of ATM:
- because it offers direct ATM connection between MPOA devices, without intermediate hops
- because it supports Native ATM, exposing QoS to layer 3 protocol stacks
- MPOA reduces infrastructure costs by defining a new network architecture. Instead of deploying common routers with both the functionality of switching, which is very cheap as it can be done in hardware, and route computation, which is rather expensive as it needs to run on a high performance platform, the switching is distributed in Edge Devices and there is only a single, centralised router
- MPOA provides an universal approach for layer 3 protocols over ATM
- MPOA easily integrates with LANE

Apart from its complexity, a disadvantage of MPOA is that host protocol stacks have to be changed.

E.5.3 QoS Support with Existing Technologies

This section discusses some of the solutions available today to bring ATM's high speed and QoS support to IP based applications. Most of these solution were born as proprietary solutions of router vendors or educational institutions and then put forward to the IETF to make them standards (RFC).

Section 5.3.1 discusses Arequipa, an extension to Classical IP over ATM, which allows applications to request their own SVC with guaranteed QoS by bypassing the IP layer during connection establishment.

The rest of Section 5.3 discusses two of the various solutions of how to use the fast Layer 2 label switching of ATM in conjunction with network layer routing. The basic idea behind all of these technologies is to increase the packet forwarding performance of routers by replacing slow and expensive network layer forwarding decisions with fast, low cost Layer2 label-swapping based forwarding (cut-through packet forwarding) while at the same improving routing functionality, scalability and flexibility. If these technologies are seamlessly deployed in an ATM based network, end-to-end ATM VC connection with guaranteed QoS can be established for IP traffic, without having to use ATM addressing, routing and signalling like in the overlay model. The IETF Working Group MPLS (Multiprotocol Label Switching) is currently working on unifying and generalising the different approaches which vary in such things as the type of used labels, the trigger event for label binding, the way how labels are distributed in the networks and the protocols they support. Section 5.3.2 introduces the concept of IP Switching because this was the first proposal in this area and Section 5.3.3 presents Tag Switching, which is probably the most advanced solution in this area today. Other related approaches like Cell Switch Router (CSR, Toshiba), Aggregate Route-Based IP Switch (ARIS, IBM) or Switching IP Through ATM (SITA, Telecom Finland) are not discussed here.

Arequipa extension to Classical IP over ATM

General description

The Arequipa (Application REQuested IP over ATM) protocol is a mechanism which allows IP based applications to request their own SVCs with guaranteed QoS. It was developed by EPFL (a member of the ACTS-EXPERT project) as an extension to CLIP, and is described in RFC2170 [9].

Arequipa is a mechanism which allows applications to establish end-to-end ATM connections under their own control, and to use these connections at the lower protocol layer to carry the IP traffic of specific sockets, as illustrated in Figure E.17.

Unlike the connections set up by Classical IP over ATM or by LANE, which are shared by the entire IP traffic flow between the connection endpoints, Arequipa connections are used exclusively by the applications that requested them. The

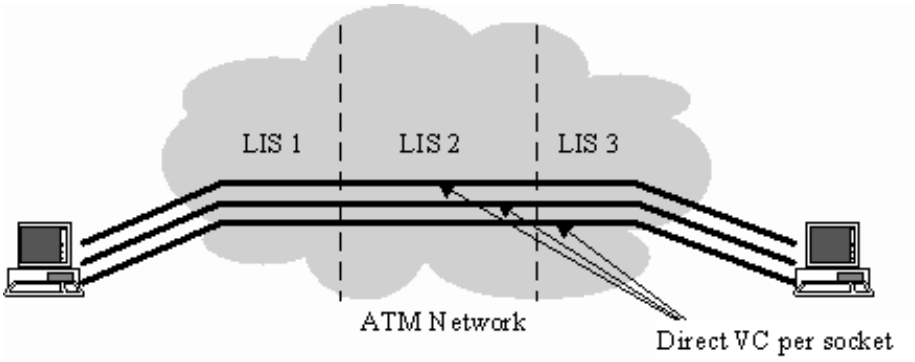


Figure E.17: Arequipa established VCs across LIS borders.

applications can therefore exactly determine what QoS will be available to them.

Figure E.17 illustrates that Arequipa connections are end-to-end, despite the LISs topology, in line with the extensions to IP architecture described in [129]. It shows also that each flow has its own connection with QoS requirements.

In its broadest sense, Arequipa offers the means to use properties of a network technology that is used to transport another network technology (e.g. IP on ATM) without requiring the explicit design and deployment of sophisticated interworking mechanisms and protocols.

Traditional protocol layering typically only allows access to functionality of lower layers if upper layers provide their own means to express that functionality. This approach can introduce significant complexity if the semantics of the respective mechanism are dissimilar. Also, if the upper layer fails to provide that interface, no direct access is possible and the lower layer functionality may be wasted or used in an inefficient way (e.g. if using heuristics to decide on the use of extra features). This is apparent in the case with the QoS functionality of ATM which is hidden by the IP layer when IP is run on top of ATM. Arequipa enables applications to exploit the hidden properties of lower layers by allowing applications to control them directly.

It is important to note that Arequipa coexists with "normal" use of the networking stacks, i.e. applications not requiring Arequipa do not need to be modified and they will continue to use whatever other mechanisms are provided. Moreover, although traffic between applications using Arequipa does not pass the normal routed

IP path anymore; general IP connectivity may still be necessary, e.g. for ICMP messages or for traffic of other applications.

Protocol overview

Arequipa provides two new socket primitives to applications:

- Arequipa_preset(): opens an end-to-end SVC and sends all data from the socket over that connection
- Arequipa_expect(): allows incoming Arequipa connections in the reverse direction

Typically the server side of the application opens and binds a socket and then calls Arequipa_expect(), preparing the socket for incoming Arequipa connections. The client side opens a socket, calls Arequipa_preset() with the desired QoS and the server's ATM address and port number and then connects the socket.

Note that in order to establish the direct VC connection, the ATM address and port number of the server has to be known.

In the protocol stack Arequipa can be seen as a device. Figure E.18 shows the protocol stack for Arequipa and the interaction for a Arequipa_preset call.

Applicability

Arequipa is applicable, for IP and ATM, if the following two conditions are met:

- applications can control "native" connections over the lower layer communication media, that is that there has to be a signaling API which can be used by an application
- both IP and ATM allow communication between the same endpoints (or they share at least a useful common subset of reachable endpoints)

The next two conditions do not have to be met, but without them the use of Arequipa may be questionable:

- all IP traffic between a pair of hosts typically shares the same ATM SVC

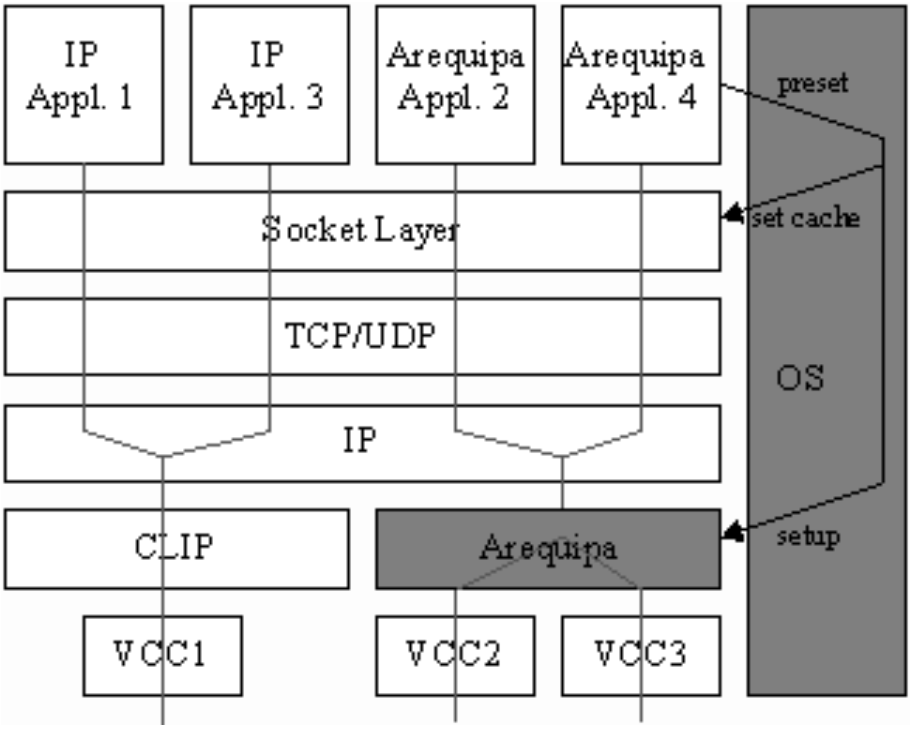


Figure E.18: Arequipa in the protocol stack.

- multiple lower layer connections are possible between a pair of endpoints

In order to simplify interaction with the protocol stack, Arequipa assumes that data sent to destinations for which no Arequipa lower layer connection has been established will be delivered by some default mechanism.

Note that despite its name (Application REQuested IP over ATM), Arequipa is not only limited to IP and ATM. The upper layer is typically IP or some similar protocol (e.g. IPX). The lower layer can be ATM, Frame Relay, N-ISDN, etc.

Application changes

TCP/IP based applications have to be slightly changed in their socket opening behaviour to enable them to run over Arequipa. Basically all that has to be changed is the calling of new socket functions Arequipa_preset() and Arequipa_expect().

There are already two Arequipa based applications publicly available to demonstrate the power of the Arequipa approach. A Web-over-ATM application and an

Arequipa-enabled version of Vic (video conferencing Mbone tool) both written by EPFL, which allows HTML pages to be downloaded with QoS guarantees.

