An Admission Control Algorithm for Predictive Real-Time Service (Extended Abstract)

Sugih Jamin[1], Scott Shenker[2], Lixia Zhang[2], and David D. Clark[3]

[1] Department of Computer Science, University of Southern California
[2] Palo Alto Research Center, Xerox Corporation
[3] Laboratory for Computer Science, Massachusetts Institute of Technology

1 Introduction

The technical and legal developments of the past decade have created the possibility of merging digital telephony, multimedia transport, and data communication services into a single Integrated Services Packet Network (ISPN). For an ISPN to be able to support this diverse set of applications, it must be capable of providing real-time service. Such real-time service will require network switches to implement special packet scheduling and flow admission control algorithms. In Reference [1] we proposed such a real-time scheduling algorithm (which, for convenience, we will call the CSZ scheme) for ISPN's, but did not fully define the required admission control algorithm. The purpose of this paper is to define an admission control algorithm for the CSZ scheme, and then analyze its performance through simulation. In the next section, we motivate the need for admission control and examine some of the issues involved in providing one. In Section 3 we describe a prototypical design, and, in Section 4, we report on some simulation results.

2 Real-Time Services and Admission Control

The ability of the network to meet its real-time service commitments is directly related to the criteria the network uses in deciding whether to accept another request for service. The admission control algorithm embodies these criteria and makes the decision to accept or reject a request for service.

The traditional form of real-time service provides a hard or absolute bound on the delay of every packet; in Reference [1], we labeled this *guaranteed* service. The hard bound in guaranteed service reflects the worst-case behavior of network traffic. Even though the algorithms themselves may be very complex, the underlying principle behind admission control algorithms for such guaranteed service is conceptually simple: if this request for service is granted, will the worst case behavior of the network violate any delay bound? An example of this form of admission control can be found in Reference [3]. In the CSZ scheme, guaranteed service is provided by the weighted fair queueing algorithm (WFQ) described in Reference [2], also called generalized processor sharing in Reference [6], which assigns a share of the bandwidth to each active flow; the admission control criterion is merely that the sum of the previously assigned bandwidths plus the bandwidth requested by the prospective flow does not exceed the link capacity.

In Reference [1] we made the observation that some real-time applications, such as many audio and video applications, can tolerate occasional violations of the delay bound. In some cases, this tolerance allows the application to adapt to moderate variations in the delivered delay, and thus take advantage of the fact that the actual delivered delay is often much less than the bound on delay. Such applications do not require an absolute delay bound, but merely desire a reliable bound; i.e., some very high fraction of the packets obey the bound. To meet the needs of these tolerant (and perhaps adaptive) clients, the CSZ scheme provides a type of service called predictive service which offers a fairly reliable but not absolute bound. When making an admission control decision, the validity of these bounds are assessed using the measured characteristics of the current traffic load, rather than the theoretical worst-case behavior. Thus, the admission control algorithms for predictive service must reliably predict the future behavior of the network based on its measured past behavior. The scheduling algorithm for this predictive class is described in Reference [1]; it attempts to minimize the maximal delays actually experienced, but does not guarantee an absolute maximum delay bound. Note that predictive service does not even provide a probabilistic bound. Probabilistic bounds, as discussed in Reference [3], are based on the statistical characterizations of the current and requesting traffic sources; these characterizations are given to the network by the flows when they request service. In predictive service, the validity of the delay bounds are based on actual network measurements of the current aggregate traffic load.

For guaranteed service, a flow can request any amount of bandwidth and thereby flexibly tune its resultant delay bound. In contrast, the delay bounds for predictive service are less flexible; each switch has a few predictive service classes which have pre-established target delay bounds. These delay bounds will typically be chosen to be roughly an order of magnitude apart, and prospective flows can choose which class of predictive service they desire. The key to predictive service is then having the ability to accept enough traffic to efficiently utilize the network, but yet not accept so much traffic that these target delay bounds are violated more than a tiny fraction of the time. Thus, the viability of predictive service depends on its admission control algorithm. We are not aware of any literature on admission control algorithms of exactly this type. The most similar work is found in Reference [5] where there is a small set of well-characterized traffic sources and the admission control is based on a feasibility graph which is derived from either simulations or approximate calculations. The key difference between that work and what we present here is that we are not limiting ourselves to a small and well-characterized set of traffic sources, and thus cannot rely on pre-existing measurements or calculations; in our situation, measurements of a traffic source can only be performed after the flow has started, and the characterization of the flow is limited to a perhaps rather loose worst-case description. Reference [4] also discusses related work on admission control. While the main emphasis is on analytical calculations for admission control, Reference [4] does raise the possibility of basing admission control decisions on measurements of the current aggregate network load.

