Improving Node Behavior in a QoS Control Environment for Local Broadcast Networks

Bernd E. Wolfinger[1] **Jürgen Wolf**[1] **Gwendal Le Grand**[2]

[1] **Dept. of Computer Science, TKRN, Hamburg University**
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
[2] **Ecole Nationale Supérieure des Télécommunications - INFRES, Paris, France**

E-mail: {`wolfinger`|`jwolf`}`@informatik.uni-hamburg.de, gwendal.legrand@enst.fr`

Keywords: Quality of Service, Resource Management, Traffic Management, Real-Time Communications, Perfomance Evaluation

Abstract

An important means to guarantee an acceptable quality of service in networks with real-time communication requirements is the reservation of resources at connection setup time. However, such reserved resources, e.g. transmission bandwidth, may be unused as a consequence of the variations in the actual resource demands. Therefore, a more efficient resource utilization is possible if communicating stations or end-users dynamically hand over some of the free resources temporarily to the other communication partners, e.g. of a "broadcast network".

This paper concentrates on two fundamental problems of such a demand-based sharing of resources: on the one hand, estimation of the current resource requirement on the basis of load measurements is investigated and, on the other hand, we elaborate efficient algorithms for resource sharing respecting real-time requirements. The algorithms proposed for load estimation and for resource sharing are evaluated analytically with respect to their efficiency for worst-case, average-case and realistic load scenarios. Our approach suggested for resource and traffic management allows one to achieve significantly better utilization of network resources.

1 INTRODUCTION

The trend to distributed multimedia communications via local-area networks has led to the requirement to transmit continuous media streams, such as audio and video streams, with sufficiently good quality. Basically, two common possibilities exist in order to guarantee a certain QoS as required by the users of a network: either, one can use *resource reservation* or, alternatively, one could rely on *priorization combined with admission control*. In the context of the Internet, the resource reservation approach is reflected e.g. by the IntServ proposal [1] and the priorization approach is inherent to the DiffServ proposal [2].

In this paper we focus on the resource reservation approach and its usage in local broadcast networks. In particular, we study ways of how communication resources which are already reserved and allocated to communicating end-users or end-systems can be dynamically handed over to other end-users or end-systems in case that they are not needed by their original "owner" for some time. Our interests center around methods which make resources given back sufficiently quickly, if they are needed again by their owner, in order to make sure that real-time requirements are not neglected.

The approach we suggest assumes that the problem of reserving and statically allocating resources to communicating entities has already been solved [3]. We also suppose that the resource reservation has been established for a rather long-term time interval, e.g. duration of an audio/video stream, typically in the order of minutes. In our approach, each owner of resources determines whether its communication load, i.e. the amount of time critical data waiting for transmission at the owner, justifies the continued reservation of all the resources as allocated to the owner. If a sufficiently large amount of resources is observed to be temporarily free, the owner of the resources will pass some of these resources to its "neighbors", i.e. the other end-systems of the broadcast LAN. In order to respect real-time requirements for its data to be sent each owner will continue to observe the arrivals to its transmission queue. If its local load is increasing again, the owner informs its neighbors that they are no longer allowed to make use of the resources which were offered for public access at an earlier instant. The freeing of resources and their reclaiming could lead to oscillations in the resource redistribution. Therefore, our approach for making resource allocation, depending on the actual communication load, is based on load estimators, which produce some kind of smoothed estimates for the load as it is generated locally over time. Moreover, we introduce systems of load thresholds which, only when being crossed by the load estimates of an owner of resources, leads to a broadcast of messages for freeing or reclaiming resources.

The remainder of the paper is organized as follows: Section 2 gives a survey of the state of the art in load estimation, scheduling resource allocations and dynamic redistribution of resource reservations in the context of QoS management. Section 3 presents and analyzes the behavior of a set of functions for estimating the actually generated load at a LAN station, based on geometric or on arithmetic weighting of measured utilization samples. In Section 4 we introduce several threshold systems which differ in their responsiveness to observed load changes and in their overhead, in terms of number of messages, for sending resource redistribution messages. Section 5 investigates, by means of mathematical models, how load estimators can be combined with threshold systems by a single station if it has to respect real-time requirements for sending data. The main lessons learned are summarized in Section 6.

2 RELATED WORK

Reservation of resources is an important means to achieve quality of service guarantees in computer and communication networks. In continuous media communications with real-time requirements, such as audio/video communications, it is typical to reserve the resources (as they are expected to be required later on) during connection setup of a stream, cf. IntServ [1]. Some real-time protocols and services, such as CMTP/CMTS [4], allow that resources may be reallocated by means of setting up a new stream dynamically with different, in particular less strict, resource reservations. Other approaches assume that variations of load within single stations are smoothed out to a great extent by means of multiplexing a larger number of streams and doing the reservation for the complete overlay of those streams [5]. Up to now, only few proposals exist which - as in our approach - advocate for a dynamic freeing and recalling of bandwidth between stations, cf. [6] and [7]; but these earlier proposals do not provide a redistribution of the unused prioritized bandwidth to the other nodes in the LAN. Thus, these solutions satisfy strict real-time bounds for transmission delays, but at the cost of a worst case reservation.

Estimation of load, as a fundamental problem of our approach, has been an important topic, e.g., in adaptive routing and in load-balancing since more than a decade [8, 9]. Besides the idea of basing load estimation directly on the utilizations observed [10] it has also been suggested to observe arrival times, thus estimating arrival rate, and service-time requirements, thus estimating mean service-times, directly. However, in scenarios relevant to our paper service-time requirements of data transfer requests typically would not vary to a sufficient extent in order to justify the additional expenditure by roughly duplicating the measurement efforts (estimation of two parameters instead of one). Still, as an alternative to observing utilization one could observe the current backlog and base resource redistribution on this estimate.

Regarding the *weighting functions* needed for calculating load estimates, geometric weighting [11] seems to be the dominating approach. This can be explained by the easy and straight-forward way to execute the calculations for geometric weighting and also by the fact it has been used successfully in the context of TCP [12]. In *load sampling* besides the periodic sampling which we suggest for simplicity, a more complex measurement procedure based on so-called random sampling has been suggested just recently [13]. Measurements, as required in our paper, refer to observations of primary load rather than of secondary load in the sense of [14].

In order to *reduce the fluctuations of load* at the source besides using methods of traffic shaping, e.g. based on rate-based flow control such as [15], one could also apply techniques for smoothing the traffic generated at the source [16]. Our use of *threshold-systems* to reduce the number of control messages to be exchanged has been inspired again by similar techniques which have been successfully applied in practice in adaptive routing protocols [17].