Assessment

Arequipa has the following advantages:

- Arequipa enhances CLIP to allow IP based applications to make full use of ATM's QoS guarantees by allowing them to set up and control their own VC connections.
- Arequipa is a rather light software that only needs to be run on hosts and needs no network support like NHRP or RSVP.
- By establishing direct end-to-end connections routing overhead can be avoided.
- Arequipa is a solution that works and is available today.
- Arequipa is co-existent with the normal CLIP stack allowing "normal" and "Arequipa enhanced" applications to run simultaneously.

The only disadvantage in using Arequipa can be seen to be the fact that existing IP applications need to be "Arequipa enhanced" to be able to take full advantage of its features, though software changes are only minimal.

IP Switching

General Overview

An IP Switch is a hybrid between an ATM Switch and a gigabit router. Datagram forwarding is handled by an ATM switching fabric (as opposed to a router backplane) and routing is performed by traditional router software on an IP switch controller (Figure E.20).

By using the high performance, low cost switching hardware of ATM together with the simple, well tuned IP software for addressing and routing, IP Switching combines the strength of both technologies (Figure E.20).

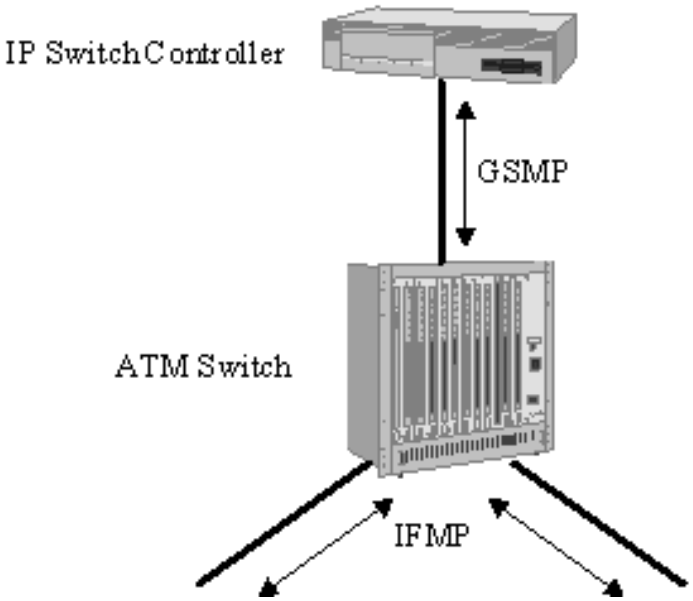


Figure E.19: IP Switching Architecture.

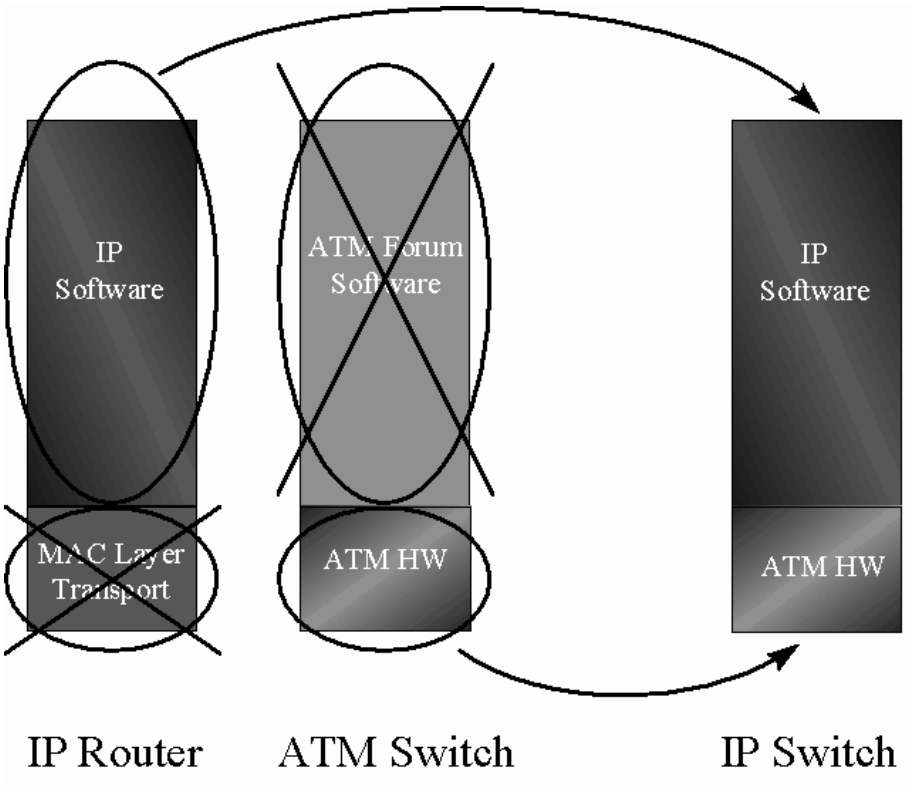


Figure E.20: IP Switching concept.

IP Switching uses flow classification to optimise the load on the IP switch controller. A flow is an extended IP conversation. More specifically, a flow is a sequence of IP packets sent from a particular source to a particular destination sharing the same protocol type (such as UDP or TCP), type of service, and other characteristics, as determined by information in the packet header. The switch controller identifies longer duration flows, as these can be optimised by cut-through switching in the ATM hardware. The rest of the traffic continues to receive the default treatment - hop-by-hop store-and-forward routing.

Flow Classification

The main task of the flow classification process is to select those flows that are to be switched in the ATM switch, and those that should be forwarded packet by packet by the IP switch controller. The decision to switch flows directly through the ATM switch is called short-cut routing. Long duration flows are well adapted for such a short-cut routing. Short duration flows should be handled directly by the forwarding engine of the IP switch controller. Application information provides an approximate indication for flow duration. Multimedia traffic (voice, image, video-conferencing) is an example of long duration flows, whereas name server queries, are typically of short duration.

For the flows selected for short-cut routing, a VC must be established across the ATM switch and the association of flow and VCI label has to be communicated to the upstream IP switch in order that this switch can use a short-cut route. The Ipsilon Flow Management Protocol is a means to communicate this information, another solution would be to use RSVP.

Ipsilon Flow Management Protocol (IFMP)

IFMP [132][133] enables communications between multiple IP Switches or between hosts and IP Switches. It associates IP flows with ATM virtual channels and defines the format for flow-redirect messages and acknowledgements. IFMP is implemented in end stations, such as routers, shared-media hubs, LAN switches, or TCP/IP hosts equipped with an ATM NIC to connect directly to an IP Switch. On

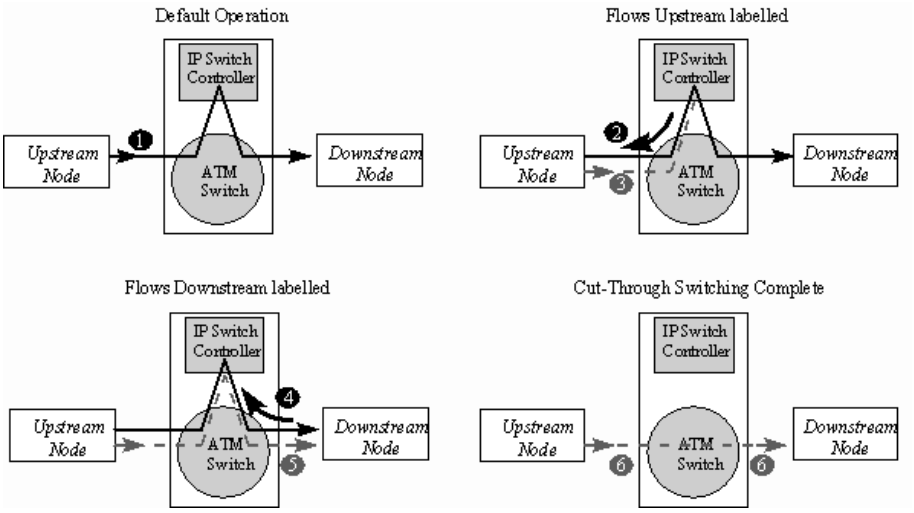


Figure E.21: From default routing to cut-through ATM connection.

ATM links it uses a default VC (VPI 0, VCI 15). The ATM VCI for a specific IP flow is selected by the receiving end of the link. All packets of flows that have not been switched are forwarded hop-by-hop between IP switch controllers using the default VC.

At system start-up, each IP node sets up a virtual channel on each of its ATM physical links to be used as the default forwarding channel. IP data traffic from existing network devices flows into an upstream host, edge router, or IP Switch gateway equipped with an ATM network interface card (NIC) and IP Switching software.

An ATM input port inside the IP Switch receives incoming traffic from the upstream device on the default channel and sends it to the intelligent routing software of the IP Switch Controller (1). The ATM switch hardware functions simply as a high speed I/O extension of the routing software. The IP Switch Controller forwards the packet in the normal manner over the default forwarding channel. It also performs flow classification, a decision-making process that enables IP Switches to optimise data traffic. Once a flow is identified, the switch controller asks the upstream node via IFMP to label that traffic using a new virtual channel (2). If the upstream node concurs, it selects a new virtual channel and the traffic starts to flow on this virtual channel (3). Independently, the downstream node can also ask the IP

Switch Controller to set up an outgoing virtual channel for the flow (4). When the flow is isolated to a particular input channel and a particular output channel (5), the IP Switch Controller instructs the switch to make the appropriate port mapping in hardware, bypassing the routing software and its associated processing overhead (6). This design allows IP Switches to forward packets at rates limited only by the aggregate throughput of the underlying switch engine. First-generation IP Switches support up to 5.3 million PPS throughput. Further, because there is no need to re-assemble ATM cells into IP packets at intermediate IP Switches, throughput remains optimised throughout the IP network.