The advantage of offering predictive service is that one can more fully utilize the network. The disadvantage is that the admission control algorithm, since it is based on measurement data and not on worst-case characterization of the traffic, is much more difficult to design and justify. In particular, in order to contend that predictive service is viable, one must demonstrate affirmative answers to the following two questions. First, can one provide reliable delay bounds for predictive service with an admission control algorithm that is based on measured history of the network? Second, if one can indeed achieve reliable delay bounds, does predictive service allow a higher level of network utilization as compared to the more traditional guaranteed service? The purpose of our study is to demonstrate that our approach of providing predictive service with measurement-based admission control can meet these two challenges. To this end, we have designed a prototypical admission control algorithm for the CSZ scheme, which is described in the next section. increase network utilization?

with an admission control algorithm that is based on measured history of the network?

3 Prototypical Admission Control Algorithm

For the network to make a service commitment to a particular client, it must know beforehand some characterization of the client's offered load. Based on the characterization of the offered load and the estimated current load of the network, the admission control algorithm decides whether to accept the new flow. Unfortunately, it is difficult to accurately characterize actual flows in advance. If admission decisions are based solely on clients' load characterization, then the characterizations either have to be very tight (which would severely constrain the burstiness of flows) or otherwise be very loose (which would lead to overly conservative reservations).

Our system is less dependent on the accuracy of a flow's claimed characterization because our admission decision is mainly based on the measured behavior of existing load. Only when processing a new request, before the system has any observed history of the new flow, are the client-specified parameters used in the decision process. Once a flow starts sending, however, the system will use the measured, rather than the given, values to characterize the current load in making future admission decisions.

We require that each real-time flow α be characterized by a token bucket filter with two parameters: the token generation rate r^α and bucket depth b^α. The load of a switch is represented by the measure of both bandwidth utilization $\hat{\nu}$ and maximal experienced queueing delay $\hat{d}$; following Reference [1], we use the convention that the hat symbol denotes measured quantities. Because the system provides different classes of service, each switch keeps separate utilization and delay measures for each class of service. For each predictive service class j, we keep the aggregate utilization $\hat{\nu}_j$, experienced queueing delay of the class $\hat{d}_j$, and a pre-defined delay bound for the class D_j. For guaranteed service, the switch keeps the sum R_G of the reserved rates of guaranteed flows and the aggregate utilization $\hat{\nu}_G$ of all guaranteed flows. Let μ denote the link bandwidth and n denote the total number of predictive classes. Our prototypical admission control algorithm is based on many heuristic and approximate criteria, which are detailed in the equations below. Due to space limitations, we cannot explain their rationale here; they will be more fully explained in a forthcoming paper. We hasten to note, however, that the validity of these heuristic and approximate criteria does not rest on the quality of the underlying rationale but on the ability of the algorithm to perform adequately in practice. With these caveats, our prototypical admission control algorithm is described below.

Consider some incoming flow α requesting real-time service from the network. The decision about accepting the flow depends, of course, on the quality of service requested by the flow. If the incoming flow α requests service in predictive class k, the admission control algorithm performs the following checks:

1. Determine if the bandwidth usage, after adding the new load r^α, will exceed the link capacity:

$$\mu > r^\alpha + \hat{\nu}_G + \sum_{i=1}^{n} \hat{\nu}_i \tag{1}$$

2. Determine whether the worst possible behavior of the new flow (i.e., flushing the entire token bucket in one burst of length b^α) can cause violation of the delay bound of classes at lower or same priority levels:

$$D_j > \hat{d}_j + \frac{b^\alpha}{\mu - \hat{\nu}_G - \sum_{i=1}^{n} \hat{\nu}_1 - r^\alpha} \qquad k \leq j \leq n \tag{2}$$

This specifies how we make the admission control decision when the incoming flow asks for predictive service.

If the incoming flow α requests guaranteed service, the admission control algorithm performs the following checks:

1. Determine that the total bandwidth usage is within capacity using Equation 1, then check that the reserved bandwidth of all guaranteed service flows will not exceed link capacity:

$$\mu > r^\alpha + R_G \tag{3}$$

2. Determine that if all guaranteed flows use up their reserved bandwidth, the network will still be able to meet all the predictive service delay bounds assuming the current predictive load remains constant:

$$D_j > \frac{\hat{\nu}_j \hat{d}_j}{\mu - R_G - r^\alpha} \qquad 1 \leq j \leq n \tag{4}$$

3. Determine that the the delay bounds of predictive service classes are still observed when the remaining bandwidth (the bandwidth not reserved for guaranteed flows) is decreased:

$$D_j > \hat{d}_j \frac{\mu - \hat{\nu}_G - \sum_{k=1}^{j} \hat{\nu}_k}{\mu - \hat{\nu}_G - \sum_{k=1}^{j} \hat{\nu}_k - r^\alpha} \qquad 1 \leq j \leq n \tag{5}$$

The five equations above completely describe our admission control criteria. Their effectiveness requires that the heuristic criteria expressed in the above equations are appropriately conservative.

