Proc. IEEE GLOBECOM '97, Phoenix, AZ (Nov. 1997).

OCP_A: An Efficient QoS Control Scheme for Real Time Multimedia Communications

Yin Bao and Adarshpal S. Sethi
Department of Computer and Information Sciences
University of Delaware, Newark, DE

Abstract

Multimedia applications communicating over a packet-switched network present new challenges to the current networking technology. One of them is Quality of Service (QoS) requirement. Due to the increasing dynamics of traffic introduced by multimedia applications, dynamic and flexible QoS (Quality of Service) control is needed to ensure both QoS satisfaction and resource efficiency. In this paper, we present an efficient quality assurance scheme to satisfy a diversified combination of delay bound and loss ratio requirements by dynamically adjusting the resource allocation. It is a simple, general QoS control scheme which can be used on a variety of real-time multimedia sources. The scheme is evaluated in extensive simulation studies using real MPEG video sources and shown to be superior to static resource allocation in terms of efficiency, fairness, and flexibility.

I. Introduction

A major goal of current integrated networks is to provide transmission service to diversified applications with different QoS (Quality of Service) requirements. Various QoS control schemes have been proposed and studied [1] — [9]. The emergence of new network applications poses new demands on QoS control. For example, in a real-time MPEG encoded video transmission, the source may generate data in bursts with a high burstiness. Some newly emerging interactive video and the future residential broadband services [10] may be incapable of predicting their own bandwidth requirements. For those applications, it is impossible to set a specific amount of resources aside before the start of transmission to ensure QoS. It is therefore necessary to have *dynamic resource allocation* schemes that operate adaptively according to the variation of traffic characteristics over the lifetime of a transmission to ensure the twin objectives of QoS satisfaction and network efficiency.

For applications such as real-time audio and video, some of the most important QoS requirements are delay bound and loss ratio. Such applications may even require specific end-to-end delay bound and loss ratio QoS. For real-time video transmission, streams with 1% packet loss and streams with 5% packet loss may result in very different subjective effects at playback; a video clip showing a person talking may look better than a clip of MTV subjected to the same loss ratio. Different encoding schemes may also present different QoS requirements. Therefore, sometimes the network should expect to admit applications with *specifically-chosen* QoS requirements. Thus, QoS control should try to provide *accurate* realization of QoS as close as possible to what is requested, not just to a certain degree.

This work was supported in part by the U.S. Department of the Army, Army Research Laboratory under Cooperative Agreement DAAL01-96-2-0002 Federated Laboratory ATIRP Consortium.

The aim of our work is to design a dynamic resource control scheme which can provide accurate QoS realization for real-time applications. In this paper, we will present such a scheme called OCP_A (OCcuPancy_Adjusting). OCP_A is an enhanced dynamic rate-based QoS control scheme based on VirtualClock [5]. Even though most rate-based schemes [3] [6] [8] can adjust the resource weights among the flows dynamically to achieve efficient QoS control for real-time traffic (whose traffic pattern tends to change with time), less work has been done to address *how* the weights should be adjusted. The contribution of our work is to provide a solution for this problem by adjusting the resource share of each flow based on its QoS performance and resource utilization. Due to the focus on QoS performance, our scheme can provide specific delay bound and loss ratio QoS control in a resource-efficient and fair way. The proposed scheme is also adaptive so that it can be applied on bursty and unpredictable real-time transmissions with only minimal knowledge of the traffic characteristics.

The rest of the paper is organized in the following way: Section II describes OCP_A in detail. Section III is devoted to study the performance of OCP_A by comparing it with the original static VirtualClock in extensive simulations performed on various real MPEG video clips from movies, news, and video conferencing applications. Finally, Section IV ends with conclusions.

II. OCP_A (OCcuPancy_Adjusting)

In this section, we start our discussion by introducing the general idea of using dynamic resource allocation adjustment to achieve QoS. Section II-B presents OCP_A for single flow and Section II-C describes OCP_A when applied to multiple flows sharing the resources.