General Switch Management Protocol (GSMP)

The control protocol used between the IP switch controller and the ATM switch is the General Switch Management Protocol (GSMP) [134]. This allows IP switching to be used with ATM switches from different suppliers. Different ATM switches are designed with different size, cost and functionality trade-offs, so a choice has to be made. GSMP can also support a standard ATM Forum control protocol stack instead of the IP switch controller software. Thus, a choice of network control software is possible for the same hardware.

GSMP is a simple master-slave, request-response protocol, and the switch issues a positive or a negative response, when the operation is complete. Unreliable transport is assumed between controller and switch for speed and simplicity. All GSMP messages are acknowledged, and the implementation handles its own retransmission.

GSMP runs on the default VC (VPI 0, VCI 15) over AAL 5 with LLC/SNAP encapsulation. The most frequent messages (connection management) are designed to fit into single cell AAL 5 packets. An adjacency protocol is used to synchronise states across the control link and to discover the identity of the entity at the far end of the link. There are five types of message: configuration, connection management, port management, statistics, and events.

GSMP has been implemented on at least eight different ATM switches. The code for the GSMP slave is about 2000 lines. A reference implementation is available. The measured performance of the GSMP slave on Ipsilon's IP switch is just under 1000

connection setups per second. This could be improved by hardware SAR support.

Assessment

IP Switching is describing an optimised and scalable way of supporting IP over ATM. It makes use of the strength of both ATM and IP to increase the throughput of the Internet: ATM hardware offers fast speeds at relatively low prices; IP routing is much simpler than the complicated ATM addressing, routing and signalling protocols defined by the ATM Forum (UNI, P-NNI). Persistent flow traffic (e.g. file transfer) is typically worth the connection establishment delay and ATM overhead because once the direct VC is established only fast cell switching is performed by the network node without having to reassemble and analyse IP datagrams for routing. On the other hand, the delay and overhead of establishing an ATM connection does not make sense for short duration, non-persistent data flows (e.g. DNS lookup), which consists only of a few datagrams, where normal IP datagram routing is much better suited.

End-to-end QoS can in principle be achieved in a homogeneous, IP Switching equipped network. However QoS is only expressed with a priority for a flow and not with the usual ATM parameters for QoS. Furthermore it is important to note here, that it is not the application itself but the network that initiates the connection setup. This means that an application has no means to request a special QoS.

Tag Switching

Tag Switching is a proprietary proposal by CISCO [135], [136]. Its objective is to increase router performance in WANs (for example in the global Internet or in the backbone of ISPs) by reducing the complexity of packet forwarding while providing better scalability and richer functionality to network layer routing. Unicast packet forwarding in an IP router involves searching in a table of IP address prefixes (called network layer reachability entries) for the prefix, which has the longest match. Tag switching aims at replacing this operation as much as possible by a simple fixed length label lookup in hardware, exactly as is done with ATM or Frame Relay. This improves packet forwarding performance and introduces new functionality, increased

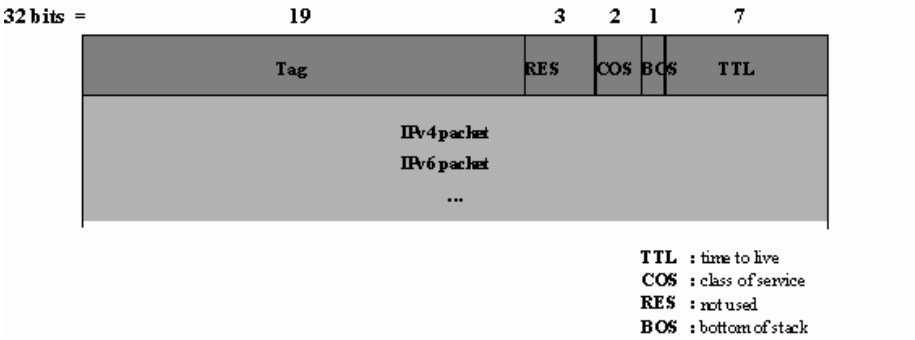


Figure E.22: A tag forma.

scalability and more flexibility in the network layer routing.

Tag Switching consists of two components, the forwarding component, that uses the tag information in the packets and the Tag Information Base in the switch to perform fast packet forwarding, and the control component which is responsible for tag creation and distribution.

Tag Switching is not restricted to use IP as network layer protocol and ATM on Layer 2 but is a general approach applicable to any network layer and Layer 2 protocol.

Tags

Tags are short, fixed length labels, enabling Tag Switches to do simple and fast table lookups in hardware. Tag Switching does not define its own packet format it only adds a tag to an existing packet format. The tag information can be carried in a packet in a variety of ways. For example a 32-bit tag is added in front of a network layer package, which could be IPv4, IPv6, Appletalk or another format. Figure E.22 shows this tag format. A tagged packet is carried on any layer 2 mechanism (e.g.: Ethernet, ATM) and is identified by a layer 2 protocol type (i.e., there would be an Ethertype defining unicast tagged packets, and another Ethertype for multicast packets). A minor difference compared to the 'tags' used in ATM and Frame Relay is the presence of a time-to-live field, which allows using normal IP routing for tag distribution.

Given the variety of ways to carry tag information enables the use of Tag Switch-

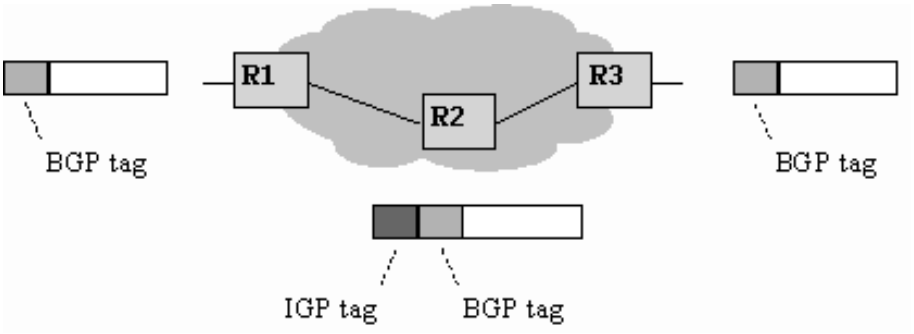


Figure E.23: Stacked tags.

ing over any kind of media.

Tags may optionally be stacked. This enables aggregation of traffic flows. It can be used to speed up packet processing in backbones, and also to scale reservation mechanisms. Figure E.23 shows a possible use of stacked tags in the Internet. Tag switch R1 adds an IGP tag to a BGP tagged packet to route it inside the domain. Tag switch R2 makes its forwarding decision solely on the IGP tag. Tag switch R3, the egress router of the domain, removes the IGP tag.

Forwarding Component

The forwarding component of a tag switch is based on the notion of label swapping. Every tag switching node maintains a Tag Information Base (TIB), which is similar to ATM label swapping tables. If an incoming packet is tagged, then the Tag Information Base is searched for an exact matching entry. If one is found, then the Tag Information Base entry indicates the outgoing interface to which the packet should be forwarded, and the value of the new tag to be used. Unlike with ATM switching, if no entry is found, then the network layer information contained in the packet is used.

It is important to note that the forwarding component of Tag Switching is network layer independent.

Control Component

The control component of a tag switch is responsible for creating and distributing the tag binding information among tag switches.

In contrast to IP Switching where tag bindings are triggered by the detection of a persistent data flow (data traffic driven) Tag Switching uses topology driven tag binding, which means that a tag switch is populating its TIB with incoming and outgoing tags for all routes to which it has reachability.

Tag Switching supports a wide range of forwarding granularities to supports a wide range of forwarding granularities to provide good scaling characteristics and accommodate diverse routing functionality: at one extreme a tag could be bound to a group of routes, at the other extreme a tag could be bound to an individual information flow.

There are three permitted methods for tag allocation and TIB management:

- downstream tag allocation
- downstream tag allocation on demand
- upstream tag allocation

In downstream allocation a switch is responsible for creating tag bindings that apply to incoming data packets and receives tag bindings for outgoing packets from its neighbours (see Figure E.24 (top)). Upstream allocation is the other way round.

There are two families of methods for tag distribution, namely tag distribution by explicit reservations and tag distribution based on destinations.

In tag distribution by explicit reservation, tags are distributed along with the reservation mechanism; if RSVP is used, then the value of the tag is part of the RESV message. This is very similar to the connection setup mechanism of ATM.

