Advance Reservations for Predictive Service[1]

Mikael Degermark[2], Torsten Köhler[2 3], Stephen Pink[2 3], and Olov Schelén[2]

[2]Dept. of Computer Science
University of Luleå
S – 971 87 Luleå, Sweden
{micke,olov}@sm.luth.se

[3]Swedish Institute of Computer Science
PO box 1263,
S – 164 28 Kista, Sweden
{steve,tk}@sics.se

Abstract. We extend a measurement-based admission control algorithm suggested for predictive service to provide advance reservations for guaranteed and predictive service while keeping the attractive features of predictive service. The admission decision for advance reservations is based on information about flows that overlap in time. For flows that have not yet started, the requested values are used, and for those that have already started measurements are used. This allows us to estimate the network load accurately for the near future. To provide advance reservations we ask users to include durations in their requests. We provide simulation results to show that predictive service with advance reservations provides utilization levels significantly higher than those for guaranteed service.

1 Introduction

Real time multimedia applications will share future networks with traditional data applications. To provide quality-of-service (QoS) for real time applications, it is likely that resource reservations will have to be made in the network. Current resource reservation protocols allocate resources just before communication begins, e.g., ST-2 [5] and various ATM signaling protocols reserve resources during connection establishment. This model of communication may not fit the needs of future network users, [4] pp. 44–45.

Resource reservations should be optional and decoupled from the starting time of the session. One should be able to reserve resources prior to or during a network session depending on when a specific service is needed. Users may know far in advance of their needs and would like to plan their activities by making advance reservations to ensure that they are not blocked by the network's admission control mechanism. Imagine some users with busy schedules in different time zones who want to have an important teleconference on a resource-limited network at an agreed time in the near future. They should be allowed to make an advance reservation given that they know when and for what duration their teleconference will take place.

[1] This work was supported by a grant from the Center for Distance Spanning Technology (CDT), Luleå, Sweden

In this paper we will look at an important candidate for an admission control algorithm originally proposed at this workshop some years ago [2] and later refined in [3], for a new kind of service called predictive service [1]. Predictive service provides quality of service for applications that can tolerate some loss such as real time digital audio and video applications that can adjust their playback points in response to jitter in the network. The efficiency gain of predictive service comes from allowing more flows into the network than guaranteed service, thus providing more sharing and lower cost. The architecture described in [1] supports guaranteed and predictive service, but not advance reservations.

The possibility of making advance resource reservations should be a part of a communication architecture to provide better service to the users. Whether advance reservations are actually needed depends on future resource scarcity. Where resources are plentiful, not even immediate reservations may be necessary, but where resources are scarce enough to justify reservations at all, it makes sense to be able to make them in advance. In this paper we will show that advance reservations can be provided by the network with little overhead.

2 Framework

The service model and the admission control algorithm suggested in this paper are extensions of those presented in [1] and [3]. In [1], the proposed network service interface offers guaranteed service, predictive service and best-effort (ASAP) service. The service interface is simple and relies on token bucket traffic shaping; the source specifies the bucket size b and the token generation rate r. Guaranteed service provides a minimum transmission rate and the queuing delay bound becomes the bucket size divided by the rate. Predictive service provides K different service classes with widely spaced target delay bounds D_i and it is suggested that the target bounds are spaced by an order of magnitude. The bounded quantity is the queuing delay per hop, so it is necessary to add up the target delay bounds at each hop to find the upper bound on the total queuing delay.

To support this service interface a scheduling algorithm is presented in [1]. The guaranteed service traffic is scheduled with weighted fair queuing (WFQ) [7] so that each guaranteed service client has a separate WFQ flow. All the predictive service flows and ASAP traffic share the spare bandwidth in a pseudo–WFQ flow, called flow 0. The available bandwidth for flow 0 is therefore $\mu - \hat{\nu}_G$ where μ is the link bandwidth and $\hat{\nu}_G$ is the measured bandwidth usage for all guaranteed flows over the link. Inside flow 0, there are a number of strict priority classes: one class for each target delay bound and ASAP traffic at the lowest priority. The strict priority scheme implies that queuing delay experienced by higher priority classes will be conveyed to lower priority classes.