As an alternative to provisioning deterministic *QoS guarantees*, as we strive for in our approach, stochastic QoS guarantees are possible [3]. Then, evidently, no longer a reservation for the worst case would be required but QoS requirements could become violated temporarily. For applications tolerating transient fluctuations in the QoS, different approaches for resource management exist, in particular, in the context of wireless networks [18, 19].

3 ESTIMATION OF ACTUAL NODE UTILIZATIONS BASED ON WEIGHT FUNCTIONS

In the following we present the basic proceeding which we suppose throughout this paper for resource reservation and redistribution between the stations of a broadcast network.

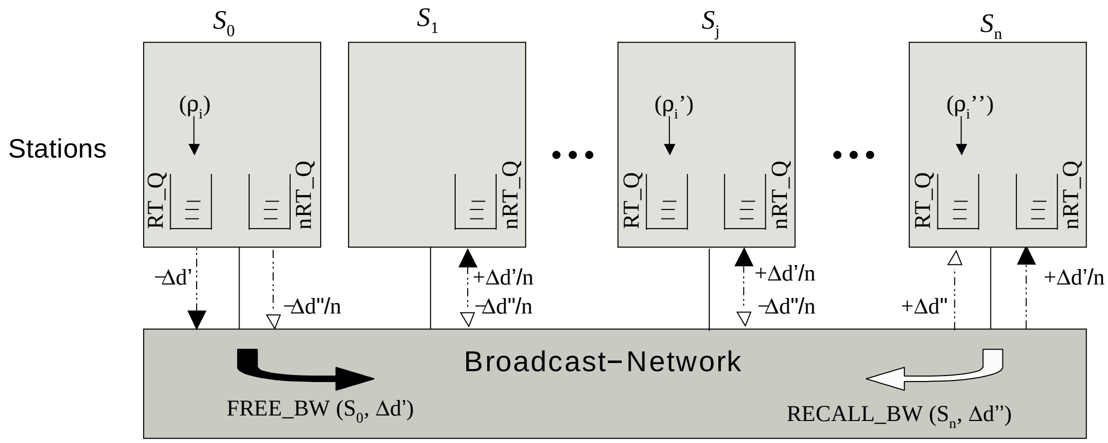

Figure 1. Redistribution of bandwidth: the depicted example scenario illustrates the freeing of $\Delta d'$ by station S_0 and recalling of $\Delta d''$ by S_n

3.1 Assumptions and underlying model for resource redistribution

- Resource reservation: We assume:

  – $n+1$ stations $S_0, \ldots, S_n$ communicating via local broadcast network (Ethernet, WLAN,...[20])

  – communication load of the stations consisting of time-critical transmissions and non-time-critical data transfers, cf. the real-time (RT_Q) and non-real-time queues (nRT_Q) in Figure 1

  – in each station with real-time communication requirements there exists at least one owner of resources, where reserved data rate (allocated "bandwidth") is the resource considered by way of example; scenarios could be that the owner reflects the complete station or an owner could model a single end-user

  – resources are supposed to be reserved before starting a new stream with real-time requirements and are available to its owner until it releases connection after end of the stream; the amount of bandwidth reserved is such that the owner's QoS requirements can be guaranteed.

- Temporary resource redistribution:

  We assume that an owner of resources can pass resources to other stations temporarily, whenever these resources are not required locally for some time. We introduce a message of type

  – FREE_BW $(S, \Delta d)$ with which a station S hands over a bandwidth (respectively data rate) of Δd bit/s to other stations, and

  – RECALL_BW $(S, \Delta d)$ to recall bandwidth Δd.

  With receipt of a RECALL_BW message the recalled bandwidth has to be released immediately. The question of how to distribute free bandwidth among stations is out of the scope of this paper; an elementary solution could be, that all neighboring stations get equal shares. A more advanced approach to distribute free bandwidth efficiently is proposed in [6].

- Approach to estimate the level of load resulting from time-critical transmission requests:

  We assume periodic measurements available at instants $t_i := t_0 + i \cdot \Delta t$, for $i \geq 1$, where the measured samples ρ_i characterize the load of time-critical transmission requests which have been generated by the source during the interval $T_i := [t_{i-1}, t_i)$. Evidently, $\rho_i := d_i/(r \cdot \Delta t)$, where d_i denotes the amount of time-critical data generated during T_i and r denotes the data rate as it was allocated to its owner at connection set-up time. This sequence of samples (ρ_i) may be used at each instant t_i to calculate an estimate $\hat{\rho}(t_i)$ of the actual level of load.

The traffic load generated by the communicating users of a station must be estimated in order to be able to adjust the allocated capacity to the actual requirement. If the capacity, in terms of transmission bandwidth, allocated is too high, only a poor level of utilization for the network can be achieved. Some flows may have more reserved capacity than what they really use. This capacity is lost for all the nodes of the network. On the other hand, if the reserved capacity is too low in comparison to the flow activity, a situation of congestion occurs and the general QoS of the flow will be degraded. It is important to find out good estimators to allow the algorithm controlling the bandwidth to react in an appropriate manner.

The estimator must take into account some kinds of fluctuations in order to enable adequate node reactions. Major changes in data rates for a complex source of traffic must be detected quickly. This is an important condition for flows with a required level of QoS. If the reaction is too slow, a QoS degradation is viewed by these flows, whereas a minor change in data rates within single streams may be ignored in order to get a certain form of smoothness over time. This constraint is to avoid an overreaction with respect to the observed fluctuation of offered load within a station.

3.2 Geometric and arithmetic weighting of utilization estimates

The methods proposed and investigated by us for load estimation are *geometric* and *arithmetic* weighting.

Here, ***geometric weighting*** is defined as follows:

$$\hat{\rho}(t_i) = \alpha\rho_i + (1-\alpha)\cdot\hat{\rho}(t_{i-1}), \quad i \geq 1$$

where $\alpha \in (0,1]$ and $\hat{\rho}(t_0)$ to be initialized. The α factor indicates how fast the estimator changes with a strongly different new sample. The geometric weighting is also known as the *exponential weighted moving average* (EWMA) [11]. As an abbreviation we denote by G_α geometric weighting with parameter $\alpha \in (0,1]$.