In tag distribution based on destination, the tags are distributed by the routing protocol. For this purpose, the tag switches also have to be routers for the protocols they support (IPv4, IPv6, Appletalk, etc.). Routing protocols are used to write the prefix entries, which are then associated with tags. Routing updates may piggyback the tags (distance or path vector protocols), or the tags may be distributed by

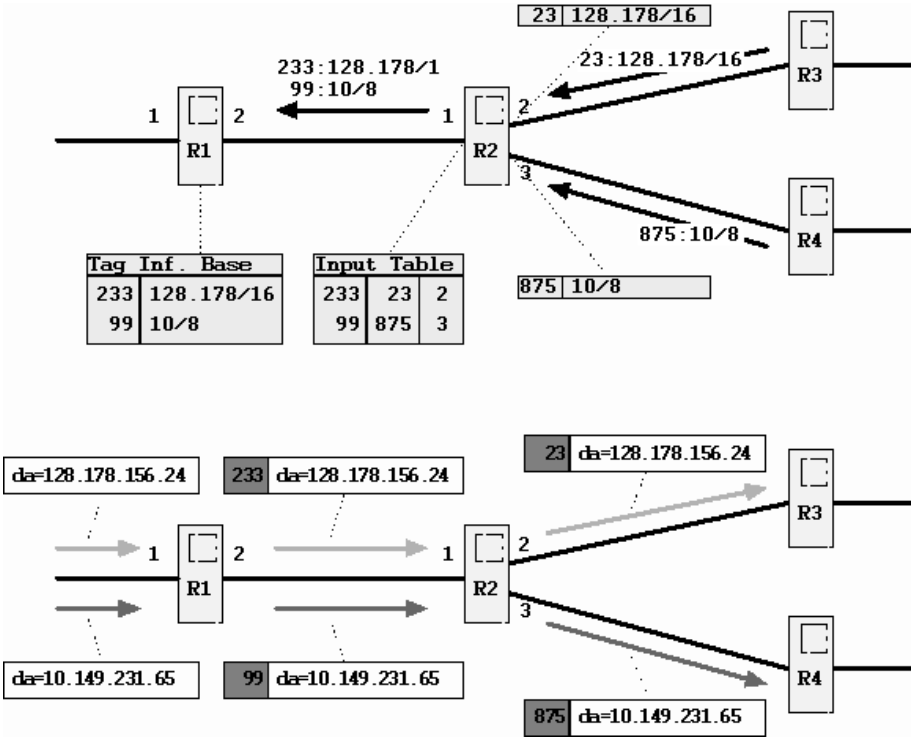


Figure E.24: Tag distribution by routing updates (top) and forwarding of tagged packets (bottom).

a separate protocol called Tag Distribution Protocol [137] (link state protocols). Binding tag distribution together with routing is much simpler than using the overlay model (like in IP over ATM). The presence of the TTL field in the tag avoids problems of temporary loops.

Figure E.24 shows an example of tag distribution with a distance vector protocol and IPv4 address formats. It also shows the resulting Tag information Bases and the forwarding of tagged packets.

Tag Switching with ATM

The characteristics of ATM switches require some specialised procedures and conventions to support tag switching (see [138]).

- Tags can be carried in the VCI field of ATM cells, or if two levels of tagging are needed in the VCI and VPI fields.
- The downstream on demand tag allocation procedure is used.

- ATM switches need to implement the control component of Tag Switching, have to actively participate as a peer in the network layer routing protocol and may need to support network layer forwarding.
- ATM tag switches are only allowed to be interconnected over conventional ATM switches if VP connections are used (only one level of tagging).
- To avoid cell interleaving an ATM tag switch needs to have several tags allocated with one route.
- The existence of the tag switching control component on an ATM switch does not preclude the ability to support the ATM control component defined by the ITU and ATM Forum on the same switch and the same interfaces. The two control components, tag switching and the ITU/ATM Forum defined, would operate independently.

Assessment

Tag Switching is a very powerful way of integrating the fast forwarding of cell-switching technologies with the simple addressing and routing of frame-switching technologies. The simplicity of the Tag Switching forwarding paradigm improves forwarding performance, while maintaining competitive price/performance. By associating a wide range of forwarding granularities with a tag , a wide variety of routing functions (destination based routing, multicast routing, QoS-based routing, hierarchy of routing knowledge) can be supported.

Tag Switching differs from IP Switching in that tags are never allocated based on flow analysis but based on the network topology. Because the network topology is quite static, topology-based tag allocation has a performance advantage over flow-based allocation. Another difference to IP Switching is that Tag Switching is a multiprotocol technology, which is neither bound to a particular network layer nor to a particular data link layer

If Tag Switching is used with IP and ATM, the whole huge ATM control plane as defined by the ATM Forum or ITU (UNI, P-NNI, etc.) can be replaced by the much simpler control component of IP Switching. A drawback of using Tag Switching with

ATM is that the ATM tag switches have to participate as a peer in the network layer routing protocol and may even need to support network layer forwarding. If ATM Tag Switching is used in conjunction with a reservation protocol like RSVP it is possible to provide VC connections with guaranteed end-to-end QoS for IP flows or even applications in a homogeneous network.

Tag Switching is mainly a backbone technology, which is well suited for Internet Service Providers to efficiently route their Internet traffic across a high speed switching technology such as ATM.

Security and charging issues were not yet addressed in Tag Switching but they depend heavily on the used protocols.

Tag Switching is defined in a series of RFC and IETF drafts and is one of today's hottest topics in networking. Cisco announced the availability of a commercial implementation of Tag Switching by autumn 97.

E.6 Summarising Table

The following table summarises some of the technical details of the discussed technologies.

Table 2 can be used to compare the potential of the discussed technologies to satisfy the requirements of an integrated services technology.

E.7 Conclusion

In this chapter we gave a technical overview on the competing integrated services network solutions, such as IP, ATM and the different available and emerging technologies on how to run IP over ATM networks, and identified their potential and shortcomings of being a solution for an integrated services network.

It remains the question, which role these networking technologies will play in the future. We try to answer this question for the short term and the medium-long term.

	IPv4	IPv6	RSVP (+IPv6)	ATM	LANE (+IPv4)	CLIP (+IPv4)	NHRP (+CLIP)	MPOA (+IPv4+NHRP)	Arequipa	IP Switching (IPv4)	Tag Switching (ATM/IPv4)
Classification											
Pure IP Solution	x	x	x								
Pure ATM Solution				x							
IP over ATM Solution (overlay model)					x	x	x	x	x		
Label Switching Solution										x	x
Native ATM support						x	x	x	x	x	x
Emulation of LAN					x			x			
Multiple Layer 3 supported								x			x
Multiple Layer 2 supported	x	x	x				x				x
Network Scope											
Local Area Networks (LAN)	x	x	x	x	x	x	x	x	x		
Wide Area Networks (WAN)	x	x	x	x			x	x	x	x	x
Addressing											
IP addressing on application level	x	x	x		x	x	x	x	x	x	x
E.164/NSAP addressing on application level				x					x		
IP --> ATM address translation						x	x	x	x		
IP --> MAC address translation	x	x	x		x			x			
MAC --> ATM address translation					x			x			
User Data Encapsulation											
IP Packets	x	x	x		x	x	x	x	x	x	x
LLC/SNAP Packet encapsulation	x	x	x		x	x	x	x	x	x	x
MAC Packet encapsulation	x	x	x		x			x			
AAL5 Packet encapsulation				x	x	x	x	x	x	x	x
Data connection											
Connectionless	x	x	x							x	x
ATM permanent VCs				x	x	x		x		x	x
ATM switched VCs (Signalling used)				x	x	x	x	x	x		
End-to-end ATM connection in subnet				x	x	x	x	x	x	x	x
End-to-end ATM connection across subnets				x			x	x	x	x	x
Traffic Type											
Best Effort / UBR	x	x	x	x	x	x	x	x	x	x	x
Priority Based		x	x							x	x
ABR				x	x			x	x		
CBR				x		x	x	x	x		

Figure E.25: Table 1: Technological details.

	IPv4	IPv6	RSVP (+IPv6)	ATM	LANE (+IPv4)	CLIP (+IPv4)	NHRP (+CLIP)	MPOA (+IPv4+NHRP)	Arequipa	IP Switching (IPv4)	Tag Switching (ATM/IPv4)
Quality of Service Support											
Best effort delivery (also UBR)	x	x	x	x	x	x	x	x	x	x	x
QoS for IP flows in subnetwork			x	n/a	x	x	x	x	x	(x)[2]	(x)[2]
QoS for IP flows across subnetwork			x	n/a			x	x	x	(x)[2]	(x)[2]
QoS on application level in subnetwork			x	x		(x)[1]	(x)[1]	(x)[1]	x	(x)[2]	(x)[2]
QoS on application level across subnetworks			x	x			(x)[1]	(x)[1]	x	(x)[2]	(x)[2]
Addressing Flexibility											
Unicast	x	x	x	x	x	x	x	x	x	x	x
Multicast	x	x	x	x				x		x	x
Anycast		x	x	x				x			
Deployment											
ATM Hardware				x	x	x	x	x	x	x	x
ATM Signalling Software				x	x	x	x	x	x		
ATM Routing Software				x	x	x	x	x	x		
IP Hardware	x	x	x		x	x	x	x		x	x
IP Signalling Software			x								
IP Routing Software	x	x	x		x	x	x	x		x	x
Integrates with legacy IP network equipment	x	x	x		x	x	x	x		x	x
Charging Support											
Flat rate network charging	x										
Traffic based network charging		x	x	x	x	x	x	x	x	x	x
Universal Connectivity											
Today	x	x	x		x	x	x	x		x	x
Future		x	x	x	x	x	x	x	x	x	x
Secuity Support											
Network Layer Security		x	x		(x)[1]	(x)[1]	(x)[1]	(x)[1]	(x)[1]	(x)[1]	(x)[1]
Availability Today											
Draft-Standards			x				x	x			x
Standards	x	x		x	x	x			x	x	x
Implementations available	x	x	x	x	x	x			x	x	(x)[3]
Applications available	x	x	x		x	x	x	x	x	x	x

[1] if IPv6 is used
[2] not yet supported but possible
[3] 3 Q 97

Figure E.26: Table 2: Meeting the requirements of an integrated services network.

E.7.1 Short term

Due to its wide diffusion, IP is keeping its leading position on the network layer and ATM is only going to be used as a transport network because of the following reasons:

- There is a vast base of Internet equipment deployed in WAN and LAN area
- ATM deployment is scarce and mainly concentrated in the campus, backbone/WAN area
- There is an unmatched variety of applications based on IP
- There are hardly any applications available that make full use of the superior ATM features

In particular for the users the migration overhead and cost are very relevant. This means, that the overlay model solutions like LANE and CLIP are playing an important role in the deployment of ATM in LANs and backbones. Especially LANE's excellent potential of interconnecting and thus re-using legacy LAN equipment will make it the first choice of corporate network providers and ISPs who are willing to introduce ATM.