Admission control is performed in each switch along the path of a flow. Admission requests will be carried to the switches by an end-to-end resource reservation protocol such as RSVP [6].

3 Duration Intervals

To achieve an efficient scheme for advance reservations we ask that each request includes a duration interval: $I = [t_s, t_e]$, where t_s is when the requested service will start and t_e when that service will end. The intervals are necessary to determine which requests overlap and when the reserved resources will be released.

We have extended the service interface so that each admission request includes a duration interval. Requests for immediate admission will specify *now* as their starting time. If a requested duration is too short, it should be possible to renegotiate the request by calling the admission algorithm again. If this request is rejected the session may continue but not necessarily with the same service quality.

If a requested duration is too long, resources are over-reserved. This reduces the chances to grant admission to other advance reservations. Fortunately, immediate reservations can be granted to a large extent anyway. This is because the measurement procedure of predictive service automatically detects unused capacity once a flow is active. Therefore, over-reservation has little impact on the total utilization as long as there are some immediate requests for admission. In addition, there is an option for clients to explicitly close the requested service before the duration expires.

4 Admission Control Decision for Advance Reservations

The admission decision for predictive service is based on requested rates for flows that have not yet started and on measured rates for currently active flows. If there are no advance reservations and a request for immediate admission arrives, our extended conditions give the same result as the conditions stated in [3].

Figure 1 is a snap-shot of admitted flows in a time/bandwidth diagram. Flows a, b and c are currently active and we have measurements of their rates and maximum delays which are used as predictions of their future behavior. When a new admission request arrives, admission is granted if the new flow would not cause any delay bounds to be violated or bandwidth limits to be exceeded. The admission conditions only consider flows that overlap with the new flow (b,c,d,e,g,h), using measured bandwidth if they have started or, otherwise, bandwidth requests; we call this the *estimated* bandwidth. The conditions are checked at all points where new flows begin (t_s, t_x and t_y).

For reservations in the distant future the number of currently active overlapping flows is small and admission decisions are based mainly on requested rates. In the near future the number of currently active overlapping flows is probably large and admission decisions are based mainly on measured values. So, in the distant future, the admission criteria are conservative, but as time proceeds more overlapping flows will become active and we get better estimates of bandwidth usage. Thus, as we get closer in time to the point at which a flow with an advance reservation is to begin, we have a more accurate knowledge of the network load

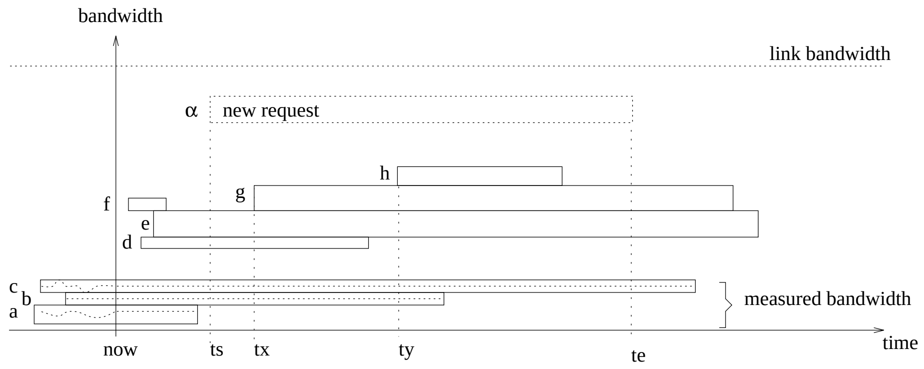

Fig. 1. Snap-shot of reservations

and more flows can be admitted. Requests for immediate reservation can fill up the remaining bandwidth.

