QoS with an Edge-Based Call Admission Control in IP Networks

Daniel R. Jeske, Behrokh Samadi, Kazem Sohraby, Yung-Terng Wang, and Qinqing Zhang

Bell Labs, Lucent Technologies
Holmdel, New Jersey 07733, USA

Abstract. Central to the viability of providing traditional services over IP networks is the capability to deliver some level of end-to-end Quality of Service (QoS) to the applications and users. IP networks continue to struggle to migrate from a cost effective best effort data service solution to revenue generating solutions for QoS-sensitive applications such as voice and real-time video. For the case of a network of Media Gateways controlled by SoftSwitches, we propose the use of a measurement based call admission control algorithm at the edge of the network as an approach to provide a cost effective QoS solution. The proposed method utilizes statistical prediction techniques based on available performance measurements without complex QoS management of the packet network. Simulation analysis shows that significant gains in QoS can be achieved with such an edge-to-edge measurement based approach.

Keywords: QoS, VoIP, CAC, SoftSwitch

1. Introduction

One of the notable recent advances in converged networks is the development of the SoftSwitch (SS) technology. With SS technology, control plane functions and inter-working between packet and circuit switched network signaling services can be implemented with standard based protocols and used to provide among other things, a locus of resource management of the bearer channels through circuit and packet switched networks [1]. Applications of SS technology include the Voice tandem solution for the Public Switched Telephone Network (PSTN) and VoIP solutions for IP end points. A reference network architecture is shown in Fig.1.

For the voice Tandem solution, voice calls originating and terminating within PSTN would be handled by signaling the ingress and the egress SoftSwitch, and then the destination PSTN switch to complete the call setup procedure. While management of voice QoS in the PSTN is well understood, a different matter is providing QoS when a packet network is used as the transport between the gateways, or when IP end points such as SIP or H.323 devices get involved with the call.

Central to the viability of providing traditional services over IP networks is the capability to deliver some level of end-to-end QoS to the applications and users. Many solutions have been proposed and implemented for ATM based packet networks and standards. On the other hand, IP based networks continue to struggle

E. Gregori et al. (Eds.): NETWORKING 2002, LNCS 2345, pp. 178-189, 2002.

to migrate from cost effective best effort data service solution to revenue generating QoS solutions for more demanding applications such as voice and real-time video.