The ***arithmetic weighting*** estimator is defined as:

$$\hat{\rho}(t_i) = C_0 \cdot \sum_{j=0}^{w-1} \frac{w-j}{w}\rho_{i-j}, \quad i \geq 1,$$

where $w \in \{1,2,\ldots\}$ and ρ_k to be initialized for all $2-w \leq k \leq 0$. $C_0 = \frac{2}{w+1}$ denotes a normalization constant. This estimator is equivalent to a sliding window with window size w and weights increasing in a linear way when the samples are more recent. Arithmetic weighting uses $\frac{k}{w}$ as weights, where k is the position in the window of size w with the most recent sample taking the position $k = w$. The size of w indicates how many past samples including the present sample must be kept to estimate the current load. As an abbreviation, A_w denotes arithmetic weighting with a window size of $w \in \mathbb{N}$.

3.3 Impact of weight functions on worst case backlog of load and on delays

In this section we discuss the impact of the weight functions and their parametrization on estimation of $\hat{\rho}(t)$. In particular, we analyze how much backlog (BL) is possible if the "worst case" occurs and determine how much "delay" is induced at most by the BL. It has been demonstrated previously [21] that all measured data (offered load) is taken into account once and only once by both of our estimators.

We constitute that the throughput available for each interval T_i (with boundaries $[t_{i-1}, t_i)$) is identical to $\hat{\rho}(t_{i-1})\cdot r\cdot\Delta t$, except that lowering throughput from one interval to the next is not admitted if backlog exists in the sending queue. Here we disregard the time required to claim back transmission capacity. As mentioned previously, within the scope of these studies we presume that the amount of data d_i generated in each sample interval does not exceed $r\cdot\Delta t$. Hence ρ_i does not exceed 1. Such behavior could be realized, e.g., by a "leaky bucket" component, which could establish some rate-based flow control [15].

The normalized "Backlog of workload" (BL) is defined in the following as the quotient of the amount of data which has to be stored temporarily due to missing transmission capacity and the amount of data that can be sent with full transmission capacity during one sample interval of length Δt. This means that $BL = 1$ corresponds to the amount of data $r\cdot\Delta t$.

(a) **Analysis of "worst case" load scenario**

A "worst case" scenario occurs, e.g., when

$$\rho_i = \begin{cases} 0 & \text{for } i = 0 \\ 1 & \text{for } i \geq 1 \end{cases} \tag{1}$$

- Using **geometric weighting**, the estimator at instant t_i, initialized with $\hat{\rho}(t_0) := 0$, for $i \geq 0$ is given by

$$\hat{\rho}(t_i) \;=\; \sum_{k=0}^{i} \alpha(1-\alpha)^k \rho_{i-k} \stackrel{(1)}{=} 1-(1-\alpha)^i.$$

Hence, as normalized backlog $BL(i)$ which accumulates in just one sample interval, we obtain for each interval T_i with $i \geq 1$ the difference between actual level of load ρ_i and the estimated level of load $\hat{\rho}(t_{i-1})$ at the beginning of the interval:

$$BL(i) = \rho_i - \hat{\rho}(t_{i-1}) = (1-\alpha)^{i-1}$$

which gives us the following expression for the overall backlog BL_i having accumulated up to time instant t_i:

$$BL_i = \sum_{j=1}^{i} BL(j) = \frac{1-(1-\alpha)^i}{\alpha} \qquad \forall i \geq 1 \tag{2}$$

The total size of generated (offered) load to be buffered, expressed in multiples of $r\cdot\Delta t$, is therefore for all $\alpha \in (0,1]$ limited by

$$BL_i \leq BL_\infty = \lim_{k\to\infty} \frac{1-(1-\alpha)^k}{\alpha} = \frac{1}{\alpha} \quad \forall i \geq 1$$

Examples:

$\alpha = 1$: Since, at each instant t_i, ρ_i is the only value

used for estimation ($\hat{\rho}(t_i) = \rho_i$), as a maximum the amount of data received during the last interval T_i has to be queued in RT_Q for future transmission. Therefore the amount of data to be buffered can not exceed $1 \cdot r \cdot \Delta t$, i.e. $BL_{max} = 1$.
$\alpha = 0.5$: The maximum amount of data accumulating in the worst case is $BL_{max} = 2$.

- Using **arithmetic weighting**, and taking into account Equation (1), the estimator at instant t_i is given by

$$\hat{\rho}(t_i) = \begin{cases} \frac{2}{w+1} \sum_{j=0}^{i-1} \frac{w-j}{w} = \frac{2wi+i-i^2}{(w+1)w} & 0 \le i \le w \\ 1 & i > w. \end{cases} \quad (3)$$

Again we obtain the maximum backlog for each interval T_i as difference between actual and estimated load

$$BL(i) = \begin{cases} 1 - \frac{(i-1)\cdot(2w-i+2)}{(w+1)w} & \text{for } 1 \le i \le w \\ 0 & \text{for } i > w \end{cases}$$

which leads to the maximum backlog at instant t_i with $1 \le i \le w$

$$BL_i = \sum_{j=1}^{i} (1 - \tfrac{(j-1)\cdot(2w-j+2)}{(w+1)w}) \quad (4)$$

and $BL_i = BL_w$ for $i > w$. The total size of generated (offered) load to be buffered is limited by BL_w, i.e.

$$\begin{aligned} BL_i &\le BL_w = \sum_{j=1}^{w} (1 - \frac{(j-1)\cdot(2w-j+2)}{(w+1)w}) \\ &= \frac{w+2}{3} \qquad \forall w \ge 1,\ i \ge 1 \end{aligned}$$

Examples:
$w = 1$: With $w = 1$ the maximum backlog is $BL_{max} = 1$ as the amount of data received within T_i has to be queued in the worst case, but is completely served during T_{i+1}.
$w = 4$: Maximum backlog results to $BL_{max} = 2$.