4.1 Admission Criteria

A client may request admission for predictive service in one of the classes 1 to K (where class K packets are scheduled at the lowest priority level), or for guaranteed service. The following notation will be used in the formulas[2] describing our admission criteria:

$\nu_{G(t)}$ estimated bandwidth for guaranteed flows at time t
$\nu_{P(t)}$ estimated bandwidth for predictive flows at time t
$\nu_{P_i(t)}$ estimated bandwidth for flows in predictive class i at time t
$R_{G(t)}$ requested bandwidth for guaranteed flows at time t
$\hat{D}_j$ measured delay in predictive class j
$B_j(t)$ bucket size sum for not yet started flows in predictive class j.

Predictive service: When a client requests service in predictive class k for a flow α, shaped by token bucket filter $(r_k^\alpha, b_k^\alpha, I^\alpha)$, the admission control algorithm performs the following checks:

- Determine if the bandwidth usage, after adding the new load r_k^α, will exceed the available link capacity $v\mu$ during the requested interval I^α:

$$v\mu > \max_{t \epsilon I^\alpha} \left(r_k^\alpha + \nu_{G(t)} + \nu_{P(t)} \right) \quad (1)$$

  The available link capacity, $v\mu$, is determined by the link capacity μ and the link utilization target v, that is tunable.
- Determine whether the worst possible behavior of the new flow and the other flows that have not yet started can cause violation of delay bounds for predictive service classes k through N.

[2] These formulas are extensions of those presented in [3]

The worst case is when all predictive service flows flush their entire token buckets simultaneously in one burst. The resulting queue will be emptied according to the available bandwidth.
- check the delay bound, D_k, of the same priority level:

$$D_k > \max_{t \epsilon I^\alpha} \left(\hat{D}_k + \frac{b_k^\alpha + \sum_{i=1}^{k} B_i(t)}{\mu - \nu_{G(t)} - \sum_{i=1}^{k-1} \nu_{P_i(t)}} \right) \tag{2}$$

- check the delay bound of the lower priority levels, i.e., D_j where $k < j \le K$.

$$D_j > \max_{t \epsilon I^\alpha} \left(\frac{\hat{D}_j \left(\mu - \nu_{G(now)} - \sum_{i=1}^{k-1} \nu_{P_i(now)}\right) + \left(b_k^\alpha + \sum_{i=1}^{j} B_i(t)\right)}{\mu - \nu_{G(t)} - \sum_{i=1}^{k-1} \nu_{P_i(t)} - r^\alpha} \right) \tag{3}$$

Guaranteed service: When a client requests guaranteed service for a flow α shaped by $(r_G^\alpha, b_G^\alpha, I^\alpha)$, the admission control algorithm first performs the total bandwidth check expressed in (1), then the following checks are performed:

- Determine whether the requested bandwidth of all guaranteed service flows will exceed link capacity:

$$v\mu > \max_{t \epsilon I^\alpha} \left(r_G^\alpha + R_{G(t)}\right) \tag{4}$$

- Determine that the delay bounds of each predictive service class is still observed when the remaining bandwidth is decreased (estimated bandwidth for guaranteed flows and for predictive classes with higher priority is subtracted from the link bandwidth).

$$D_j > \hat{D}_j * \max_{t \epsilon I^\alpha} \left(\frac{\mu - \nu_{G(now)} - \sum_{i=1}^{j-1} \nu_{P_i(now)}}{\mu - r_G^\alpha - \nu_{G(t)} - \sum_{i=1}^{j-1} \nu_{P_i(t)}} \right) \qquad 1 \le j \le K \tag{5}$$