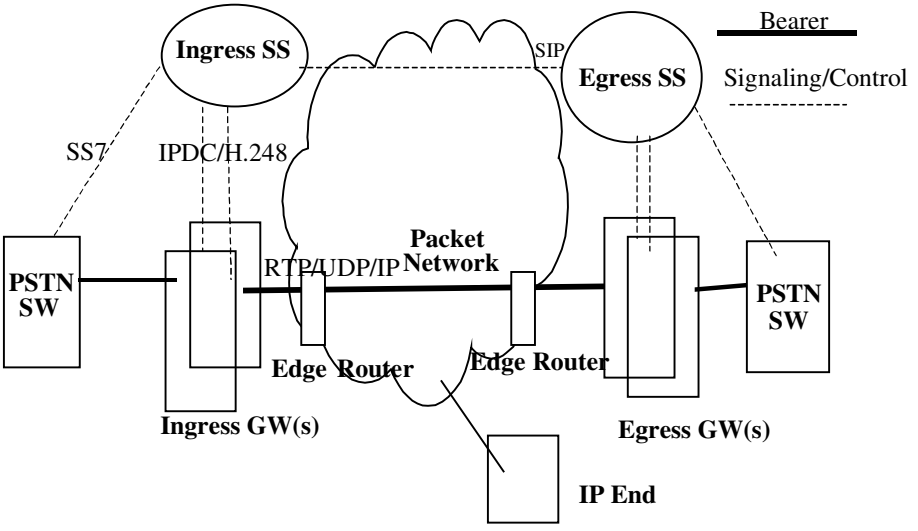

Fig. 1 . Reference Network Architecture

In this paper, we explore the possibility of utilizing available measurements at the edge of packet networks by proposing a distributed edge-to-edge measurement based call admission control (CAC) to provide a cost effective QoS solution. Controlling variables are provided to allow the service providers to influence the tradeoffs of network resource utilization and risks of compromised QoS. The proposed method utilizes statistical prediction techniques using available performance measurements. In Section 2, we discuss the framework of QoS support in IP networks. In Section 3, we present details of the statistical prediction techniques and their application to a measurement based CAC algorithm. Quantitative analysis of the achieved QoS is reported in Section 4 with comparisons to a "best case," where the packet network resources are closely managed to provide a known capacity (see e.g. [2]), and a "worst case," where every call is admitted.

2. Quality of Service Support

For ease of exposition, we use voice service as the application throughout this paper although many of the principles can be extended to include other applications such as multimedia calls. The basic characterization of QoS for the voice application is well-studied (see e.g., [3]). Performance metrics often used to measure QoS for the bearer or user plane of packet telephony include: end-to-end packet delay, delay jitter and packet loss. Note that there are other performance and QoS metrics such as call setup delay, post-dial delay and ring-back delay which are outside the scope of this paper.

The desired end-to-end delay is usually very small for toll quality, the recommended one-way value being 150 ms [4]. Delay jitter is the variation in the delay of consecutive packets. For streaming applications, the delay jitter should be small enough so that the traffic stream can be delivered at a constant rate to the receiver to prevent packet loss. If the delay jitter increases beyond a limit, the packet is regarded as lost. Usually some play-out buffer mechanism at the receiver is provided to absorb the jitter. Buffering adds to the delay and needs to be dimensioned carefully. Packet loss can also occur as a result of buffer overflows in the network. The degree of QoS degradation due to packet loss depends on the application and the coding schemes.

We now turn our attention to the issue of primary interest of this paper: providing QoS support over the packet network. The most developed paradigm is ATM QoS, which includes several key principles that are applicable beyond ATM networks (see e.g. [5,6]). Networks that are IP based are evolving toward use of DiffServ/MPLS which provides the ability to manage QoS through traffic engineering of MPLS Label Switch Paths (see e.g., [7]) and Differentiated Service markings at the edge. However, before these technologies are ubiquitous and network management solutions get developed and deployed, the service providers are left with no choice but to over-provision their networks in order to minimize the risk of degrading QoS.

An economic alternative, which does not require any network resource availability information, is to utilize the measurements available at the edge of the network in the CAC to make the best prediction of whether or not to admit a new call. The objective is as usual: admit as many calls as possible while supporting adequate QoS of the calls already in progress. An obvious implication of not having proactive control of network resources is that there is a period during which the network is vulnerable to sudden changes in congestion levels. We address this and identify other critical issues associated with using available QoS measures in connection with CAC.

3. Measurement Based CAC

In our proposed CAC scenario, both the ingress and egress SS use historical QoS measurements from the originating and terminating Media Gateways (MGW) to make a QoS prediction for each new call. In this paper, we focus on one-way packet loss rate as the primary QoS measure since it is readily available from most MGWs. If either of the one-way predicted QoS measures is unsatisfactory, the new call is rejected, otherwise it is accepted. To implement this approach, sufficiently accurate predictions for one-way packet loss rates are required.

In this section, we describe two statistical models for predicting packet loss rates. The first is a simple exponentially weighted moving average (EWMA) model. The second is an auto-regressive (AR) model which is more sophisticated and was evaluated for use so that we could determine if the EWMA model was overly simplistic. After providing some motivation for each model, we fit each model to traces that were collected, and conclude that the extra complications associated with the AR model may not be justified.