The above formulae are defined in terms of measured quantities; we now describe our method for measuring bandwidth utilization $\hat{\nu}$ and maximum queueing delay $\hat{d}$. The measurement process uses two tunable constants, T and a; T controls the length of the measurement period, which is expressed in terms of the number of packet departures, and a controls the rate of exponential averaging. When measuring utilization, we measure the average utilization A_j for each class for a period of T packet transmissions; after each measurement period we update the utilization measure $\hat{\nu}_j$ using exponential averaging as follows: $\hat{\nu}_j = (1-a) * \hat{\nu}_j + a * A_j$. To measure the maximum queueing delay for each class, we again consider a measurement period of T packet transmissions, although, as we clarify below, these periods are restarted after each flow acceptance. Define $d_j(i)$ to be the queueing delay experienced by packet i of class j. We update the delay estimate $\hat{d}_j$ upon every packet departure from class j. If the departing packet i is the last packet in the measurement period, we set $\hat{d}_j = MAX_{i'}[d_j(i')]$, where the maximum is taken over all departing packets i' in the present measurement period. If the departing packet i is not the last packet in the measurement period, then we set $\hat{d}_j$ according to the following formula: $\hat{d}_j = MAX[\hat{d}_j, d_j(i)]$. The measurement period length T controls how conservative the measurements of the delays are; increasing T will retain more history, making the $\hat{d}_j$ larger, and thus making the admission control more conservative and the delay bounds more reliable.

Immediately after a new flow is accepted, the measured utilization and queueing delay do not reflect the true load of the network because the measurements do not yet include the load of this new flow. To compensate for this, immediately following flow acceptance we artificially boost the measured values and then allow them to equilibrate. If the newly accepted flow α is in the predictive service class k, then we set $\hat{\nu}_k = \hat{\nu}_k + r^\alpha$ and increase additively the measured delay of predictive classes of priority equal to or lower than that of the new flow:

$$\hat{d}_j = \hat{d}_j + \frac{b^\alpha}{\mu - \hat{\nu}_G - \sum_{i=1}^{j} \hat{\nu}_i - r^\alpha} \qquad k \leq j \leq n \tag{6}$$

If the new flow is a guaranteed service flow, then we set $\hat{\nu}_G = \hat{\nu}_G + r^\alpha$, and scale the measured delay of all predictive classes multiplicatively:

$$\hat{d}_j = \hat{d}_j \frac{\mu - \hat{\nu}_G - \sum_{i=1}^{j} \hat{\nu}_i}{\mu - \hat{\nu}_G - \sum_{i=1}^{j} \hat{\nu}_i - r^\alpha} \qquad 1 \leq j \leq n \tag{7}$$

Since we replace the delay estimates $\hat{d}_j$ with the measured values at the end of a measurement period, we choose to restart the measurement period every time a flow is accepted. Each class has its own measurement period.

4 Simulation Results

Using the same simulator used in Reference [1], modified to generate dynamic arrival and departure of flows with random duration, we have simulated the admission control algorithm described above (with the CSZ scheme used in the switches). Our results can be summarized briefly as follows; in all of our simulations, which we performed on several different network topologies and load patterns, we find that (1) the predictive bounds are reliable (fewer than 0.1% of packets violate the bounds), and (2) if the delay bounds for the two classes are comparable, the level of real-time utilization is higher when using predictive service than when using only guaranteed service. Thus, our results, which are admittedly preliminary, support our conjecture that predictive service is viable.

level of real-time utilization is higher when using predictive service than when using only guaranteed service.

violate the bounds with $T = 1000$).

To augment the above summary, we describe below the detailed results from two specific simulations. The simulations use a simple topology of two hosts connected through two switches. Flows traverse the network unidirectionally from the first host to the second host. The links connecting the hosts to the switches are infinitely fast. The bottleneck link between the two switches has a bandwidth of 10 Mbps and latency of 1 ms. All data packets in the simulation have a uniform size of 1Kbits. The simulations lasted for 30 minutes simulation time, with the measurements from the first 200 seconds discarded. We chose T to be 1000, and set $a = 2^{-5}$.