4.2 Operation of Admission Control Algorithm

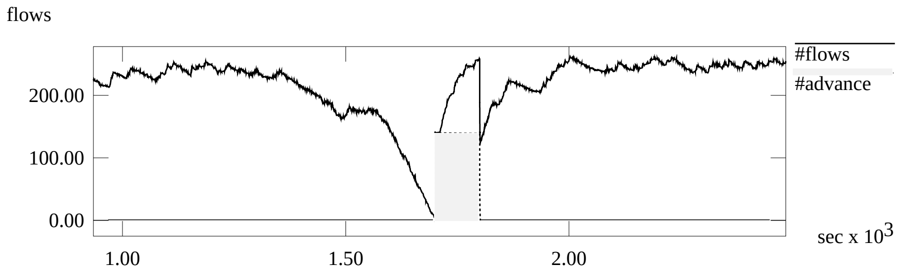

Fig. 2. Plot of number of active flows when there is a large block of flows admitted in advance.

Figure 2 illustrates how our admission control algorithm operates. At time 1700 a large number of sources start. These sources, which were admitted in advance, have reserved all of the available bandwidth and all finish at time 1800. There is a background of sources asking for immediate admission with predictive service. The figure clearly shows that the number of active flows goes down to zero just before time 1700 to honor the resource commitments to the previously admitted sources. At time 1700 the number of active flows increases sharply as the previously admitted sources begin to transmit and then increases further as those sources are measured and more sources can be admitted.

5 State Requirements

The information needed to make advance admission decisions in a switch is summarized in figures 3 and 4. Figure 3 shows the cumulative requested bandwidth

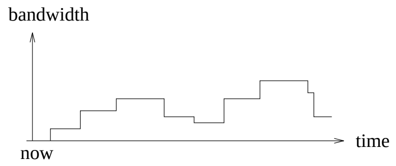

Fig. 3. Cumulative requested bandwidth

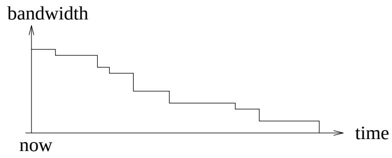

Fig. 4. Predicted bandwidth use of currently active flows

admitted to flows that have not yet started. Figure 4 shows the predicted bandwidth use of currently active flows. Present measurements are used as predictions of future bandwidth use for those flows. We need to keep state corresponding to these diagrams for the guaranteed flows collectively and for each predictive service class individually.

An attractive feature of the original admission control algorithms [2] [3] is that the only state needed is the current bandwidth use for all guaranteed flows plus maximum delay and bandwidth use for each predictive service class. A straightforward implementation for advance reservation would keep an amount of state proportional to the number of active flows plus the number of flows reserved in advance. Aggregation methods, however, can decrease the amount of state needed: flows that start or finish at the same time, or nearly the same time, can be treated collectively.

There is a tradeoff between the amount of state saved by aggregating requests and the flexibility of making requests. A simple way to aggregate requests is to use time slots. Duration intervals may then start and finish only at certain points in time. A disadvantage with this scheme is internal fragmentation: clients may have to reserve longer intervals than they will actually use.

6 Simulations

Our simulations aim to show that adding advance reservation capability to the admission control algorithm for predictive service does not decrease utilization levels very much. We have simulated a single link topology using two different source models. We have done simulations of scenarios with immediate reservations only, and with both immediate and advance reservations. We have also examined the effects of aggregating state for active flows with similar finishing times.

6.1 Simulated Topology

The simulated topology is a single 10 Mbit/s bottleneck link connecting two routers. A number of sources are connected to one of the routers with links with infinite bandwidth. All sources send data to a sink connected to the other router. Our data comes from the upstream router R (fig 5).

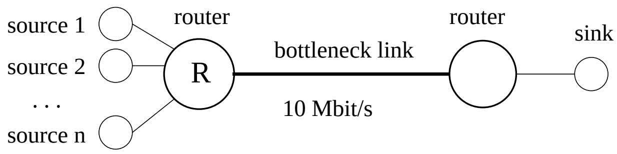

Fig. 5. Simulated topology

6.2 Source Model Parameters

We use two kinds of sources, both generate packet trains at some peak rate p. The train length is exponentially distributed with mean N. The time between packet trains is also exponentially distributed with mean I. The ratio between the peak and average rate, p/a, can be calculated from those values.