Using LANE or CLIP means, that there is no quality of service supported at the application layer, as the IPv4 layer hides all features of the underlying ATM network from higher layers. Only the high speed of ATM is exploited by these technologies.

On the other hand, Internet does not offer today any protocol that permits to obtain QoS, because RSVP is still in a sperimental phase. If QoS support is requested by IP based applications today, a proprietary solution like Arequipa has to be used. Arequipa demonstrated to be a simple solution to access the ATM QoS from IP applications [20], at a minimum cost considering that the necessary software only has to be deployed in the hosts and not in the entire network.

E.7.2 Medium-Long term

Despite the proceeding work of standardisation organisations (ATM Forum, ITU, ETSI), it is not evident whether ATM will ever become an end-to-end solution

because of various reasons:

- IP is extensively deployed (hardware and software)
- ATM software for signalling, routing, management and services is growing very complex and expensive
- Too much overhead to establish VC connections for short duration data flows
- Applications have to be changed considerably to use native ATM
- Full replacement of legacy LAN equipment needed to run end-to-end ATM

It is not clear how much a user is willing to pay in terms of bill for the network usage, complexity of reservation, hardware/software, etc.. to have a better service then that best effort. In many cases they are happy to live with that. In any case it appears very clear that in the future the application will be IP and not many applications that are able to use directly the ATM signalling will be written.

Moreover, IETF's Integrated Services framework (RSVP, ..) is catching up very fast with the ATM technology by introducing reservation, security and charging, and will probably push more and more ATM in a role of transport technology to assume the role of primary network technology.

On the other hand it is not clear when and how QoS support will be introduced in IP. For several people best-effort Internet works perfectly and multimedia applications can be supported by simply increasing the physical capacity of the network.

We can imagine that IP and ATM will have several roles in the future:

- applications will mainly run over IP (with or without QoS)
- ATM will be one of the transmission technologies on the WAN/backbone, but not a networking technology, as well as pure RSVP will be not used. It is in fact more likely, that Label Switching technologies (i.e. IP Switching, Tag Switching) or differential services mechanisms or even some more simple technology will be used in WANs/backbones because they can replace the huge control plane of ATM or the RSVP complexity with much simpler mechanisms.
- RSVP will probably be the main protocol in LAN/campus networks.

- ATM will be a networking technology used with CLIP or LANE inside some private LAN/campus. MPOA could potentially be used in the medium-long term, replacing LANE and CLIP which do not scale very well to large networks and can not offer end-to-end connections, assuming that the highly complex MPOA standard will ever be broadly accepted and implemented.

- ATM will be the networking technology for some niche solution that necessitates imperiously hard service guarantees. In fact the directions followed by IETF is more for a "good"service (something better than best-effort) than for guaranteed services.

Appendix F

Abbreviations

AAL5	ATM Adaptation Layer 5
ABR	Available Bit Rate
ACTS	Advanced Communications Technologies and Services
API	Application Programmer Interface
ARIS	Aggregate Route-Based IP Switch
ARP	Address Resolution Protocol
Arequipa	Application REQuested IP over ATM
ATM	Asynchronous Transfer Mode
BGP	Border Gateway Protocol
B-ICI	Broadband Inter Carrier Interface
B-ISUP	Broadband ISUP
BLLI	Broadband Low Layer Information
BUS	Broadcast and Unknown Server
CBR	Constant Bit Rate
CIDR	Classless Inter-Domain Routing
CLIP	CLassical IP over ATM
CSR	Cell Switched Router
DFFG	Default Forwarder Functional Group
DNS	Domain Name System
DS-1	Digital Signal Level 1
DS-3	Digital Signal Level 3

EGP	Exterior Gateway Protocol
ELAN	Emulated LAN
EPFL	Ecole Polytechnique Federale de Lausanne
ESP	Encapsulating Security Payload
GSMP	General Switch Management Protocol
HTML	Hypertext Markup Language
IASG	Inter Address Sub-Group
ICFG	IASG Coordination Functional Group
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
IFMP	Ipsilon Flow Management Protocol
IGP	Interior Gateway Protocol
IHL	Internet Header Length
InARP	Inverse ARP
ION	IP over NBMA
IP	Internet Protocol
IPng	IP next generation
IPv4	IP version 4
IPv6	IP version 6 (=IPng)
ISP	Internet Service Provider
ITU	International Telecommunication Union
LAG	Local Address Group
LAN	Local Area Network
LANE	LAN Emulation
LE_ARP	LAN Emulation ARP
LEC	LAN Emulation Client
LECS	LAN Emulation Configuration Server
LEC	LAN Emulation Client
LIS	Logical IP Subnetwork
LLC	Logical Link Control

LNNI	LANE Network to Network Interface
LUNI	LANE User to Network Interface
MAC	Media Access Control
MARS	Multicast Address Resolution Server
MCR	Minimum Cell Rate
MPOA	Multi Protocol Over ATM
MPLS	Multiprotocol Label Switching
MSF	Multicast Server Function
MTU	Maximum Transfer Unit
N-ISDN	Narrowband ISDN
NAT	Network Addressing Translation
NBMA	Non-Broadcast Multiple-Access
NHRP	Next Hop Resolution Protocol
NHC	Next Hop Clients
NHS	Next Hop Server
NIC	Network Interface Card
NNI	Network-Node Interface
NSAP	Network Service Access Point
IGMP	Internet Group Management Protocol
ISDN	Integrated Services Digital Network
ISUP	Integrated Services User Part
IPX	Internetwork Packet eXchange
OS	Operating System
OSPF	Open Shortest Path First
PCR	Peak Cell Rate
PNNI	Private NNI
PNO	Public Network Operator
POTS	Plain Old Telephone System
PVC	Permanent VC
QoS	Quality of Service
RARP	Reverse ARP

RFC	Request For Comments
RFFG	Remote Forwarder Functional Group
RSFG	Route Server Functional Group
RSVP	Resource reSerVation Protocol
SAR	Segmentation and Reassembly
SCR	Sustained Cell Rate
SITA	Switching IP Through ATM
SMDS	Switched Multimegabit Data Service
SNAP	Sub Network Access Point
SRP	Scalable Reservation Protocol
SVC	Switched VC
TDP	Tag Distribution Protocol
TIB	Tag Information Base
TCP	Transmission Control Protocol
TOS	Type Of Service
UBR	Unspecified Bit Rate
UDP	User Datagram Protocol
UNI	User-Network Interface
VBR	Variable Bit Rate
VC	Virtual Connection
VCC	Virtual Channel Connection
VCI	Virtual Channel Identifier
VLAN	Virtual LAN
VoD	Video on Demand
VPI	Virtual Path Identifier
WAN	Wide Area Network
WWW	World Wide Web

List of Figures

Bibliography

[1] J.-Y. Le Boudec, "The Asynchronous Transfer Mode : A Tutorial," *Computer Networks, ISDN Systems*, vol. 24 (4), pp. 279–309, May 1992.

[2] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, *RFC2205: Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*. IETF, September 1997.

[3] ITU Telecommunication Standardization Sector - Study group 13, *ITU-T Recommandation Q.2963.1. : Peak cell rate modification by the connection owner*, 1996.

[4] ITU Telecommunication Standardization Sector - Study group 13, *ITU-T Recommandation Q.2963.2. Broadband integrated services digital network (B-ISDN) digital subscriber signalling system No. 2 (DSS 2) connection modification - Modification procedure for sustainable cell rate parameters*, 1998.

[5] R. Guérin and V. Peris, "Quality-of-service in packet networks - basic mechanisms and directions," *Computer Networks and ISDN, Special issue on multimedia communications over packet-based networks*, 1998.

[6] M. Grossglauser, "Controle des ressources de resaux sur des echelles temporelles multiples," *Ph.D. Thesis*, 1998.

[7] J.-Y. Le Boudec, "Network Calculus, Deterministic Effective Bandwidth, VBR trunks," *IEEE Globecom 97*, November 1997.

[8] Vic-Distribution, *VIdeo Conferencing*, 1997. ftp://ee.lbl.gov/conferencing/vic.

[9] W. Almesberger, J-Y. Le Boudec, Ph. Oechslin, *RFC2170: Application REQuested IP over ATM (AREQUIPA)*. IETF, July 1997.

[10] M. N. R. Guérin, H. Ahmadi, "Equivalent capacity, its application to bandwidth allocation in high-speed networks," *IEEE JSAC*, vol. 9 (7), pp. 968–981, 1991.

[11] C. Bruni, P. DÁndrea, U. Mocci, C. Scoglio, "Optimal capacity management of virtual paths in ATM networks," *IEEE Globecom 94*, 1994.

[12] C. Bruni, U. Mocci, P. Pannunzzi, C. Scoglio, "Efficient capacity assignment for ATM virtual paths," *RACE Workshop*, 1995.

[13] C. S. U. Mocci, P. Perfetti, "Vp capacity management in atm networks for short, long term traffic variations," *COST242 TD(95)59*, 1995.

[14] U. Mocci, P. Pannunzzi, C. Scoglio, "Adaptive capacity management in Virtual Paths networks," *IEEE Globecom 96*, 1996.

[15] H. Zhang, E. Knightly, "A New Approach to Support Delay-Sentive VBR Video in Packet-Switching Networks.," *In Proceedings 5th Workshop on Network Operating System Support for Digital Audio and Video (NOSSDAV)*, April 1995.