3.1 EWMA Model

Let the sequence $\{Z_k\}_{k=-\infty}^{T}$ denote one-way packet loss rates associated with calls completed over the same path that a new call would utilize if it were admitted into the network. Alternatively, without additional complication, $\{Z_k\}_{k=-\infty}^{T}$ could represent observed aggregate (over all calls on the path) packet loss rates over consecutive and disjoint windows of a prescribed width. Intuitively, one would expect correlation among the Z_k values, and moreover, a degree of non-stationary behavior. A widely used correlation model that accounts for a slowly varying time-dependent mean is a auto-regressive integrated moving average model (ARIMA) [9] which writes:

$$Z_k - Z_{k-1} = \varepsilon_k - \theta\varepsilon_{k-1} \quad . \tag{3.1}$$

Here, the sequence $\{\varepsilon_k\}$ is a white noise process with zero mean and variance σ^2. $|\theta| < 1$ is a parameter that dictates the correlation amongst the Z_k values. In the case of a slowly varying mean function for $\{Z_k\}$, the mean of the left-hand-side of (3.1) is zero and the sequence of first-order differences, $D_k = Z_k - Z_{k-1}$, is a stationary process. It is easy to verify that θ is the lag-1 auto-correlation of the $\{D_k\}$ sequence.

At time *T*, the best predictor of Z_{T+k} ($k \geq 1$) is the conditional [given $\{Z_k\}_{k=-\infty}^{T}$] expected value of Z_{T+k}, say $\hat{Z}_{T+k}$. It can be shown [9] that $\hat{Z}_{T+k} = (1-\theta)Z_T + \theta\hat{Z}_T$, and $\hat{Z}_{T+k}$ is a weighted average of the $\{Z_k\}_{k=-\infty}^{T}$ sequence. Moreover, the weights die off exponentially. Consequently, $\hat{Z}_{T+k}$ is frequently referred to as an exponential weighted moving average (EWMA) of the $\{Z_k\}_{k=-\infty}^{T}$ sequence. Note that under the EWMA model, the *k*-step ahead predictor is independent of *k*. This is a convenient result for our CAC application. In particular, if *T* calls have completed when a new call arrives, the new call will not be the (*T*+1)st completed call in the sequence, but rather will be the (*T*+*D*+1)st completed call in the sequence, where *D* is a random variable representing the number of completed calls during the holding time of the new call. What we need is $\hat{Z}_{T+D+1}$, which by the above property is simply $\hat{Z}_{T+1}$. With the EWMA model, we are thus able to avoid the difficulty associated with *D* not only being unknown, but also being a random variable.

For many EWMA applications, the value of θ is chosen based on intuitive feelings of how much the past should be weighted. If Z_k is rapidly changing, then $\theta = 0.2$ is a common choice, while if they are not changing very fast $\theta = 0.8$ is a common choice. Since θ is the lag-1 auto-correlation of the $\{D_k\}$ sequence, it could be estimated periodically from the observed packet loss measurements, possibly even using a sliding-window scheme. Alternatively, θ can be computed adaptively, changing at each prediction epoch (a new call arrival in our case) [10,11].

3.2 AR Model

An alternative to the EWMA model (3.1) is a *p*-th order auto-regressive model, AR(*p*). The AR(*p*) model relates the current value of a process to the proceeding *p* values and a white noise innovation term. In particular, we have

$$Z_t - \mu = \sum_{k=1}^{p} \phi_k (Z_{t-k} - \mu) + \varepsilon_t \tag{3.2}$$

where μ is the mean of the $\{Z_k\}$ process and the $\{\phi_k\}_{k=1}^{p}$ are the so-called auto-regressive coefficients. Unlike the EWMA model, the AR(*p*) model assumes the $\{Z_k\}$ process is stationary. (Recall that the EWMA model assumes the first-order differences of the $\{Z_k\}$ process is stationary.)