All sources regulate their output with a token bucket filter with token generation rate r and bucket depth b. Each token is worth 1000 bits which is equal to the packet size; sending one packet consumes one token. The token bucket filter is designed so that there should always be a token available when the source wants to output a packet. However, if the bucket is empty the packet is queued until a token is available.

All source parameters are listed in table 1. In the table, D is the maximum delay for a guaranteed flow, calculated from the token bucket parameters. D_j is the requested delay bound when the source asks for predictive service. The router supports two predictive service classes, one with a delay bound of 16 ms and the other with a delay bound of 160 ms.

Model Name	Model parameters				Token bucket		Delay bounds	
	p pkts/sec	I msec	N pkts	p/a	r tkns/sec	b tkns	D msec	D_j msec
EXP1	64	325	20	2	64	1	16	16
EXP2	1024	90	10	10	320	50	160	160

Table 1. Source model parameters

6.3 Flow Generation

Sources ask for admission according to a poisson process; the times between admission requests are exponentially distributed with a mean of 400 ms. The requested duration intervals are also exponentially distributed with a mean of 300 seconds. For sources that ask for admission in advance, the times between the admission request and the start of the duration interval are exponentially distributed with a mean of 300 seconds. Note that with these parameters, the offered load is much larger than the available bandwidth, so most admission requests are rejected.

A source requests admission by sending a setup packet containing the desired service type and token bucket parameters towards the destination. If all routers along the path grant admission, the source transmits during the requested interval and then it stops.

6.4 Measuring Process

The measuring process estimates current bandwidth utilization $\hat{\nu}$ and experienced maximum delay $\hat{D}$ in the same way as in [3]. When deciding whether

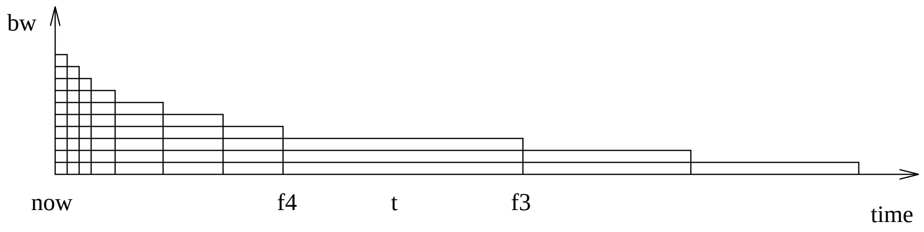

Fig. 6. Estimated future bandwidth use for currently active flows

to admit an advance reservation, the algorithm needs estimates of bandwidth utilization in the future, e.g., at time t in fig 6. This is done by continually estimating current bandwidth utilizations and using these as predictions of future utilization. The estimates are obtained by a straightforward extension of the measuring process in [3]. The packet rate of every active flow is sampled; these rates are then used as in [3] to obtain estimates of bandwidth utilization between finishing points of flows. The sum of the rates of the bottom three flows in figure 6 are used to estimate $\hat{\nu}$ between *f3* and *f4*, i.e., the finishing points of flows three and four from the bottom. This procedure ensures that the estimates are conservative in the distant future and accurate in the near future where many currently active flows will still be active.

In a straightforward implementation, calculating the estimates is linear in the number of flows. To avoid keeping track of every individual flow and reduce the overhead in calculating the estimates, we have experimented with aggregating flows that finish at about the same time. The bandwidth utilization of the aggregated flows is then estimated collectively.