[16] H. Zhang, E. Knightly, "RED-VBR: A Renegotiation-Based Approach to Support Delay-Sensitive VBR Video," *ACM Multimedia Systems Journal*, 1997.

[17] J. Liebeherr, D. Wrege, "An Efficient Solution to Traffic Characterization of VBR Video in Quality-of-Service Network," *to appear in ACM/Springer Multimedia Systems Journal*, 1998.

[18] J. Wroclawski, *RFC2211: Specification of Controlled-Load Network Element Service*. IETF, September 1997.

[19] S. Shenker, J. Wroclawski, *RFC2216: Network Element Service Specification Template*. IETF, September 1997.

[20] W. Almesberger, L. Chandran, S. Giordano, J.-Y. Le Boudec, R. Schmid, "Using Quality of Service can be simple: Arequipa with Renegotiable ATM connections," *Computer Networks and ISDN Systems*, December 1998.

[21] W. Almesberger, L. Chandran, S. Giordano, J.-Y. Le Boudec, R. Schmid, "Quality of Service Renegotiations," *SPIE Int. Symp. on Voice, Video and Data Communications proceedings, Boston*, November 1998.

[22] W. Almesberger, S. Giordano, J.-Y. Le Boudec, "Reservation Models: From Arequipa to SRP," *submitted to Communication Magazine*, 1998.

[23] W. G-2, "Implementation of ATM Forum & ITU CS2.1 Signalling Functionality, Del. No AC069/EXPERT/WP2/DS/R/P/9/A0," tech. rep., EXPERT Consortium, March 1997.

[24] W. G-2, "Report on Trials of Service Related Control, Del. No AC069/EXPERT/WP2/DS/R/P/18/A0," tech. rep., EXPERT Consortium, March 1998.

[25] The ATM Forum, *ATM User-Network Interface (UNI) Signalling Specification, Version 4.0*, 1996. ftp://ftp.atmforum.com/pub/approved-specs/af-sig-0061.000.ps.

[26] ITU Telecommunication Standardization Sector - Study group 13, *ITU Recommandation Q.2931, Broadband Integrated Services Digital Network (B-ISDN) – Digital subscriber signalling system no. 2 (DSS 2) – User-network interface (UNI) – Layer 3 specification for basic call/connection control*, 1995.

[27] W. Almesberger, J-Y. Le Boudec, Ph. Oechslin, "Arequipa: TCP/IP over ATM with QoS ... for the impatient," Technical Report 97/225, DI-EPFL, CH-1015 Lausanne, Switzerland, January 1997. ftp://lrcftp.epfl.ch/pub/arequipa/impatient.ps.gz.

[28] P. Oechslin-(Editor), "Web over ATM," Tech. Rep. 96/209, DI-EPFL, October 1996.

[29] LRC-EPFL, L. Chandran (Editor), *Web over ATM: Intermediate Report*, 1997. http://lrcwww.epfl.ch/WebOverATM/finaldemo.html.

[30] The ATM Forum, *Traffic Management ABR Addendum*, 1997. ftp://ftp.atmforum.com/pub/approved-specs/af-tm-0077.000.ps.

[31] J. Heinanen, *RFC1483: Multiprotocol Encapsulation over ATM Adaptation Layer 5*. IETF, 1993.

[32] M. Laubach, *RFC1577: Classical IP, ARP over ATM*. IETF, 1994.

[33] The ATM Forum, *LAN Emulation Over ATM, Version 1.0*, January 1995. ftp://ftp.atmforum.com/pub/approved-specs/af-lane-0021.000.ps.

[34] S. Giordano, R. Schmid, R. Beeler, H. Flinck, J.-Y. Le Boudec, "IP, ATM - current evolution for integrated services ," Technical Report SSC/1998/2, DI-EPFL, CH-1015 Lausanne, Switzerland, January 1998.

[35] S. Berson, "RSVP enabled Vic," , 1996. ftp://ftp.isi.edu/rsvp/release.

[36] H. Eriksson, "Mbone: The multicast backbone," *Commun. of the ACM*, pp. 54–60, Aug. 1994.

[37] V. Kumar, *MBone: Interactive Multimedia on the Internet*. Indianapolis, IN: New Riders, 1996.

[38] V. J. S. McCanne, "vic: A flexible framework for packet video," *ACM Multimedia*, 1995.

[39] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, *RFC1889: RTP: A Transport Protocol for Real-Time Applications*. IETF, 1996.

[40] W. Almesberger, *ATM on Linux Distribution*. EPFL, 1997. ftp://lrcftp.epfl.ch/pub/linux/atm/dist.

[41] W. Almesberger, *ATM on Linux*. EPFL, 1996. ftp://lrcftp.epfl.ch/pub/linux/atm/papers/atm_on_linux.ps.gz.

[42] W. Almesberger, "Arequipa: Design, Implementation," Technical Report 96/213, DI-EPFL, CH-1015 Lausanne, Switzerland, November 1996.

[43] P. Oechslin, *Web over ATM*. EPFL, 1997. http://lrcwww.epfl.ch/WebOver-ATM.

[44] DIANA Consortium, *DIANA AC319 home page*, 1998. http://www.telscom.ch/Diana.

[45] S. Giordano, J-Y. Le Boudec, P. Oechslin, S. Robert, "VBR over VBR: the Homogeneous, Loss-free Case," *INFOCOM97*, 1997.

[46] S. Aalto, S. Giordano, "Virtual Trunk Simulation," *in EXPERT ATM Traffic Symposium Proceedings, Mikonos*, 1997.

[47] W. G-3.2, "Initial Specification of Bandwidth Management, Del. No AC069/EXPERT/WP32/DS/R/P/4/A0," tech. rep., EXPERT Consortium, March 1996.

[48] W. G-3.2, "Interim Report on Bandwidth & Routing Trials, Del. No AC069/EXPERT/WP32/DS/R/P/8/A0," tech. rep., EXPERT Consortium, June 1997.

[49] W. G-3.2, "Evaluation of the Bandwidth & Routing Strategies, Del. No AC069/EXPERT/WP32/DS/R/P/4/A0," tech. rep., EXPERT Consortium, March 1998.

[50] ITU Telecommunication Standardization Sector - Study group 13, *ITU Recommandation I.371, Traffic Control, Congestion Control in B-ISDN*, 1995.

[51] L. Zhang, S. Deering, D. Estrin, S. Shenker, D. Zapalla, "RSVP : A New Resource ReSerVation Protocol," *IEEE Network*, September 1993.

[52] C. Topolcic, *Experimental Internet Stream Protocol, Version 2, (ST-II)*. IETF, 1990.

[53] E. Gauthier, S. Giordano, J-Y. Le Boudec, "Reduce Connection Awareness," *High-Speed Networking for Multimedia Applications, W. Effelsberg, O. Spaniol, A. Danthine, D. Ferrari (eds.)*, 1995.

[54] E. Gauthier, J-Y. Le Boudec, "Scalability Enhancements for Connection-Oriented Networks," *International Zurich Seminar on Digital Communications proceedings*, 1996.

[55] The ATM Forum, *P-NNI 1.0 Specification*, 1996.

[56] J. W. P. Varaiya, *High-Performance Communication Networks*. Morgan Kaufmann, San Francisco, October 1996.

[57] C. R. J. Mignault, A. Gravey, "A Survay of Straightforward Statistical Multiplexing Models for ATM Networks," *First International "ATM Traffic Expert" Symposium, Basel*, 1995.

[58] R. Jain, *Congestion Control, Traffic Management in ATM Networks: Recent Advances, A Survey*. ATM Forum, invited submission to Computer Networks and ISDN Systems, January 1995.

[59] J-Y. Le Boudec, P. Thiran, A. Ziedins, S. Giordano , "Multiplexing of heterogeneous VBR connections over VBR trunk," Technical Report 97/232, DI-EPFL, CH-1015 Lausanne, Switzerland, May 1997. http://lrcwww.epfl.ch/PS_files/publications.html.

[60] The ATM Forum, *ATM User-Network Interface Specification, Version 3.1*, 1994.

[61] F. Braun, "Managing the Traffic Streams in a Optimal Way," *ComTec ATM*, 1995.

[62] D. T. P.E. Boyer, "A Reservation Principle with Applications to the ATM Traffic Control," *Computer Networks, ISDN Systems*, vol. 24, pp. 321–334, 1992.

[63] E. Aarstad, "A comment on Worst Case Traffic," *COST 242 TD*, 1992.

[64] D. C. Lee, "Effects of Leaky Bucket Parameters on the Average Queueing Delay: Worst Case Analysis," *IEEE Infocom proceedings*, 1994.

[65] K. S. N. Yamanaka, Y. Sato, "Performance Limitation of Leaky Bucket Algorithm for Usage Parameter Control, Bandwidth Allocation Methods," *IEICE Trans. Communications*, 1992.

[66] S. Giordano, J-Y. Le Boudec, P. Oechslin, S. Robert, "VBR over VBR: the Homogeneous, Loss-free Case," Technical Report 96/199, DI-EPFL, CH-1015 Lausanne, Switzerland, July 1996.

[67] S. Giordano, J.-Y. Le Boudec, "On a Class of Time Varying Shapers with Application to the Renegotiable Variable Bit Rate Service," *submitted to: Journal of High Speed Networks*, 1998.

[68] S. Giordano, J.-Y. Le Boudec, "The Renegotiable Variable Bit Rate Service," *submitted to: IWQoS99*, 1998.

[69] R.L. Cruz, "Quality of Service Guarantees in Virtual Circuit Switched Networks," *JSAC, August 1995*, 1995.

[70] J.-Y. Le Boudec, "Network Calculus Made Easy," Technical Report 96/218, DI-EPFL, CH-1015 Lausanne, Switzerland, December 1996.