A. Dynamic Resource Adjustment

We adopt the idea of dynamically adjusting the resource allocation of each flow to achieve the requested QoS. Under this scheme, a referenced amount of resources is allocated before the start of transmission, and this amount is adjusted during the lifetime of the transmission based on a set of criteria such as QoS performance and resource utilization. The

gain from this is multi-fold. First is resource efficiency: by dynamically adjusting the resource allocation for each flow at a network node, we may be able to accommodate more flows into the network with satisfactory QoS performance. The second gain is that dynamic resource adjustment provides the possibility for the QoS control scheme to monitor and be responsive to the QoS performance, and thus be able to provide accurate and fair QoS realization. The third gain from dynamic resource adjustment is that due to its self-adjustability, it provides flexibility in handling different types of sources without any a priori knowledge of the traffic generated by the sources.

We use **QoS tuple** $\{D_i, L_i\}$ (D_i is the delay bound and L_i is the loss ratio) to present the QoS requirement of a given flow i. We assume these QoS requirements are pertinent to a *single* internal network node — a switch or a server. There are two situations that could cause packet loss: one is the new arrivals being discarded when the buffer is full, another is the expiration of the deadlines.

We also use **CSE (Current Service Environment)** $\{B_i, \gamma_i\}$ to represent the allocated resource of a switch to satisfy the specific QoS of a flow i, where B_i refers to the allocated buffer size, γ_i refers to the allocated service rate in terms of packets per second. As we can see in the following methods, determining the appropriate CSE dynamically is at the heart of dynamic QoS control. In the whole lifetime of a transmission, there is a sequence of CSE intervals of this flow: $CSE_i(t_1), CSE_i(t_2), ..., CSE_i(t_k), CSE_i(t_{k+1}), ..., CSE_i(t_n)$. The time average of CSE over all CSE intervals can be used as a measure of resource consumption.

In the method that we will describe, the deadlines of the packets are enforced by coordinated CSE, i.e., any CSE intends to guarantee that the packets admitted into the switch will be served within their deadlines. This is done by maintaining $\{B_i, \gamma_i\}$ under the following relationship: $B_i = D_i * \gamma_i$. Because of this relationship, given that flow i is guaranteed a service rate of γ_i packets/sec, any packet that is waiting in a buffer will be guaranteed to be serviced within its delay bound D_i.

B. OCP_A (OCcuPancy_Adjusting)— single flow case

In this section we introduce a dynamic resource allocation adjustment scheme for QoS control—OCcuPancy_Adjusting (OCP_A). It is applied to each single traffic flow passing through a network node. In OCP_A, resource adjustment is responsive to the allocated buffer resource utilization as well as the QoS requirements. Before the start of the transmission, two occupancy thresholds are chosen: OCP_l(the lower threshold) and OCP_u (the upper threshold). Generally, $0 < OCP_l < OCP_u \leq 1$ and $OCP_l < 0.5$. The following is the single flow OCP_A algorithm, providing the QoS tuple of this flow is $\{D_i, L_i\}$:

Algorithm: **Single flow OCcuPancy_Adjusting**

Start with an initial referenced CSE $\{B_i, \gamma_i\}$ *where* $B_i = B_i^{ref}, \gamma_i = \gamma_i^{ref}$;
Serve packets from this flow under the current CSE;
if (there is a new arrival at a full buffer **and** *losing this arrival will cause the loss ratio performance to be worse than* L_i*)*
    increase B_i *till the new buffer occupancy is* OCP_u;
    $\gamma_i \leftarrow B_i/D_i$;
if (the current buffer occupancy is below OCP_l*)*
    decrease B_i *by half down to* B_i^{min};
    perform $\gamma_i \leftarrow B_i/D_i$ *when there are no transit packets* [1];

In the above algorithm, γ_i^{ref} is chosen to be the predicted average traffic rate[2] before the start of transmission and B_i^{ref} to be $D_i * \gamma_i^{ref}$. B_i^{min} is the minimum buffer size that needs to be allocated to this flow at any time. It is decided by γ_i^{min}—the minimum service rate that needs to be reserved for flow i at all time—by the relationship $B_i^{min} = \gamma_i^{min} D_i$. It is our opinion that it is important to allocate at least the minimum CSE $\{B_i^{min}, \gamma_i^{min}\}$ to each flow at any time to avoid future sudden degradation of QoS. Notice that decreasing the buffer size by half is just one of the many ways to decrease the buffer resource allocation.