At time *T*, the best-unbiased predictor of an observation *D*+1 steps into the future, say $\hat{Z}_{T+D+1}$, is the conditional expected value of Z_{T+D+1}, given $\{Z_k\}_{k=-\infty}^{T}$. It can be shown [9] that $\hat{Z}_{T+D+1} = \mu + \sum_{j=1}^{p} \pi_j^{(D+1)} (Z_{T+1-j} - \mu)$, where coefficient $\pi_j^{(D+1)}$ is computed in a bootstrap recursive manner according to: $\pi_j^{(1)} = \phi_j$, $\pi_j^{(2)} = \phi_1 \pi_j^{(1)} + \phi_{j+1}$, $\pi_j^{(3)} = \phi_1 \pi_j^{(2)} + \phi_2 \pi_j^{(1)} + \phi_{j+2}$, … , $\pi_j^{(D+1)} = \sum_{i=1}^{D} \phi_i \pi_j^{(D+1-i)} + \phi_{j+D}$. Unlike the EWMA model, the prediction equation explicitly depends on the unknown random variable *D*, as is clearly evident by the general form of $\hat{Z}_{T+D+1}$. The best we can do is replace *D* by its mean value, introducing another source of variability into the prediction error.

Use of $\hat{Z}_{T+D+1}$ requires an estimate of μ and $\{\phi_k\}_{k=1}^{p}$. Moreover, these estimates must be updated periodically to reflect the changing conditions of the underlying network. Let $Z_1, \ldots, Z_n$ denote the set of observations corresponding to a particular update interval. The estimate of μ is simply $\bar{Z} = \sum_{t=1}^{n} Z_t / n$ and method of moment estimators of $\{\phi_k\}_{k=1}^{p}$, say $\{\hat{\phi}_k\}_{k=1}^{p}$, can be obtained by solving the $p \times p$ linear system of Yule-Walker equations [9] which utilize the first *p* sample autocorrelations.

In order to estimate a sample autocorrelation reliably from a statistical point of view, at least 50 observations are recommended, implying $n > 50 + p$. On the other hand, the time required to collect *n* observations is n/λ, where λ is the call arrival rate (arrivals/minute), and we need $n/\lambda < S$, where *S* is the duration (minutes) over which we are willing to assume the network is stationary. Hence, we require $(50 + p) < n < \lambda S$, an interval constraint on the magnitude of *n*. (For example, if *p*=10, $\lambda = 10$ calls/minute and $S = 10$ minutes, then we would require $60 < n < 100$.)

An additional constraint linking the call arrival rate, the call holding time and the duration of the stationarity interval derives from the following conditions. First, the new call arrival should complete within the interval of stationarity, and second, the *p* calls used for the packet loss prediction for the new call arrival should arrive and complete within the interval of stationarity (see Figure 2). It follows that $p/\lambda + 2HT < S$, where *HT* is the average call holding time. (For example, if *p*=10, $\lambda = 10$ calls/minute, and $S = 10$ minutes, then we must have $HT < 4.5$ minutes.)

Since AR(p) has a higher dimensional parameter space than EWMA, we could expect more precise predictions from it. However, its application is hard in that D is a random variable and the best we can do is replace it by its expected value, $\lambda \times HT$, which would seem to offset some of the additional precision. Moreover, it is clear that the AR(p) model is more cumbersome to implement. In particular, the need to frequently update the parameter estimates is a drawback since at each update epoch, the Yule-Walker equations must be solved and then the bootstrap recursive scheme must be used to obtain the necessary prediction coefficients required by $\hat{Z}_{T+D+1}$. These observations raise the question as to whether or not the extra precision seemingly available from the AR(p) model is significant enough to justify its added complexity. We examine this question in the next section by fitting each model to two different packet loss traces and comparing their respective prediction capabilities.