Datagram traffic is generated by a single TCP source with an unlimited amount of data and a TCP window size of 128 packets. Real-time flow requests are generated with exponential interarrival times, where the average interarrival time is 700 ms. Flow durations are uniformly distributed between 1 and 2 minutes. We use two kinds of real-time traffic sources, both constrained by a token bucket filter. A new flow (either guaranteed or predictive) will choose the first traffic model with probability 0.7, and the second model with probability 0.3.

The first real-time source model is a packet-train source modeled with two-state Markov processes. In each train, a geometrically distributed random number of packets are generated at some peak rate P. Let N denote the average train length and let I denote the exponentially distributed intertrain gap. The average packet generation rate G is given by: $G^{-1} = I/N + P^{-1}$.

The second real-time source model is similar to the first one, except that instead of just sending randomly spaced trains of packets, the source will occasionally send out much larger bursts. At the beginning of every train the source will, with probability p, remain quiescent for enough time to fully replenish its token bucket and then send the whole bucket of packets back-to-back. Although this second source model may not imitate the behavior of any real applications, it is used as a worst-case test of our admission control algorithm.

For our simulations, we let $N = 5$ packets, $G = 80$ packets/sec., $P = 160$ packets/sec., $I = 5/160$ sec., and $p = 0.3$. The token bucket used by all flow sources has a token generation rate r of 120 tokens/sec. with bucket depth b of 30 tokens. Sending each packet consumes one token; packets which do not pass the token bucket filter are discarded. Note that with these parameters for both

Table 1. Average Link Bandwidth Utilization for Mixed Guaranteed and Predictive Real-Time Traffic.

Real-Time Traffic	Guaranteed	Class 1	Class 2	Datagram
89.0%	22.1%	24.9%	42%	10.7%

the flow request generation process and the source model, the offered load is much higher than the link capacity. Thus, most flow requests must be denied. In the first experiment, we considered only a single predictive service class with a delay bound of 20 ms. When all incoming flows request predictive service, the link utilization due to real-time traffic is roughly 92.5% (with the datagram traffic using the leftover bandwidth). Furthermore, none of the 14.8 million packets violated the delay bound. For a guaranteed flow to obtain the same delay bound of 20 ms, it would have to request a reserved rate of 1500 kbits/sec. If all flows requested guaranteed service, the maximal network utilization would be 7.2%. Thus, predictive service in this case allows the network utilization to increase by an order of magnitude, while still providing reliable delay bounds.

The second experiment was designed to test whether the delay bounds are still reliable when faced with a heterogeneous set of service classes. Here, we used two classes of predictive service, with delay bounds of 20 ms and 80 ms, in addition to the guaranteed service. For a guaranteed flow, we further allow it to reserve either its token generation rate r (with probability 0.6) or its peak data generation rate P (with probability 0.4). When a new flow request is generated, there was a 0.2 probability of it being a guaranteed flow, a 0.32 probability of it being a class 1 predictive flow, and a 0.48 probability of it being a class 2 predictive flow. The utilization levels are displayed above in Table 1; there were no violations of the predictive service class delay bounds (out of 4 million class 1 and 6.7 million class 2 predictive service packets). We should also point out that the datagram traffic did not suffer starvation even though no bandwidth was preallocated for it.

Thus, our preliminary evidence indicates that our admission control algorithm allows predictive service to be sufficiently reliable while also increasing the network utilization. In the future we will continue to simulate this algorithm on a wider variety of network configurations. This is certainly not the definitive admission control algorithm. In particular, we hope that our additional simulations will lead to further refinements of the heuristic criteria embedded in the above equations.

References

1. Clark, D. D., Shenker, S., Zhang, L.: Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism. Proc. ACM SIGCOMM '92, August, 1992.
2. Demers, A., Keshav, S., Shenker, S.: Analysis and Simulation of a Fair Queueing Algorithm. Internetworking: Research and Experience, **1** 1990, 3-26.
3. Ferrari, D., Verma, D.C.: A Scheme for Real-Time Channel Establishment in Wide-Area Networks: IEEE JSAC, **SAC-8**:3 1990, 369-379.
4. Guérin, R., Ahmadi, H., Naghshineh, M.: Equivalent Capacity and Its Application to Bandwidth Allocation in High-Speed Networks: IEEE JSAC, **SAC-9**:9, 1991, 968-981.
5. Hyman, J. M., Lazar, A. A., Pacifici, G.: Joint Scheduling and Admission Control for ATS-based Switching Nodes: Proc. ACM SIGCOMM '92, August, 1992.
6. Parekh, A. K.: A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: Tech. Report LIDS-TR-2089, Lab. for Information and Decision Systems, MIT, 1992.