(b) **Analysis of maximum backlog delay**
Now we determine how much backlog delay at most is induced by the data being stored temporarily. The backlog delay is the time elapsing between receiving data and being ready to send it and its maximum for each interval T_i is denoted with τ_i. The data is processed with throughput $\hat{\rho}(t_{i-1}) \cdot r \cdot \Delta t$ within each interval T_i following the FIFO-principle. As mentioned previously, we constitute in our study that the available throughput is not lowered while further backlog is present. Nevertheless, the generated load might decrease at any instant whereby the remaining backlogged data would be sent with a data rate less than the maximum, possibly resulting in higher delay. Hence, considering the highest increase of generated load, which is described by Equation (1), it follows that the maximum delay is upper bounded by

$$\tau_{max} = \max_{i \ge 1} \tau_i = \max_{i \ge 1} \frac{BL_i \cdot r \cdot \Delta t}{\hat{\rho}(t_i) \cdot r}.$$

- Using **geometric weighting**, the maximum backlog delay is given by

$$\begin{aligned} \tau_{max} &= \max_{i \ge 1} \tfrac{BL_i \cdot r \cdot \Delta t}{\hat{\rho}(t_i) \cdot r} \\ &\overset{(2)}{=} \max_{i \ge 1} \tfrac{\frac{\hat{\rho}(t_i)}{\alpha} \cdot r \cdot \Delta t}{\hat{\rho}(t_i) \cdot r} = \tfrac{1}{\alpha} \cdot \Delta t. \end{aligned} \quad (5)$$

- Using **arithmetic weighting**, the maximum backlog delay is given by

$$\begin{aligned} \tau_{max} &= \max_{i \ge 1} \frac{BL_i \cdot r \cdot \Delta t}{\hat{\rho}(t_i) \cdot r} \\ &\overset{(4)(3)}{=} \max_{1 \le i \le w} \frac{\frac{1}{3} i^2 - (w+1)i + w^2 + 2w + \frac{2}{3}}{-i + 2w + 1} \cdot \Delta t. \end{aligned}$$

Since the above fraction is monotonically decreasing (with $1 \le i \le w, w \ge 1$) it follows

$$\tau_{max} = \frac{BL_1}{\hat{\rho}(t_1)} \cdot \Delta t = \frac{w+1}{2} \cdot \Delta t. \quad (6)$$

From the equations above an upper bound for τ_{max} can be computed, which allows us to determine the acceptable range of values which can be chosen for α or for w, if a given delay bound τ_{QoS} and a sampling interval size Δt have to be respected. Table 1 shows the maximum backlog delay for different estimators. As we will show in Section 5.2 upper bounds for τ_{max} for estimators combined with threshold systems can be computed using a recursive approach leading to significantly better bounds than the closed-form expression as derived in [21].

Table 1. Upper bounds for τ_{max} given as multiples of Δt.

Estimator	$G_{0.1}$	$G_{0.3}$	A_{10}	A_{20}
max. Delay	10	$3.\overline{3}$	5.5	10.5

3.4 Responsiveness of different load estimators

Besides requiring appropriate throughput, applications with real-time requirements need guarantees for upper limits of delay. In order to keep these limits, the

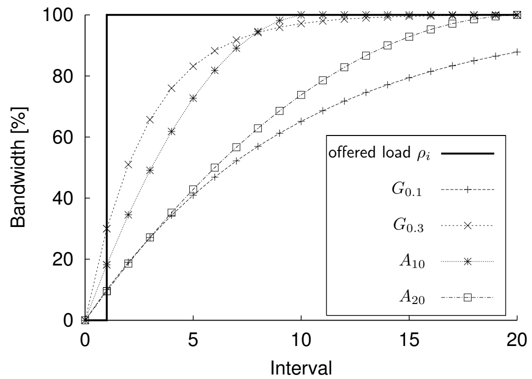

Figure 2. Estimators in the worst case

parameters, in particular Δt and α or w respectively, have to be chosen carefully.

The Δt value is bounded below, e.g., by the component used responsible for smoothing incoming data, guaranteeing $\rho_i \leq 1$. As can be seen from Equations (5) and (6) an upper bound for Δt is defined depending on the delay limit from the application and on the deployed weighting estimator and its parameter α or w respectively.

Figure 2 compares the reactivity of four estimators, two estimators for either arithmetic and geometric weighting, again assuming the worst case load scenario being the one showing the highest possible fluctuation after the estimator has taken its lowest value (cf. Equation (1)).

We observe that functions A_{10} and $G_{0.3}$ are more reactive than functions A_{20} and $G_{0.1}$. This results from the fact that parametrization with α values near to 1 using geometric weighting or small w values using arithmetic weighting leads to the advantage, that an increase of load is taken into account rather quickly and thus the past ("history") is "faded out" fast as well. But with this also quite high variations in function $\hat{\rho}(t)$ come along.

We also observe that $G_{0.3}$ is more reactive than A_{10} during the first samples (it yields the highest estimates during 8 samples) but $G_{0.3}$ takes much longer than A_{10} to estimate the traffic as being approximately 100% (4 more samples). This means that $G_{0.3}$ reacts faster to traffic fluctuations, but it needs more time to converge.

4 SYSTEMS OF LOAD THRESHOLDS TO CONTROL DISTRIBUTION OF BANDWIDTH

In this section we introduce systems of load thresholds, which determine when to free or claim back bandwidth based on current load estimation as discussed in the preceding section. The main tasks of such systems are on the one hand to assure real-time requirements to be respected and secondly to avert oscillations caused by small variations of the estimator's value. We present and discuss three example systems which illustrate as well their representation by state models as their analysis concerning potential increase of network utilization.

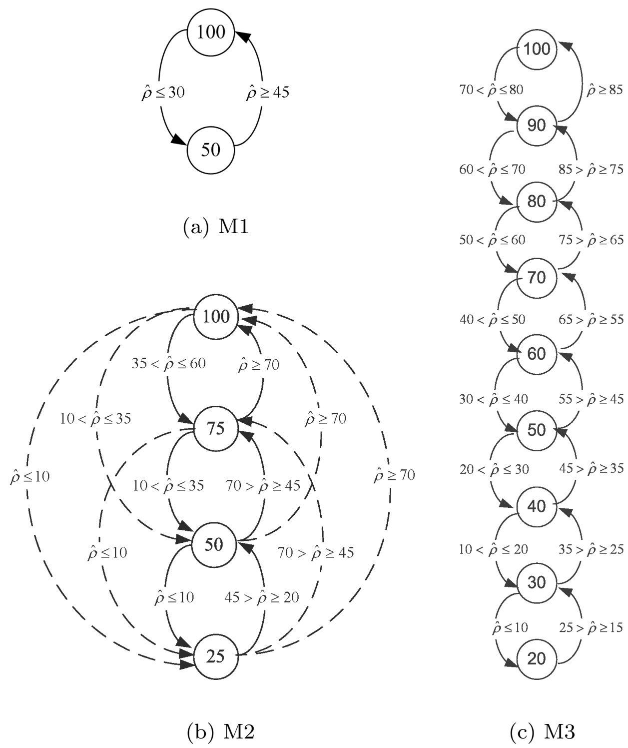

Figure 3. Proposed state models with 2 (a), 4 (b) and 9 (c) states. For simplicity, M3 is not represented with all transitions.