In the OCP_A algorithm, the occupancy thresholds OCP_l and OCP_u should be chosen carefully to prevent indefinite resource oscillation [11]. Meanwhile, to prevent frequent resource adjustment, a confidence period can be used for decreasing resources: the resources can be only decreased if the buffer occupancy is *consistently* lower than OCP_l for a certain period of confidence time. Simulations have been conducted on various sources to study the effects of different values of confidence time [11]. As a result, it is recommended that the system should always choose the confidence period based on the frequency of the traffic rate change, plus the consideration of the resource efficiency and control overhead trade-off.

C. OCP_A (OCcuPancy_Adjusting) for multiple flows

The single flow OCP_A QoS control algorithm can be easily extended to multiple flows sharing the resources with the assistance of a rate-based scheduling algorithm, such as VirtualClock [5]. VirtualClock schedules packets from different flows according to their VC (VirtualClock) values, which are calculated according to the allocated rates of the flows. The VC value of flow i when its allocated CSE is $CSE_i(t_k)$ (the allocated resource tuple is $\{B_i(t_k), \gamma_i(t_k)\}$) is calculated as

[1] *transit packets*: the packets in the queue at the moment of resource adjustment. Decreasing service rate γ immediately could result in the violation of deadlines of these packets. Therefore, the service rate γ can only be decreased after transmissions of all the transit packets.

[2] *predicted average traffic rate*: the average source generation rate by the best knowledge of the source. The algorithm does not assume this best-knowledge average rate to be accurate.

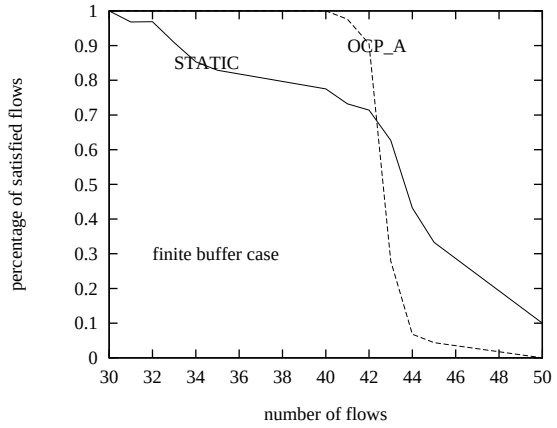

Fig. 1. **QoS satisfaction percentage**

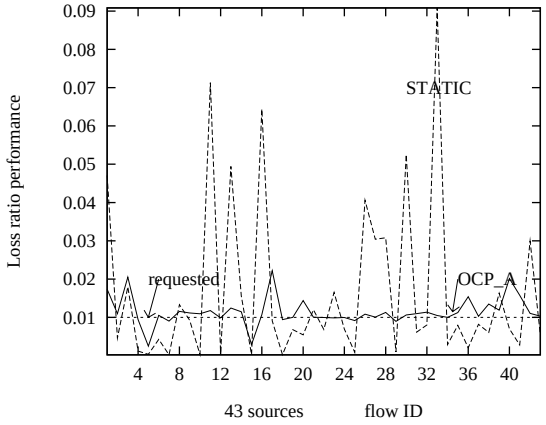

Fig. 2. **Loss ratio performance of each flow**

the following:

$$VC_i = max\{VC_i, realtime\} + 1/\gamma_i(t_k) \qquad (1)$$

Equation (1) is applied for each arrival from flow i. The packet with the smallest VC value among all flows is served. If $CSE_i(t_k)$ $\{B_i(t_k), \gamma_i(t_k)\}$ is changed to $CSE_i(t_{k+1})$ $\{B_i(t_{k+1}), \gamma_i(t_{k+1})\}$, the only change in Equation (1) is to change $\gamma_i(t_k)$ into $\gamma_i(t_{k+1})$. Basically a CSE change only affects the *tick time* of a flow. If a flow i has used more ticks than what is allocated in CSE period $CSE_i(t_k)$, it will have a relatively large VC value as a penalty. It is our opinion that this penalty should be carried to the next CSE period $CSE_i(t_{k+1})$ because the change of CSE is stepwise, which can be considered as a small change of allocated resources. Therefore, the penalty still serves well as a rate meter even across the CSE boundaries.

The combination of OCP_A and VirtualClock algorithms provides us with a dynamic rate-based control method: the OCP_A adjusts the allocated resources according to the QoS performance and resource utilization, while the VirtualClock schedules the packets according to the allocated resources at that time.