[71] C-S. Chang, "On Deterministic Traffic Regulation, Service Guarantees: A Systematic Approach by Filtering," *Submitted to:* , 1997.

[72] R.L. Cruz, C.M. Okino, "Service Guarantees for window flow control," *Proceedings of Allerton Conf, Monticello*, 1996.

[73] S. Giordano, J-Y. Le Boudec, P. Oechslin, S. Robert, "VBR over VBR: the Homogeneous, Loss-free Case," *INFOCOM97*, 1997.

[74] W. P. 4, "Specification of Integrated Traffic Control Architecture," Deliverable Del06, ACTS project AC094 EXPERT, September 1997.

[75] C. Chang and R. L. Cruz, "A time varying filtering theory for constrained traffic regulation and dynamic service guarantees," in *Prepring*, July 1998.

[76] F. Baccelli, G. Cohen, G. J. Olsderand, and J.-P. Quadrat, *Synchronization and Linearity, An Algebra for Discrete Event Systems.* John Wiley and Sons, 1992.

[77] J.-Y. Le Boudec and P. Thiran, "Network Calculus viewed as a Min-plus System Theory applied to Communication Networks," Technical Report 98/276, DI-EPFL, CH-1015 Lausanne, Switzerland, April 1998.

[78] J.-Y. Le Boudec, "Application of Network Calculus To Guaranteed Service," Technical Report 97/251, DI-EPFL, CH-1015 Lausanne, Switzerland, November 1997.

[79] C. Chang, "On deterministic traffic regulation and service guarantee: A systematic approach by filtering," *IEEE Transactions on Information Theory*, vol. 44, pp. 1096–1107, August 1998.

[80] R. Agrawal, R.L. Cruz, C.M. Okino, R. Rajan, "A Framework for Adaptive Service Guarantees," *Proceedings of Allerton Conf, Monticello*, 1998.

[81] S. Giordano, J.-Y. Le Boudec, "QoS based Integration of IP and ATM: Resource Renegotiation," *In Proceedings of 13th IEEE Computer Communications Workshop*, 1998. http://lrcwww.epfl.ch/g̃iordano/publications.html.

[82] W. G-2, "Specification of IP and ATM Technology Integration to Support Quality of Service in Heterogeneous Networks, Del. No AC319/DIANA/WP2/DS/R/P/2/A0," tech. rep., DIANA Consortium, Dec 1998.

[83] A. Ziedinsh and J.-Y. Le Boudec, "Adaptive CAC Algoritms," *in proceedings of ITC 15*, 1996.

[84] C.-Y. Hsu, A. Ortega, "Joint Selection of Source, Channnel Rate for VBR Video Transmission under ATM Policing Constraints," *IEEE Journal on Selected Areas in Communications*, 1997.

[85] J. Wroclawski, *RFC2210: The Use of RSVP with IETF Integrated Services.* IETF, September 1997.

[86] A. Viterbi, "Error Bounds for Convolutional Codes, Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, 1967.

[87] A. Viterbi, "Convolutional Codes, Their Performance in Communication Systems," *IEEE Transactions on Communication Theory*, 1971.

[88] W. Almesberger, S. Giordano, P. Cremonese, H. Flink, J. Loughney,M. Lorang, "A Framework for the QoS Based Integration of IP and ATM in the DIANA Project," Technical Report SSC/1998/028, DI-EPFL, CH-1015 Lausanne, Switzerland, 1998.

[89] S. Shenker, C. Partridge, R. Guérin, *RFC2212: Specification of Guaranteed Quality of Service*. IETF, September 1997.

[90] C. Fogg, "mpeg2encode/mpeg2decode," *MPEG Software Simulation Group*, 1996.

[91] P. Cremonese, "Diana Contribution: ARMIDA4 RSVP-enabled," technical report, ACTS, 1999.

[92] E. Bomitali, S. Dal Lago, G. Franceschini, M. Mesturino, P. Marchisio, G. Venuti, "ARMIDA Multimedia Services," *CSELT ARMIDA Web Server*, 1997.

[93] Digital Audio-Visual Council, *DAVIC 1.0 Specification*, 1985.

[94] International Standard Organisation, *Information technology, Generic coding of moving pictures and associated audio information, Part 1: System*, 1988.

[95] International Standard Organisation, *Information technology, Generic coding of moving pictures and associated audio information, Part 6: Delivery Multimedia Integration Framework*, 1988.

[96] J.-Y. Le Boudec and O. Verscheure, "Optimal Smoothing for Guaranteed Service," Technical Report SSC/98/032, EPFL, October 1997.

[97] S. A. J. Virtamo, "Blocking probability in a transient system," *COST242 TD(97)14*, 1997.

[98] EXPERT Consortium, *EXPERT AC094 home page*, 1997. http://www.elec.qmw.ac.uk/expert/intro.html.

[99] S. Giordano, R. Schmid, R. Beeler, H. Flinck, J.-Y. Le Boudec, "IP, ATM - current evolution for integrated services ," *in Interworkin98 Proceedings, Ottawa*, 1998.

[100] NIG-G3 Chain-Group, *NIG-G3: Internet, ATM coexistence Guideline*, 1998. http://gina.iihe.ac.be/nig-g3.

[101] W. G-2, "Specification of Service Related Control Requirements, Del. No AC069/EXPERT/WP2/DS/R/P/7/A0," tech. rep., EXPERT Consortium, Sep. 1996.

[102] J. Postel, *RFC791: Internet Protocol.* IETF, 1981.

[103] R. Braden, *RFC1122: Requirements for Internet hosts - communication layers.* IETF, 1981.

[104] D. Plummer, *RFC 826: Ethernet Address Resolution Protocol: Or converting network protocol address to 48 bit Ethernet address for transmission on Ethernet hardware.* IETF, 1982.

[105] J. Mogul, *RFC 919: Broadcasting Internet datagrams.* IETF, 1984.

[106] R. Gilligan, E. Nordmark, *RFC 1933: Transition Mechanisms for IPv6 Hosts and Routers.* IETF, 1981.

[107] S. Deering, R. Hinden, *RFC 1883: Internet Protocol, Version 6 (IPv6) Specification.* IETF, 1981.

[108] S. Deering, R. Hinden, *RFC 1884: IP Version 6 Addressing Architecture.* IETF, 1981.

[109] A. Conta, S. Deering, *RFC 1885: Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6).* IETF, 1981.

[110] R. Atkinson, *RFC 1826: IP Authentication Header.* IETF, 1981.

[111] R. Atkinson, *RFC 1827: IP Encapsulating Security Payload.* IETF, 1981.

[112] K. Egevang, P. Francis, *RFC 1631: The IP Network Address Translation (NAT).* IETF, 1994.

[113] L. Zhang, *RSVP: A new Resource ReSerVation Protocol*, 1993.

[114] S. Deering, *RFC 1112: Host Extensions for IP Multicasting.* IETF, 1981.

[115] S. Herzog, *IETF draft: Building Blocks for Accounting and Access Control in RSVP: draft-ietf-rsvp-policy-arch-01.txt.* IETF - INTERNET-DRAFT, 1981.

[116] K.Nichols, V. Jacobson, L. Za, *A Two-bit Differential Services Architecture for the Internet.* Internet Draft, November 1997.

[117] D. Clark, J. Wroclawski, *An Approach to Services Allocation in the Internet.* Internet Draft, July 1997.

[118] W. Almesberger, T. Ferrari, J.-Y. Le Boudec, "Scalable Reservation Protocol (SRP)," Technical Report 97/234, DI-EPFL, CH-1015 Lausanne, Switzerland, January 1997.

[119] W. G-2.1, "Specification of ATM Forum & CS2,1 Signaling Functions, Del. No AC069/EXPERT/WP21/DS/R/P/3/A0," tech. rep., EXPERT Consortium, March 1996.

[120] The ATM Forum, *P-NNI 1.0 Specification*, 1996.

[121] The ATM Forum, *LAN Emulation Over ATM, Version 1.0*, January 1995. ftp://ftp.atmforum.com/pub/approved-specs/af-lane-0021.000.ps.

[122] The ATM Forum, *LANE v. 2.0 LUNI Interface: af-lane-0084.000,*, 1997.

[123] M. Laubach, *RFC1577: Classical IP, ARP over ATM.* IETF, 1994.

[124] M. Perez, F.A. Mankin, E. Hoffman, G. Grosman, A.Malis, *RFC 1755: ATM Signalling Support for IP over ATM.* IETF, 1998.

[125] T. Bradley, C. Brown, *RFC 1293: Inverse Address Resolution Protocol.* IETF, 1992.

[126] IETF - INTERNET-DRAFT, *IETF draft: Multicast Address Resolution Server (MARS): draft-ietf-ipatm-ipmc-12.txt.*

[127] D. P. D. Katz, *RFC2332: NBMA Next Hop Resolution Protocol (NHRP).* IETF, 1998.

[128] J. Lawrance, D. Piscitello, *RFC 1209: Transmission of IP datagrams over the SMDS service.* IETF, 1991.

[129] Yakov Rekhter, Dilip Kandlur, *RFC 1937: Local/Remote" Forwarding Decision in Switched Data Link Subnetworks.* IETF, 1996.

[130] The ATM Forum, *MultiProtocol Over ATM, Version 1.0*, July 1997. ftp://ftp.atmforum.com/pub/approved-specs/af-mpoa-0087.000.ps.

[131] Y. Rekhter, *IETF draft: NHRP for Destinations off the NBMA Subnetwork: draft-ietf-rolc-r2r-nhrp-01.txt.* IETF - INTERNET-DRAFT, 1996.