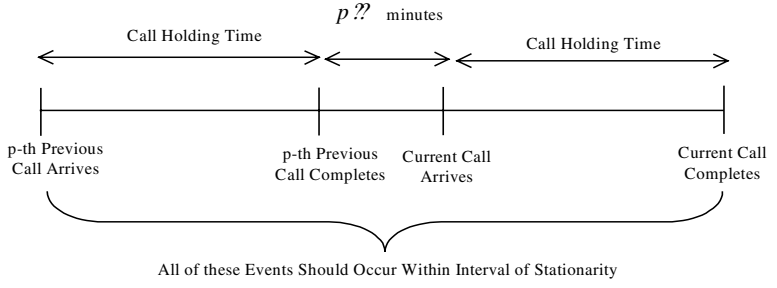

Fig. 2. Feasibility Condition for AR(p) Model

3.3 Empirical Comparison of Models

To compare the relative prediction accuracy of the EWMA and AR(p) models, we applied them to two packet loss traces which were collected from two different IP networks. Below we describe the traces and the results obtained from the fitting analyses. We show that the simpler EWMA models compete very well with the AR(p) model, thereby making a case for their use in real applications.

3.3.1 University of Massachusetts Trace. The first packet loss trace we examined was obtained from the University of Massachusetts at Amherst (UMASS). The trace is one of many traces collected by sending out packet probes (RTP headers) along unicast and multicast end-to-end connections over the public Internet [8]. The packets were sent out every 20*ms* between UMASS and UCLA. The trace is a binary time series with zeros and ones indicating whether the packet probe arrived successfully or not. There are total 358,000 packets in the trace (a two-hour trace). Post processing on the trace divided it into 179 time intervals. Each time interval represents about 40 seconds and 2000 packets. The packet loss percentage of each interval was used as Z_k, in our analysis. Since there are no overlapping intervals, the analysis is window-based in terms of the generation of the time series and the prediction. The estimation algorithms, EWMA or AR(p), apply to either case**.**

Figure 3 shows the analytical results of the packet loss estimation using three different models. AR(5) is a 5th order AR prediction model with coefficients (0.65, -0.26, 0.34, -0.07, 0.22). SEWMA is a static EWMA with fixed weight $\theta = 0.75$. AEWMA is an adaptive EWMA with an average weight $\theta = 0.75$. The circles are the actual packet loss for each interval. The results demonstrate strong predictive power in this data trace. The 5th order AR model implies that the correlation goes back at least 10,000 packet, or 200 seconds. The AR model predicts slightly better than the two EWMA models. But simpler EWMA model competes well.

Figure 4 shows the prediction errors from the AR(5) model are unbiased. Although there is a fair amount of variance in the prediction error, the algorithm predicts well enough to indicate when the packet loss is likely to be "high".