III. Performance study of OCP_A

Extensive simulations were conducted to compare the performance of OCP_A with the static resource allocation algorithm in terms of efficiency, fairness, and flexibility. In order to get accurate comparisons, both schemes were applied to each of many simulation data sets. These simulations were done by using Opnet [12] on Sun-SPARC systems. Various types of real MPEG-encoded video sources [13] have been used in the simulations including movie, news, and video conferencing traffic. Combinations of different delay bounds and loss ratio requirements (ranging from 10^{-1} to 10^{-5}) were used in the simulations. A full description of simulation data can be found in [11].

A. Resource Effectiveness and QoS Fairness

In order to compare the resource effectiveness, we used the simulations to determine the percentage of the sources whose requested QoS could be satisfied under different resource allocation schemes given the same amount of total resources. Figure 1 shows the QoS satisfaction percentage for one simulation data set, in which each flow starts its transmission at a random point from the same movie. Therefore they have slightly different average rates but similar statistical traffic patterns. All the sources have the same QoS requirements $\{D = 10ms, L = 10^{-2}\}$. Under the static resource allocation scheme, each flow is allocated the same amount of resources; while under OCP_A, the amount of resources allocated to each flow is adjusted dynamically. Figure 1 plots the relationship between the total number of flows sharing the resources and the percentage of satisfied flows under different resource allocation schemes. It shows that when the QoS satisfaction percentage is large ($> 70\%$), OCP_A performs better than static resource allocation by presenting higher percentage of flows with satisfactory QoS. For example, when the number of flows is 40, OCP_A provides satisfactory QoS to all the flows, while static scheme only provides satisfactory QoS to 77% of the flows. The effectiveness of OCP_A also can be observed by noting that under a fixed QoS satisfaction level, OCP_A can incorporate more flows than the static scheme does.

It is also noticeable that when the QoS satisfaction percentage drops ($< 70\%$), the static scheme gradually out-runs OCP_A on the QoS satisfaction percentage. Figure 2 illustrates the performance of each flow when this happens: even though the satisfaction percentage of OCP_A is low, the loss ratio performances of most flows are just *slightly* over the requested loss ratio; while in the static scheme, there is still a considerable number of sources whose loss ratios are way beyond what was requested. The dropping of satisfaction percentage in OCP_A is because OCP_A is a fairer scheme. When insufficient resources are available to satisfy all active flows, OCP_A compensates by reducing resources allocated to all flows in a fair manner, thereby causing most of them