Figures 3(a) - 3(c) represent the three state transition models we consider in this work. These models exemplify different parametrizations for the decisions how much bandwidth to be allocated to a specific node depending on its momentary traffic estimation. The states are labeled with numbers in the range of 0 to 100 representing the percentage of the bandwidth reserved at setup time which should actually be allocated to the node. State transitions represent the crossings of thresholds constituted by the corresponding model. State model M3 has been represented in a simplified way to clarify the figure. The fast transitions, representing thresholds which are not the closest to the current state, are not represented by the figure, but they exist in the analytical model.

By taking care of recalling bandwidth before the level of utilization is reached, real-time requirements can be respected. The three sample models were assorted considering the following principles: In this study we analyze equidistant upper and lower thresholds. Since perception of load variations is delayed as a consequence of the discrete sample interval size, the actual available bandwidth should always exceed the estimate, e.g. by at least 5%. Furthermore, to effectively avoid oscillations it is essential that upper and lower threshold have sufficient distance. We use a minimum difference of 10%. As mentioned previously, we presume that any host will distribute transmission capacity only in the case that no local backlog exists.

The fundamental differences between the state models are their granularities. State model M1 consists of two states, state model M2 consists of four states, and model M3 is composed of nine states. Intuitively, more states will be more efficient in terms of local loss, but globally, more states will cause more signalling and this will not necessarily be more efficient from a network's point of view. Evidently, different stations can use different threshold systems as well as different load estimators depending on their application. Both granularity and levels of thresholds can be customized to individual needs.

4.1 Potential increase of overall network utilization: worst case study

We consider resource reservation and static allocation to communicating entities. Hence, if just a small part of the reserved data rate is used, the remainder is lost for the overall network – in the worst case this sacrifice can reach 100% of the initially reserved bandwidth, obviously, if no real-time data is transmitted at all. This study is focused on methods to reduce such sacrifice even if quality of service guarantees are provided. We investigate in the following how much capacity is made available using the presented threshold systems.

Let us consider how much network capacity can, in the worst case, be reserved but remain unused by a station though a system for sharing bandwidth using our proposal was employed. This sacrificed capacity, in multiples of data rate r, can not exceed the maximum difference between $\hat{\rho}$ and its associated state. As mentioned in Section 3.3 the entire load measured in each sample interval is considered adequately in the long run by the presented estimators. Therefore, it is acceptable to consider the estimated load $\hat{\rho}$ instead of the actual load in order to assess the utilized and the provided capacity respectively.

Using the proposed models M1, M2, or M3, respectively, at least 30%, 60%, or 80% of the a-priori reserved capacity is utilized or placed at other stations' disposal. These values regarding the worst case result from the difference between reserved data rate and maximum sacrificed capacity.

So, the minimum utilized or freed capacity can be enhanced by increasing the number of states of the model. Evidently, communication overhead used for signalling increases as a consequence. We neglect this overhead, depending on individual implementations, for all of the analysis within Section 4. As results from Section 5 will show, this simplification can be acceptable in real load scenarios.

Table 2. Station's state and sacrified capacity for different ranges of $\hat{\rho}(t_i)$.

$\hat{\rho}(t_i)$ [%]	state	sacrified capacity [%]
0-10	25	20
10-20	$25 \vee 50$	10-35
20-35	50	22.5
35-45	$50 \vee 75$	10-35
45-60	75	22.5
60-70	$75 \vee 100$	10-35
70-100	100	15

It is worth mentioning that already a very simple 4-state model such as Model M2 is, in any case, able to make sure that at least 60% of the available real-time bandwidth is utilized by the communicating stations – assuming that the offered load, e.g. in terms of best effort traffic, is sufficiently high.

4.2 Potential increase of overall network utilization: average case study

With the worst case assessment above, we derived hard boundaries. In the following we estimate the potential increase of overall network utilization, in form of upper bounds of sacrifice, for an average case. We do not assume further restrictions on the sending behavior of the communicating stations. Therefore, for this evaluation we presume $\hat{\rho}(t_i)$ as being uniformly distributed in the interval [0,1]. We demonstrate the assessment using the 4-state model M2.

To analyze the overall sacrifice, we determine sacrified capacity for ranges of possible values for $\hat{\rho}(t_i)$. These are determined as follows: In case of $0\% \leq \hat{\rho}(t_i) \leq 10\%$, illustrated in the first data line of Table 2, the station is in state 25 what corresponds to reserved capacity of 25%. Thus, the mean sacrified capacity is 20% for this partition. In the next line the range $10\% < \hat{\rho}(t_i) \leq 20\%$ is shown. In this case the station's state can not be determined uniquely, it might be state 25 or 50. Hence the unused, but reserved, capacity averages to a value between 10% and 35% depending on the size of the station's states' shares.

From Table 2 we derive two values namely σ_o and σ_p, respectively, which correspond to the optimistic (or pessimistic) assumptions that the station's state is the favorable (or unfavorable) one all along. The assumption that $\hat{\rho}(t_i)$ is uniformly distributed leads to the assessments $\sigma_o = 16.25\%$ and $\sigma_p = 23.75\%$.

To sum up, Table 3 shows the assessments under optimistic and pessimistic assumptions as well as the esti-

Table 3. Average case: Estimated sacrifice σ_o and σ_p as well as estimated overall utilization of bandwidth

Model	σ_o	σ_p	used/lent out
M1	27.5%	35%	65-72.5%
M2	16.25%	23.75%	76.25-83.75%
M3	10%	14%	86-90%

mated utilization of the reserved data rate for the proposed threshold systems. The estimated values let us expect significant enhancements for real-time communications within broadcast networks. In Section 5 we will check those potential enhancements under realistic load scenarios – and show that the derived assessments meet well reality.

5 NODE BEHAVIOR BASED ON LOAD ESTIMATOR AND THRESHOLD ALLOCATION SYSTEM

5.1 Investigation to study responsiveness of capacity allocation

The main goal of this section is to evaluate how well applications' real-time constraints can be guaranteed by using our proposed threshold system in combination with different load estimation functions. In particular, we evaluate the time until transmission capacity is called back from other stations when it is needed again by its owner. We verify that resources are called back sufficiently quickly in order to ensure that real-time requirements are met – that is as a rule within 100ms, but depends on the relevant application.