[132] P.W. Edwards, R.E. Hoffman, F. Liaw, T. Lycon, G. Minshall, *RFC1953: Ipsilon Flow Management Protocol Specification for IPv4, version 1.0.* IETF, 1996.

[133] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. Liaw, *RFC 1954: Transmission of Flow Labelled IPv4 on ATM Data Links, Ipsilon Version 1.0.* IETF, 1996.

[134] P. Newman, W. Edwards, R. Hinden, E. Hoffman, F. Liaw, *RFC 1987: Ipsilons General Switch Management Protocol Specification Version 1.1.* IETF, 1998.

[135] Y. Rekhter, " Tag Switching Overview," *Berkeley EECS Seminar* , 1997.

[136] Y. Rekhter, B. Davie, D. Katz, E. Rosen, G. Swallow, *RFC2105: Cisco System' Tag Switching Architecture Overview.* IETF, 1997.

[137] P. Doolan, B. Davie, D. Katz, Y. Rekhter, E. Rosen, *IETF draft: Tag Distribution Protocol: draft-doolan-tdp-spec-01.txt*. IETF - INTERNET-DRAFT, 1997.

[138] P. Davie, P. Doolan, J. Lawrence, K. McCloghrie, Y. Rekhter, E. Rosen, G. Swallow, *IETF draft: Use of Tag Switching With ATM: draft-davie-tag-switching-atm-01.txt*. IETF - INTERNET-DRAFT, 1997.

Appendix G

List of publications related to this thesis

G.0.3 Papers

E. Gauthier, S. Giordano, J-Y. Le Boudec, "Reduce Connection Awareness," *High-Speed Networking for Multimedia Applications, W. Effelsberg, O. Spaniol, A. Danthine, D. Ferrari (eds.)*, 1995.

S. Giordano, J-Y. Le Boudec, P. Oechslin, S. Robert, "VBR over VBR: the Homogeneous, Loss-free Case," *INFOCOM97*, 1997.

J-Y. Le Boudec, P. Thiran, A. Ziedins, S. Giordano , "Multiplexing of heterogeneous VBR connections over VBR trunk," Technical Report 97/232, DI-EPFL, CH-1015 Lausanne, Switzerland, May 1997, http://lrcwww.epfl.ch/PS_files/publications.html.

S. Aalto, S. Giordano, "Virtual Trunk Simulation," *in EXPERT ATM Traffic Symposium Proceedings, Mikonos*, 1997.

S. Giordano, R. Schmid, R. Beeler, H. Flinck, J.-Y. Le Boudec, "IP, ATM - a position paper," *in EXPERT ATM Traffic Symposium Proceedings, Mikonos*, 1997.

S. Giordano, R. Schmid, R. Beeler, H. Flinck, J.-Y. Le Boudec, "IP, ATM - current evolution for integrated services ," *in Interworkin98 Proceedings, Ottawa*, 1998.

W. Almesberger, S. Giordano, P. Cremonese, H. Flink, J. Loughney,M. Lorang, "A Framework for the QoS Based Integration of IP and ATM in the DIANA Project," Technical Report SSC/1998/028, DI-EPFL, CH-1015 Lausanne, Switzerland, July 1998.

W. Almesberger, S. Giordano, J.-Y. Le Boudec, "Reservation Models: From Arequipa to SRP," submitted to IEEE Communication Magazine, 1998.

S. Giordano, J.-Y. Le Boudec, "QoS based Integration of IP and ATM: Resource Renegotiation," *In Proceedings of 13th IEEE Computer Communications Workshop*, October 1998. http://lrcwww.epfl.ch/g̃iordano/publications.html.

W. Almesberger, L. Chandran, S. Giordano, J.-Y. Le Boudec, R. Schmid, "Quality of Service Renegotiations," *SPIE Int. Symp. on Voice, Video and Data Communications proceedings, Boston*, November 1998.

S. Giordano, J.-Y. Le Boudec, "On a Class of Time Varying Shapers with Application to the Renegotiable Variable Bit Rate Service," submitted to Journal on High Speed Networks, December 1998. http://lrcwww.epfl.ch/g̃iordano/tvShaperv33.ps.

S. Giordano, J.-Y. Le Boudec, "The Renegotiable Variable Bit Rate Service," submitted to IWQoS 99, December 1998. http://lrcwww.epfl.ch/g̃iordano/tvShaperv33.ps.

W. Almesberger, L. Chandran, S. Giordano, J.-Y. Le Boudec, R. Schmid, "Using Quality of Service can be simple: Arequipa with Renegotiable ATM connections," *Computer Networks and ISDN Systems*, December 1998.

NIG-G3 Chain-Group, *NIG-G3: Internet, ATM coexistence Guideline*, 1998. http://gina.iihe.ac.be/nig-g3, work in progress.

G.0.4 Deliverables

W. G-3.2, "Initial Specification of Bandwidth Management,
Del. No AC069/EXPERT/WP32/DS/R/P/4/A0," tech. rep., EXPERT Consortium, March 1996.

W. G-2, "Implementation of ATM Forum & ITU CS2.1 Signalling Functionality,
Del. No AC069/EXPERT/WP2/DS/R/P/9/A0," tech. rep., EXPERT Consortium, March 1997.

W. G-3.2, "Interim Report on Bandwidth & Routing Trials,
Del. No AC069/EXPERT/WP32/DS/R/P/8/A0," tech. rep., EXPERT Consortium, June 1997.

W. G-3.2, "Evaluation of the Bandwidth & Routing Strategies,
Del. No AC069/EXPERT/WP32/DS/R/P/4/A0," tech. rep., EXPERT Consortium, March 1998.

W. G-2, "Report on Trials of Service Related Control,
Del. No AC069/EXPERT/WP2/DS/R/P/18/A0," tech. rep., EXPERT Consortium, March 1998.

W. G-1, "Specification of IP and ATM Technology Integration to Support Quality of Service in Heterogeneous Networks,
Del. No AC319/DIANA/WP1/DS/R/P/2/A0," tech. rep., DIANA Consortium, Dec 1998.

Appendix H

Curriculum Vitae

Education

- I obtained my degree in Computer Science in 1989 from the Computer Science Department of the University of Pisa, Italy. The title of the thesis was: "Recovery Analysis for the Transactional Systems and proposals of new Semantic Knowledge based techniques". Prof. L.Lenzini, Ing. E.Gregori and Dr. A.Bondavalli were my Master's thesis supervisors.

Employment History

- Since 1995 I have worked in the Institute of Communications and Applications (ICA) at EPFL, Lausanne (`http://icawww.epfl.ch/`). I participate in the European ACTS projects EXPERT and DIANA. I am one of the main contributors of the NIG-G3 Chains work. For EPFL, I also participate in the activities of the Quantum and CCIRN programmes, and follow the IETF activities. I am Associated Technical Editor of IEEE COMMUNICATION Magazine of IEEE Communication Society (`http://www.comsoc.org/`).

  - The EXPERT project (`http://www.elec.qmw.ac.uk/expert/`) concentrates on validating traffic engineering aspects for optimising the utilisation of broadband network resources. The project developments enable

trials of cost-effective ATM access and multi-service integration to take place. This project also provides the platform for a number of trials to be performed by other ACTS projects in the fields of Mobile (in a local hospital environment), Multimedia (in the Finnish Multimedia Programme), Charging and Signalling for ATM, telecommunication and CATV network integration, tele-learning and TMN aspects. EXPERT has close associations with EURESCOM and the Swiss, Dutch, Finnish, UK and Danish National Hosts. EXPERT also made application trials with broadband facilities of the "CANARIE" testbeds in Canada and with North Carolina State University.

- DIANA (`http://www.telscom.ch/Diana`) started in March '98 as a new project in the European Union 4th Framework Programme ACTS. The partners in DIANA will develop, integrate, validate and demonstrate resource reservation and traffic control functionality which interoperate seamlessly between ATM and IP networks to achieve end-to-end QoS.

  The protocols used for the exchange of this control information will preferably be RSVP or ATM signalling, however, the design of the trial platform is planned to be generic enough to investigate several solutions for the convergence of IP and ATM, such as Differentiated Services with ATM or others.

- Ni-chains NIG-G3 working group (`http://gina.iihe.ac.be/nig-g3/`) aims to propose Internet and ATM coexistence Guidelines to support business strategists and end-users for evaluating the competing IP/ATM architectures and technologies, helping to clarify, from an unbiased point of view, the state-of-the art.

- The Quantum Test Programme (QTP) `http://www.dante.net/quantum/QTP/` has the objective of testing and validating new technologies, products and services with a view to introducing them into the operational TEN-155 service at some future date.

- CCIRN (Coordinating Committee for Intercontinental Research Networking) (`http://www.ccirn.org/`) provides a forum for members to estab-

lish a set of activities to achieve inter-operable networking services between participating international entities to support open research and scholarly pursuit. Policy, management, and technical issues will be examined, based on agreed requirements.

- From 1991 to 1994, I was working in the Network and Security Group of CSCS (Centro Svizzero di Calcolo Scientifico). In this period I participate to plan, develop and maintain the advanced network of CSCS based on heterogeneous technologies like ATM, IP and protocols like Ethernet, FDDI and HIPPI. I was also involved in the ISUS Working Group of RARE activity, and in the project of an "inter-boundary" high speed network. I also lead the ATM Pilot Project for a Swiss backbone.

- From 1989 to 1991 I worked in the Network and Telecommunication Group of CNUCE (Italian Council Research C.N.R. Institute). During this period, I was involved in the OSIRIDE,ASTRA and EARN projects. I was the OSI expert of the CNUCE Data Transmission and Communication Group, as well as the CNUCE key-person for the GARR-PE commission.