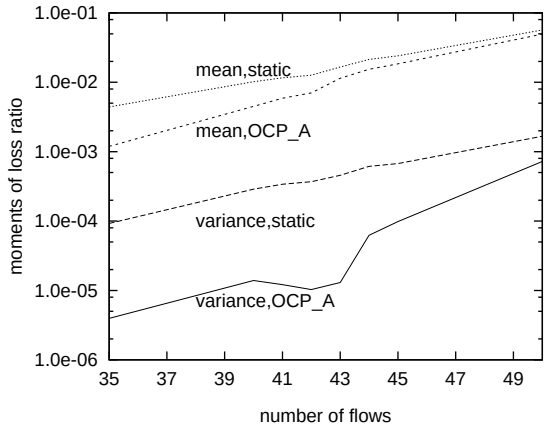

Fig. 3. **Moments of loss ratio**

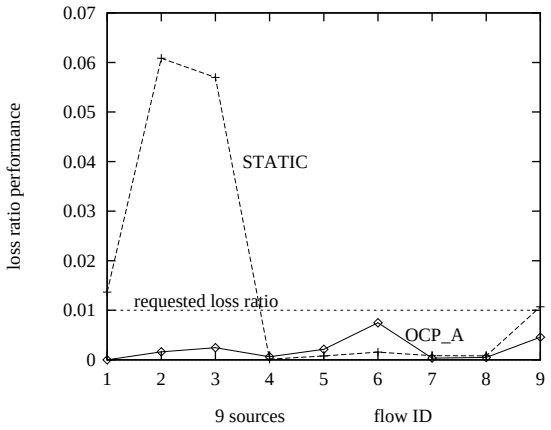

Fig. 5. **Flexibility for diversified sources**

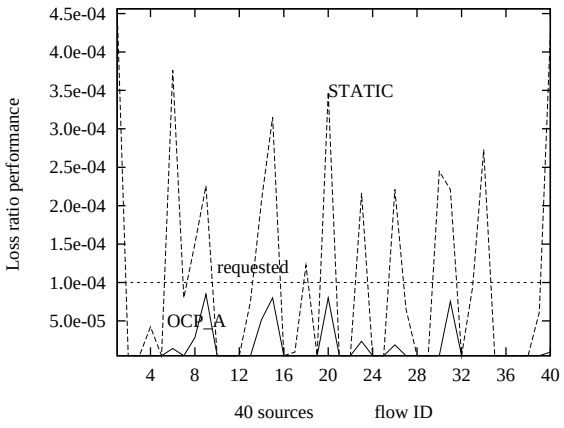

Fig. 4. **Loss ratio performance:** $L = 10^{-4}$

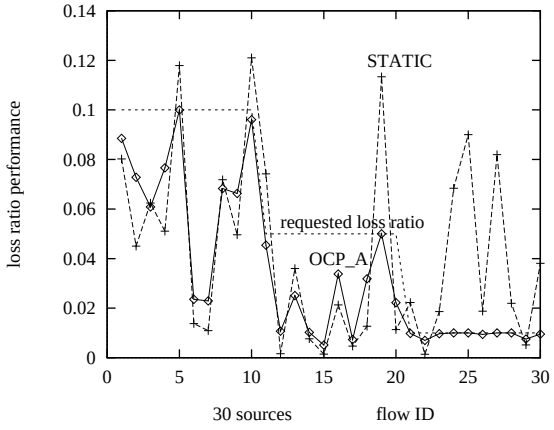

Fig. 6. **Flexibility for diversified QoSs**

to miss their QoS by a small margin. In contrast, static allocation arbitrarily favors a few flows by allowing them to meet their QoS needs while all the other flows are starved and receive very poor services. Figure 3 depicts the loss ratio averages and variances among all the flows when the static scheme or OCP_A is applied: OCP_A constantly presents smaller average (over all flows) loss ratio performance as well as smaller loss ratio variances compared to the static scheme. Figure 4 depicts the loss ratio performance when the requested loss ratio is relatively small: $L = 10^{-4}$. The effectiveness and fairness performance of OCP_A are similar to the previous observation. The reason why OCP_A provides fairer realization of QoS among the flows with same QoS requirements is that it observes the performance in real time and responds to the change actively.

B. Flexibility for Diversified Sources

Due to the increasing diversity of network sources, the flexibility in incorporating different types of traffic is crucial for the viability of a QoS control scheme. In static resource allocation, the amount of resources allocated to a particular source needs to be decided at the start of transmission. This could be a problem even if the average rate of the traffic is known because of the variability of the real-time traffic. On the other hand, a dynamic scheme such as OCP_A is extremely flexible in incorporating diversified sources. Figure 5 shows the loss ratio performance of a simulation data set, in which the 9 sources sharing the resources are a mixture of movie, news, and video conferencing traffic with the same QoS tuple $\{D = 10ms, L = 10^{-2}\}$. Under OCP_A, at the start of transmission, each source is assigned an amount of resources that is approximately comparable to its average traffic rate; this amount of resources is adjusted by OCP_A during the lifetime of the transmission. Under static allocation, we divide the total resources among each source according to the weights of their actual average rates. Figure 5 plots the final achieved loss ratio for each flow. We can see that under the same amount of total resource, all the flows can make their QoS under OCP_A, while some of them cannot under the static scheme.

C. Flexibility for Diversified QoS

Another flexibility feature of OCP_A is to provide QoS control for diversified QoS requirements. Figure 6 shows the simulation results in which there are 30 flows and each of them is a movie source. Flow 1-10, 11-20, 21-30 have dif-

ferent loss ratio requirements—10%, 5%, 1% accordingly. Under OCP_A, despite the differences of QoS requirements, in the beginning of the transmission each source is simply allocated approximately the resources comparable to their average rates. Under the static scheme, we use Bruneel's approximation [14] to calculate the desired amount of resources needed for the specific QoS requirement; and then distribute the amount of resources among all the flows according to the weights of the calculated desired amounts. Figure 6 plots the final achieved loss ratio for each flow. It is apparent that OCP_A enables all the sources to have their QoS satisfied; while under the static scheme, 11 out of 30 sources have their requested QoS violated. In summary, OCP_A provides flexibility to assign the initial amount of resources without knowing much about the requested QoS and the traffic pattern (some basic knowledge such as average rate is helpful for assigning an appropriate initial amount), since the initial amount of resources assigned can be adjusted to achieve the requested QoS. On the other hand, it is very hard or sometimes even impossible for a static scheme to pre-assign the appropriate shares of resources to achieve the same purpose.