6.5 Simulation Results

To verify our simulation environment we first replicated some relevant results from [3] in our simulator. In this first set of simulations, all sources in a single simulation conformed to the same source model and all requested immediate admission for the same type of service. The results of these simulations are summarized in table 2 under IMM. In table 2; *util* is the utilization of the

Name	Model	Serv	util	delay	# sources (avg)			Measuring params		
			%	(ms)	act	adv	adm	T (s)	S (s)	A (s)
IMM	EXP1	G	45	2.6	140	—	—	5.0	0.80	—
ADV	EXP1	G	44	2.3	137	137	282	5.0	0.80	—
IMM	EXP1	P	78	2.3	244	—	—	5.0	0.80	—
ADV	EXP1	P	68	1.9	213	160	265	5.0	0.80	—
GRA	EXP1	P	70	1.9	219	164	249	5.0	0.80	32
GRA	EXP1	P	75	2.5	232	153	240	5.0	0.80	128
IMM	EXP2	G	28	9.3	28	—	—	1.0	0.12	—
ADV	EXP2	G	27	8.4	27	27	109	1.0	0.12	—
IMM	EXP2	P	76	37.0	75	—	—	1.0	0.12	—
ADV	EXP2	P	59	13.4	58	37	124	1.0	0.12	—
GRA	EXP2	P	54	11.1	54	37	122	1.0	0.12	32
GRA	EXP2	P	50	11.5	49	33	123	1.0	0.12	128

Table 2. Simulation results

bottleneck link and *delay* is the maximum experienced queuing delay. *# sources* are averages of flow counts; *act* is the average number of sources that were transmitting. The *measuring params* are the size of the T and S windows used in the measuring process (see [3]).

The utilization target v was 90% in all simulations. All simulations ran for at least 3000 seconds simulated time. The data in table 2 comes from the second half of the simulated time. Visual inspection confirmed that no startup transients remained at that time.

In the simulations with advance reservations, 50% of the sources asked for immediate admission and 50% for admission in advance. The choice was random. All sources conformed to the same model and asked for the same type of service. These simulation results are summarized in table 2 under ADV. There, *adv* is the average number of sources that were transmitting and were admitted in advance, and *adm* is the average number of sources that were admitted in advance but have not begun transmitting.

The GRA simulations are similar to ADV, the only difference being the measuring process: all flows that finish within the same A seconds are aggregated and measured collectively. This also implies that for purposes of admission control, finishing times are rounded upwards to the nearest A seconds. In the table, A is the granularity of the measuring process.

6.6 Discussion

Our simulations clearly show that predictive service with advance reservations provides higher network utilization than guaranteed service with advance reservations. They also show that adding advance reservation capability to predictive service decreases bandwidth utilization. The levels are not much lower though for smooth traffic, but for bursty traffic the utilization level decreases more. When the fraction of sources asking for advance admission is lower the decrease in utilization is lower. E.g., in simulations when 10% of the sources (instead of 50%) ask for admission in advance, the utilization is 69% for the bursty EXP2 traffic.

The reason for the decrease in utilization is that advance reservations will block requests for immediate admission. This blocking effect is larger when the sources are bursty since the token bucket parameters are larger. Moreover, when token buckets are deep the admission decision is based on delay considerations more than on available bandwidth. To make a good admission decision in this case, the algorithm would need to know how much each flow contributes to the current queuing delay. This would enable the algorithm to estimate future delay since it knows which flows will be active at any future time. Instead, the algorithm uses the current delay as an estimate of future delay. Since this is a very conservative estimate, network utilization suffers.

Aggregation of flows in the measuring process increases utilization for the smooth traffic generated by EXP1 sources, but decreases utilization for the burstier EXP2 sources. The utilization increase for smooth sources may be due to the fact that sources asking for admission very soon in advance (i.e., they wish to start very soon after admission) are denied admission. This would improve utilization since such reservations would block sources asking for immediate admission. The utilization decrease for the bursty EXP2 sources is connected to a higher rejection rate of sources asking for immediate admission.

An interesting observation is that for guaranteed service, sources asking for immediate admission are almost completely shut out by sources asking for admission in advance. This is due to the fact that for guaranteed service the admission decision is based on requested values only, regardless of estimated bandwidth use.

The sources asking for admission in advance are admitted first and so can starve out sources asking for immediate admission since no bandwidth is freed when the sources with guaranteed service begin to transmit.