First, we investigate the worst case, that is a situation when the offered load increases to its maximum after a node has redistributed all of its capacity to the other nodes in the network, cf. Equation (1). In Section 3.4 we discussed the responsiveness of different estimators and now we evaluate how fast the up-thresholds are crossed. Our concern is to get back full capacity as soon as possible. Therefore, the reactivity of each model corresponds to the number of intervals needed to get back full capacity.

Using $G_{0.3}$, full capacity is reached within only 2 sampling intervals using state model M1. That is faster than in the case of state model M2 which needs 4 sampling intervals and faster than in the case of state model M3 (6 sampling intervals). This is due to the fact that an estimator $\hat{\rho}$ greater than 45% in model M1, 70% in model M2, and greater than 85% in model M3 is needed to get back full capacity. In other words, a high value of the estimator is needed to reach an allocated value of 100% in the case of model M3. Moreover, $G_{0.3}$'s slope decreases significantly when 80% of the capacity is reached; therefore, in this case model M2 is more reactive than model M3.

From a networking point of view, the gain obtained using model M1 is even greater since each threshold crossing is accompanied by signalling information and thus more signalling is needed if model M2 or model M3 is used.

5.2 Maximum delay induced due to threshold systems

In Section 3 we analytically derived upper bounds for the maximum backlog delay which resulted from the estimators' underestimations of the actual load and from the time lag between load appearance and its measurement. In the following we discuss the impact of the threshold systems combined with weight functions as introduced previously. In particular, we upper-bound the maximum delay τ_i that can occur for data received during interval T_i.

For this we determine that instant $t' \in T_i$ at which data is delayed longestly. The delay is computed then as difference between the instant t'', that is the instant when all data waiting in the sending queue at instant t' has just been sent, and t'.

The backlog delay within T_i is maximal for the instant $t' := t_{i-1} + \frac{\rho_i}{r}$. Assuming the worst case that data is generated with the maximum possible data rate r during the first part of the interval, t' is the instant when the last data unit within T_i has been added to the sending buffer. In order to determine t'', with S_i denoting the portion of r allocated during T_i by the threshold system, let B_i denote the amount of backlogged data buffered after interval T_i:

$$B_i = \max(0, B_{i-1} + (\rho_i - S_i) \cdot r \cdot \Delta t) \qquad \text{for } i > 0$$

Algorithm 1 $\textsc{Delay}(x, i)$

Require: Amount of data x and index i of interval; S_i denotes portion of r allocated within T_i.
Ensure: Time needed to send amount of data x starting at interval T_i;

```
x_r ← x − S_i · r · Δt          /* data not sent during T_i */
if x_r > 0 then
    return 1 + DELAY(x_r, i + 1)         /* start sending
                                            remainder in T_{i+1} */
else
    return x / (S_i · r)                  /* time to send x in T_i */
end if
```

Furthermore, we introduce $\textsc{Delay}(x, i)$, described by Algorithm 1, computing the maximum time needed to send the amount of data x where sending starts at the beginning of interval T_i. With

$$t'' = t_{i-1} + \textsc{Delay}(B_{i-1} + \rho_i \cdot r \cdot \Delta t, i)$$

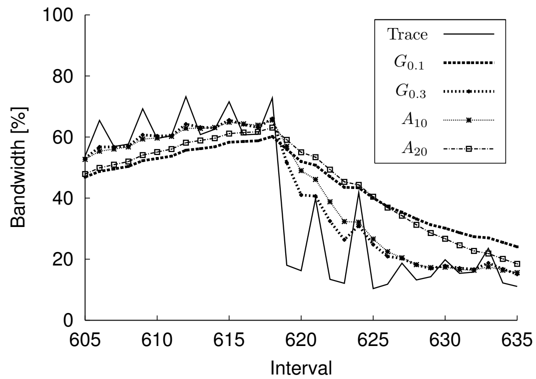

Figure 4. Jurassic Park/Mr Bean trace – different estimators

it follows

$$\tau_i \le t'' - t' = \textsc{Delay}(B_{i-1} + \rho_i \cdot r \cdot \Delta t, i) - \frac{\rho_i}{r}.$$

Applying the load scenario given by Equation (1), the maximum delays shown by Table 4 can be computed. It can be seen that even hard real-time requirements can be solved by selecting adequate estimators and models. To give an example, by using a geometric estimator with $\alpha = 0.3$ combined with Model M1, the maximum backlog delay does not exceed Δt.

Table 4. Maximum backlog delay in multiples of Δt in the load scenario given by Equation (1) for different threshold systems.

	$G_{0.1}$	$G_{0.3}$	A_{10}	A_{20}
Model M1	3	1	1.5	3
Model M2	5.25	1.75	2.5	4.75
Model M3	6.7	2.2	3.1	5.5

5.3 Dynamic behavior of utilization for realistic load situations within a station

We evaluate now our scheme in a realistic situation, using real MPEG4 traces. The input MPEG4 trace chosen is a mix of the two MPEG4 traces obtained from [22]. The traces are that of the movies "Jurassic Park" and "Mr. Bean" which we superimpose. We study the reactivity of each transition model using arithmetic and geometric estimators with different weights with this typical trace of total size 573.8 MB, lasting 59.37 minutes. The size of the sample intervals is $\Delta t = 40\ ms$ with maximum measured data within one interval of $d_{max} = 178.41\ kbit$ which induces an a-priori reserved capacity of $r = 4.4\ Mbit/s$ in order to fulfill the precondition $\rho_i \le 1\ \forall i$.

Figure 4 shows a clipping from the trace and its corresponding estimator values. We have selected sampling intervals during which many fluctuations can be observed in the input trace in order to see how reactive the models are. As mentioned previously, in all cases, $G_{0.3}$ is the most reactive of the traffic estimators compared. The bandwidth allocated by each of the models M1-M3 with estimators $G_{0.3}$ and A_{10} are represented in Figure 5. This figure describes how the different state models realize the bandwidth allocation in the case of real traffic fluctuations.