IV. Conclusion

In this paper, we introduced a simple dynamic QoS control algorithm called OCP_A. OCP_A monitors the QoS performance of each flow and adjusts its allocated resources accordingly. Because the amount of allocated resources is adjusted dynamically, OCP_A presents better resource efficiency than the conventional static resource allocation scheme. Since OCP_A monitors the QoS performance in real time, it provides fair and accurate realization of the requested QoS, as well as great flexibility in achieving diversified QoS. OCP_A is self-adjustable based on the QoS performance and the source traffic changes, thus can be used as a generic scheme for a variety of real time applications within various network environments. It is also a simple scheme with little control overhead and complexity. All these merits make it an appropriate QoS control scheme for real time multimedia applications, which usually have diversified QoS requirements and whose traffic is unpredictable and bursty.

We also studied the performance of OCP_A when it is applied to achieve end-to-end QoS requirement. To achieve end-to-end QoS: first, an end-to-end QoS distribution scheme is applied to determine the local QoS requirement on each router on the path; then a local QoS control scheme such as static resource allocation or OCP_A is used to achieve the distributed local QoS requirement on each router. Simulation results show that OCP_A presents similar advantages over the static scheme as described previously in achieving end-to-end QoS. Due to space limitation, the performance studies of different end-to-end QoS distribution schemes are not described here.

We are now in the process of designing a coordinated admission control scheme by using the observation of resource availability and QoS performance from OCP_A.

REFERENCES

[1] R. Chipalkatti, J.F. Kurose, and D. Towsley. Scheduling policies for real-time and non-real-time traffic in a statistical multiplexer. In *Proc. IEEE INFOCOM '89*, volume 3, pages 774–783, Ottawa,Canada, April 1989.

[2] A.A. Lazar and G. Pacifici. Real-time scheduling with quality of service constraints. *IEEE Journal on Selected Areas in Communications*, 9(7):1052–1063, September 1991.

[3] S.J. Golestani. Self-clocked fair queueing scheme for broadband applications. In *Proc. IEEE INFOCOM '94*, volume 2, pages 636–646, Toronto, Canada, April 1994.

[4] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. In *Proc. SIGCOMM '89 Symposium: Communications Architectures & Protocol*, pages 1–12, Austin, TX, September 1989.

[5] L. Zhang. Virtualclock: A new traffic control algorithm for packet switching networks. In *Proc. ACM SIGCOMM '90*, pages 19–29, Philadephia, PA, September 1990.

[6] A.K. Parekh and R.G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.

[7] D.D. Clark, S. Shenker, and L. Zhang. Supporting real-time applications in an integrated service packet network: Architecture and mechanism. In *Proc. ACM SIGCOMM '92*, pages 14–26, Baltimore, MD, August 1992.

[8] S.S. Lam and G.G. Xie. Burst scheduling: Architecture and algorithm for switching packet video. In *Proc. IEEE INFOCOM '95*, volume 3, pages 940–950, Boston, MA, April 1995.

[9] R. Bolla, F. Danovaro, F. Davoli, and M. Marchese. An integrated dynamic resource allocation scheme for ATM networks. In *Proc. IEEE INFOCOM '93*, volume 3, pages 1288–1292, San Francisco, CA, 1993.

[10] T. Kowk. A vision for residential broadband services: ATM-to-the-Home. *IEEE Network*, 9(5):14–28, Sept-Oct 1995.

[11] Y. Bao and A.S. Sethi. An efficient, dynamic QoS control scheme to satisfy diversified QoS guarantees. under preparation.

[12] MIL3 Inc. Opnet modeler, v3, 1996. Washington, DC.

[13] O. Rose et al. All MPEG-I traces from ftp: ftp-info3.informatik.uni-wuerzburg.de, /pub/MPEG/traces, more information available in traces.README, September 1995.

[14] H. Bruneel and B.G. Kim. *Discrete-Time Models for Communication Systems Including ATM*. Kluwer Academic Publishers, 1993.