7 Setting up Advance Reservations using RSVP

With some minor changes, RSVP [6] could be used to set up advance reservations. RSVP is a receiver initiated reservation protocol supporting unicast and multicast reservations along a distribution tree. It can be used for setting up advance reservations in almost the same way as it is used for setting up immediate reservations. To establish an advance reservation for a multi-party session the senders have to announce their session by periodically sending announcement messages (in RSVP terms, "path" messages) down a multicast tree. Receivers respond to those announcements by sending reservations towards the senders. The resources have at this point been reserved for some time in the future. At the time the session starts, resources are allocated and the service to each session participant increases from best-effort to the requested quality.

For reservations made far in advance, there is potentially a very large number of path and reservations messages that must be sent before the session begins. To reduce overhead, the frequency of sending these messages should start low and increase as the time of the session approaches. RSVP could support advance reservations efficiently while allowing the admission control algorithm and measuring process to aggregate sessions if the following two minor changes are made:

- To support advance reservations the flow specification carried by RSVP path and reservation messages should include session durations. A sender will state a duration for the session and the receivers are free to reserve any interval within that duration. Since senders may lengthen or shorten durations, special wildcard durations can be used by the receivers to follow the changes made by the sender.
- To cancel a reservation, RSVP should provide the original flowspec in the interface between RSVP and the admission control mechanism. This is because our admission control algorithm aggregates requests for sessions of similar duration to save state and for measuring purposes. We propose that RSVP provide the original flowspec when making a call to the admission control mechanism to delete a session.

8 Conclusions

We have shown how the predictive service admission control algorithm developed in [2] and [3] can be extended to support advance reservations provided that requests for admission specify the duration of their reservation. The extended admission control algorithm proposed in this paper relies on knowledge of which flows overlap in time with the flow that requests advance reservation, measuring those overlapping flows that are active, and assigning the requested rate to the flows that have not yet started. Thus, more requests for a certain duration of time can be granted as we get closer to that duration of time increasing sharing and lowering cost for those flows that occupy that duration. We have also suggested ways to minimize the amount of state information necessary to provide advance reservations, and to simplify the measuring process that estimates future bandwidth use.

Our simulations show that predictive service with advance reservations provides higher network utilization than guaranteed service with advance reservations. They also show that adding advance reservation capability to predictive service decreases bandwidth utilization. The levels are not very much lower though, and they are still significantly higher than for guaranteed service. The decrease in utilization is due to the fact that advance reservations will block sources asking for immediate admission. This blocking effect is larger for bursty sources which request more resources.

References

1. D. Clark, S. Shenker, L. Zhang: *Supporting Real-Time Applications in an Integrated Packet Services Network: Architecture and Mechanism.* Proc. ACM SIGCOMM'92, 1992.
2. S. Jamin, D. Clark, S. Shenker, L. Zhang: *Admission Control Algorithm for Predictive Real-Time Service.* Proc. 3rd International Workshop on Network and Operating System Support for Digital Audio and Video, November 1992.
3. S. Jamin, P. Danzig, S. Shenker, L. Zhang: *A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks.* Submitted for publication, February 1995.
4. C. Partridge, S. Pink: *An Implementation of the Revised Internet Stream Protocol.* Internetworking Research and Experience, 3(1), March 1992, pp. 27–54.
5. C. Toplocic: *Experimental Internet Stream Protocol, Version 2 (ST-II).* RFC 1190, October 1990.
6. L. Zhang, S. Deering, D. Estrin, S. Shenker, D. Zappala: *RSVP: A New Resource ReSerVation Protocol.* IEEE Network Magazine, September 1993, pp. 8–18.
7. A. Demers, S. Keshaw, S. Shenker: *Analysis and Simulation of a Fair Queuing Algorithm.* Journal of Internetworking: Research and Experience, 1, 1990, pp. 3–26.

This article was processed using the LaTeX macro package with LLNCS style