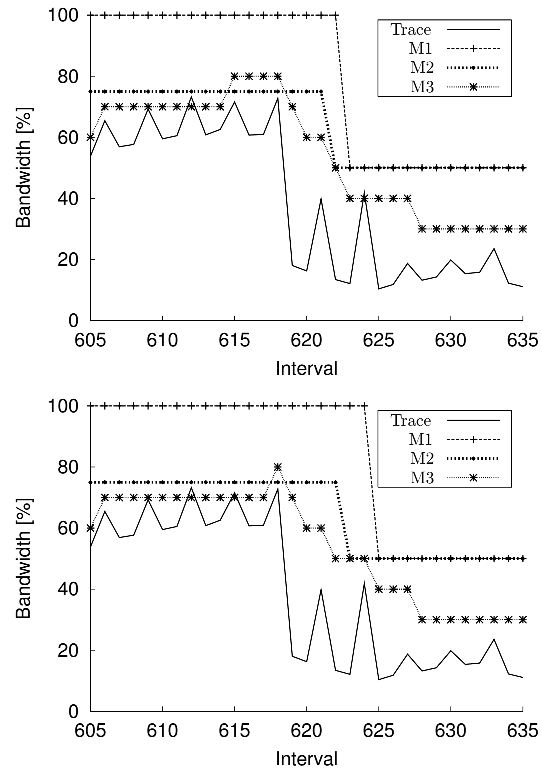

Figure 5. Jurrassic Parc/Mr Bean trace – $G_{0.3}$ (top picture) and A_{10} combined with models M1-M3

Table 5 shows the network capacity put at other stations' disposal, the used buffer, the maximum and mean time the transmission of data was delayed for as well as the number of FREE/RECALL messages exchanged by the system. Mentionable, we neglect the signalling communication overhead needed for controling the sharing of bandwidth; the actual gained capacity depends on the details of individual implementations.

Previously, we have chosen the data rate r so that it is never exceeded by the stream. Since we can not always act on this presumption of excessive over-reservation, we start further experiments with the trace. The data rate is set to $r = 125\ kbit/40\ ms$, whereby the video mix uses 42.2% of the bandwidth (instead of 29.6%) which appears more pragmatical. Table 6 shows the results of the run without prior smoothing, tolerating $\rho > 1$.

Contrary to the worst case, in all experiments transition model M3 achieves the minimal capacity loss. This

Table 5. Gained bandwidth (G), used buffer (B), additional delay (τ), and number of FREE/RECALL messages for video mix sent with $r = 4.4\ Mbit/s$ using different combinations of estimators and threshold systems ($\cdot^* \widehat{=}$ max., $\bar{\cdot} \widehat{=}$ mean)

Video mix Jurassic Park / Mr Bean. Data rate: $r = 178.41\ kbit/40ms (\widehat{=} 4.4\ Mbit/s)$ Mean data rate: 1.29 $Mbit/s$ (29.58%)	M1				M2				M3			
	$G_{0.1}$	$G_{0.3}$	A_{10}	A_{20}	$G_{0.1}$	$G_{0.3}$	A_{10}	A_{20}	$G_{0.1}$	$G_{0.3}$	A_{10}	A_{20}
G [%]	41.6	38.9	40.8	41.3	47.1	46.1	47.0	47.0	57.8	56.7	57.5	57.9
B^* [KB]	28.1	11.7	17.0	28.1	28.1	11.7	17.0	28.1	42.0	19.2	24.1	39.7
$\overline{B}$ [Byte]	166.0	133.8	151.1	160.9	206.0	175.2	192.3	201.4	538.3	423.8	474.0	526.2
τ^* [ms]	85.8	40.8	47.0	85.8	85.8	52.5	62.1	85.8	133.4	59.9	77.1	126.3
$\overline{\tau}$ [ms]	9.5	8.5	9.2	9.4	11.1	10.5	11.0	11.1	18.2	16.7	17.6	18.2
Messages	114	394	208	120	159	917	345	187	1008	8817	3330	1178

Table 6. Video mix, not smoothed, sent with $r = 3.1\ Mbit/s$ - see Table 5 for legend.

Video mix Jurassic Park / Mr Bean (not smoothed, no additional delay). Data rate: $r = 125.00\ kbit/40ms (\widehat{=} 3.1\ Mbit/s)$ Mean data rate: 1.29 $Mbit/s$ (42.22%)	M1				M2				M3			
	$G_{0.1}$	$G_{0.3}$	A_{10}	A_{20}	$G_{0.1}$	$G_{0.3}$	A_{10}	A_{20}	$G_{0.1}$	$G_{0.3}$	A_{10}	A_{20}
G [%]	18.4	16.0	19.6	19.0	35.7	33.7	35.7	35.8	45.1	44.0	44.6	45.2
B^* [KB]	24.4	24.4	24.4	24.4	43.5	24.4	24.4	37.2	51.4	25.9	31.1	45.9
$\overline{B}$ [Byte]	213.7	162.0	205.2	213.2	452.1	351.1	410.5	441.8	883.2	627.3	722.6	858.5
τ^* [ms]	92.8	62.4	62.4	79.6	121.7	62.4	67.0	105.9	149.7	66.4	85.7	132.3
$\overline{\tau}$ [ms]	5.6	4.4	5.8	5.7	11.2	9.7	11.0	11.2	17.5	15.4	16.4	17.6
Messages	213	1141	457	237	456	3842	1302	516	2106	14781	6975	2291

is due to the fact that real traces often produce variations which are less radical than the worst case, whereas more detailed models can better adapt the resource allocation to the actual requirements. However, the allocated bandwidth of M3 is often smaller than the value required by the input trace. M2 fits this requirement better since it is very rarely smaller than the bandwidth required by the trace. Therefore, a more detailed model achieves better performance in terms of lost bandwidth, but also introduces more delay if the trace fluctuations are rapid and also more communication overhead is generated.

It can be seen, that significant enhancements under realistic load scenarios are achieved by the threshold systems. It is also worth mentioning that the derived assessments from Section 4.2 meet well the realistic traces we tested, cf. Table 3. Further experiments, in particular with smoothed traces, show similar results. Though offered load sometimes exceeds the presumed upper bound of $\rho_i \leq 1$, neither mean nor maximum delay differ significantly, although the number of exchanged messages rises.

An overall interpretation of the results obtained in our various series of experiments leads to the following insights:

- As to be expected, the gained bandwidth G increases starting with model M1 via M2 towards M3 and this holds independently of the weighting function used $(G_{0.1}, G_{0.3}, A_{10}, A_{20})$. Evidently, the advantages of more refined threshold models become much more significant if one reduces the amount of resources (bandwidth) as reserved during connection-setup-time.

- Some direct dependency between the results regarding buffer occupancy B and additional backlog delay τ exists as a direct consequence of *Little's Law* [23].

- The mean backlog delay $\overline{\tau}$, in all cases, remains below 20 ms, which is encouraging and seems to be fully acceptable for a large variety of multimedia applications with real-time requirements. The maximum backlog delay τ_{max}, too, remains pleasingly small, at least for cases where the weight function $G_{0.3}$ or A_{10} is used.

- The number of control messages to be exchanged remain practicably even in the worst case as the control messages have a rather small size. So we conclude that the overhead of control messages to be broadcasted seems to be perfectly acceptable, too. And this holds even for network configurations where a relatively large number of stations (e.g. 50 to 100) with real-time communication requirements is present in each one of possibly many interconnected broadcast networks.

6 CONCLUSIONS

In this paper, we have investigated how the efficiency of reservation strategies that execute some fixed resource reservations (in particular reservations for the complete duration of continuous media streams) can be increased. On the one hand, we have proposed two classes of parametrized weight functions for calculating estimates of the actual load as induced by the connections for which resource reservations have been established for the present. On the other hand, we have introduced threshold systems in order to reduce the update traffic for exchanging control information the purpose of which is to hand over free bandwidth and to recall own resources between the stations of a broadcast network. We have demonstrated how weight functions and threshold systems can be combined.

Usage of mathematical performance predictions allowed us to prove that the algorithms we suggest are able to respect real-time bounds for maximum delays in the order of 50 to 100 ms (as they are quite typical in a large variety of audio/video applications) and that, at the same time, they achieve a high overall utilization of resources even under worst case load fluctuations. Additional studies have provided the encouraging result that in practice, i.e. for realistic load scenarios, the efficiency to be expected for our proposal to execute dynamic and load-adaptive resource reservations will even be considerably higher than the "worst case bounds".

Future investigations are planned by us to get a still better understanding regarding the dependencies between the parametrization as used for specifying the load estimators and the granularity of the threshold system being chosen. As a final goal we strive to solve the optimum configuring of adaptive reservation systems, for given load scenarios and given QoS requirements, in a largely automatic way and at the same time limiting the overhead resulting from the exchange of control information supporting the dynamic redistribution of resources.

REFERENCES

[1] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet Architecture: An Overview. RFC 1633, IETF, June 1994.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services. RFC 2475, IETF, Dec. 1998.

[3] H. J. Chao and X. Guo. *Quality of Service Control in High-Speed Networks*. J. Wiley, 2002.

[4] M. Moran and B. Wolfinger. Design of a Continuous Media Transport Service and Protocol. Technical Report TR-92-019, International Computer Science Institute, Berkeley, CA, April 1992.

[5] L. Massoulié and J. Roberts. Bandwidth Sharing: Objectives and Algorithms. *IEEE/ACM Transactions on Networking*, 10(3):320–328, 2002.

[6] P. Anelli and G. Le Grand. Differentiated Services over Shared Media. In *IWQoS*, pages 288–293, Karlsruhe, 2001.

[7] G. Le Grand. *Qualité de service dans des environments Internet mobile*. PhD thesis, Université P. et M. Curie, LIP 6, Paris, July 2001.

[8] C. R. Kalmanek, H. Kanakia, and S. Keshav. Rate Controlled Servers for Very High-Speed Networks. In *Proc. of the Conference on Global Communications (GLOBECOM)*, pages 12–20, 1990.

[9] A. Shalkh, J. Rexford, and K. G. Shin. Load-Sensitive Routing of Long-Lived IP Flows. In *Proc. of ACM SIGCOMM '99*, pages 215–226, Cambridge, USA, 1999.

[10] D. D. Clark and W. Fang. Explicit Allocation of Best-Effort Packet Delivery Service. *IEEE/ACM Transactions on Networking*, 6(4):362–373, August 1998.

[11] E. P. Rathgeb. Modelling and Performance Comparison of Policing Mechanisms for ATM Networks. *IEEE Journal On Selected Areas In Comm.*, 9(3):325–334, 1991.

[12] W. R. Stevens. *TCP/IP Illustrated: The Protocols*, volume 1. Addison-Wesley, 1994.

[13] B.-Y. Choi, J. Park, and Z.-L. Zhang. Adaptive Random Sampling for Load Change Detection. *Performance Evaluation Review*, 30(1):272–273, 2002.

[14] B. E. Wolfinger, M. Zaddach, K. D. Heidtmann, and G. Bai. Analytical Modeling of Primary and Secondory Load as Induced by Video Applications using UDP/IP. *Computer Commun.*, 25(11/12):1094–1102, 2002.

[15] S. Radhakrishnan, S. V. Raghavan, and A. K. Agrawala. A Flexible Traffic Shaper for High Speed Networks: Design and Comparative Study with Leaky Bucket. *Computer Networks and ISDN Systems*, 28(4):453–469, 1996.

[16] G. Bai and B. Wolfinger. Possibilities and Limitations in Smoothing MPEG-coded Video Streams: A Measurement-based Investigation. In *ITG/GI-Fachtagung MMB'97*. VDE-Verlag, 1997.

[17] M. Schwartz and T. Stern. Routing Techniques Used in Computer Communication Networks. *IEEE Transactions on Communications*, 28(4):539–552, April 1980.

[18] M. El-Kadi, S. Olariu, and H. Abdel-Wahab. A Rate-Based Borrowing Scheme for QoS Provisioning in Multimedia Wireless Networks. *IEEE Trans. on Parallel and Distributed Systems*, 13(2):156–166, February 2002.

[19] A. Malla, M. El-Kadi, S. Olariu, and P. Todorova. A Fair Resource Allocation Protocol for Multimedia Wireless Networks. *IEEE Trans. on Parallel and Distributed Systems*, 14(1):63–71, January 2003.

[20] A. S. Tanenbaum. *Computer Networks*. Prentice-Hall, 4th edition, 2003.

[21] J. Wolf, B. Wolfinger, G. Le Grand, and P. Anelli. Leistungsbewertung von Algorithmen zur dynamischen Ressourcenverwaltung in lokalen "Broadcast"-Netzen. In *Proc. of the GI/ITG-Conf. "Kommunikation in Verteilten Systemen" (KiVS)*. Springer Verlag, 2003.

[22] A. Wolisz. MPEG-4 and H.263 Video Traces for Network Performance Evaluation, Available under http://www-tkn.ee.tu-berlin.de/research/trace/trace.html, 2000.

[23] J. D. C. Little. A Proof of the Queueing Formula $L = \lambda W$. *Operations Research*, 9, 1961.