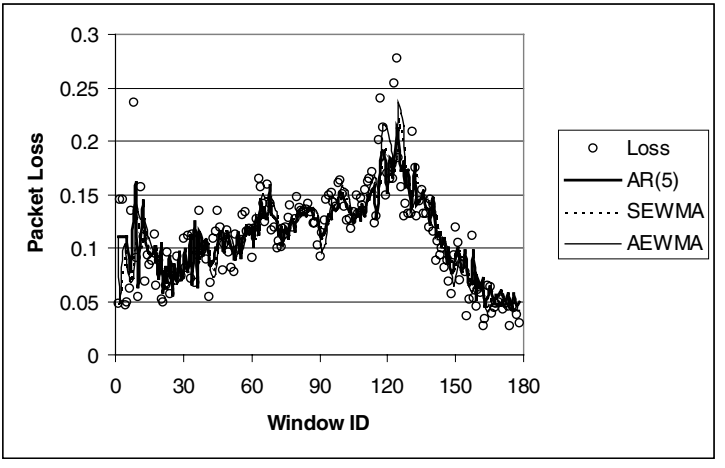

Fig. 3. Analysis of a UMASS trace using different models

3.3.2 NetMeeting Audio Trace. To further evaluate the prediction models, we conducted another experiment to generate a voice over IP trace. The experiment was set up between two desktop computers, located in New Jersey and California. A presentation was sent through NetMeeting over Lucent's Intranet. The talk was encoded by PCM (Pulse Coded Modulation) with a sampling interval of 30 *ms* into a 64 *kbps* audio stream. A commercial software tool, NetXray, was used to capture the packets transported over two end points. Another software tool, Xdecode, was used to process and analyze the captured packets trace. Xdecode reads Ethernet capture files and decodes them in an ASCII format. We further processed the decoded file using shell scripts to identify and subtract the RTP/UDP packets that carried the audio stream. We obtained the RTP packet's sequence number and relative delay offset and generated the corresponding binary packet loss trace. The trace we analyzed had 50,000 packets and lasted about 25 minutes. We divided the trace into 100 intervals of 500 packets each. The corresponding time series $\{Z_k\}$, was thus generated with the packet loss rate of each interval as the dependent variable. Again, the estimation model is window based instead of per call based.

Figure 5 shows the analytical results of the predication using three models. AR(2) is a 2nd order AR predication model with coefficients (1.12, -0.17), SEWMA is a static EWMA with weight $\theta = 0.24$. AEWMA is an adaptive EWMA with average

weight $\theta = 0.24$. The actual packet rates, denoted by the circles, show jumps between low packet loss and high packet loss. The results demonstrate strong prediction power in this data set also. This time the correlation goes back at least 1000 packets, or 30 seconds. The AR(2) model performs a little better than the EWMA models, particularly for the very clean transitions from low high loss. The EWMA models compete fairly well.

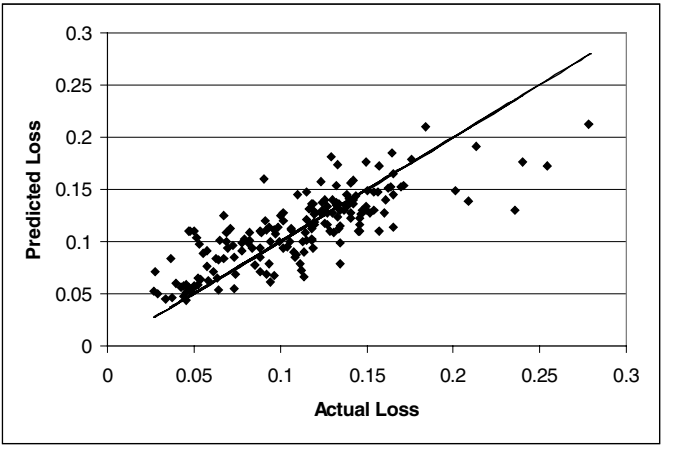

Fig. 4. Prediction error from the AR model on a UMASS trace

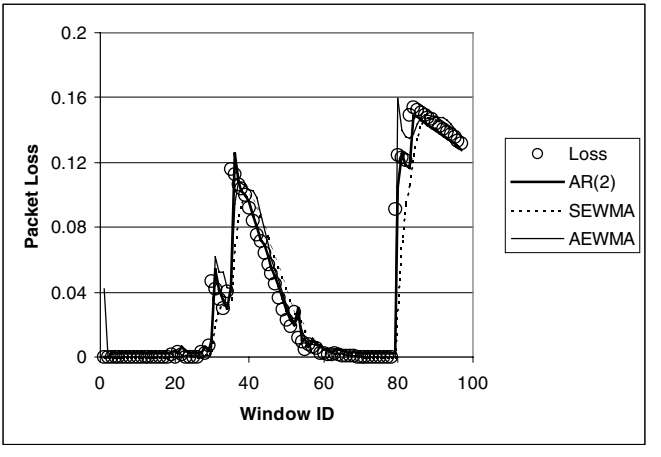

Fig. 5. Analysis of a NetMeeting trace using different models

Figure 6 shows the prediction error from the AR(2) model. The variance appears smaller than the UMASS trace. It is very likely that the predication error variance is proportional to the overall congestion level.

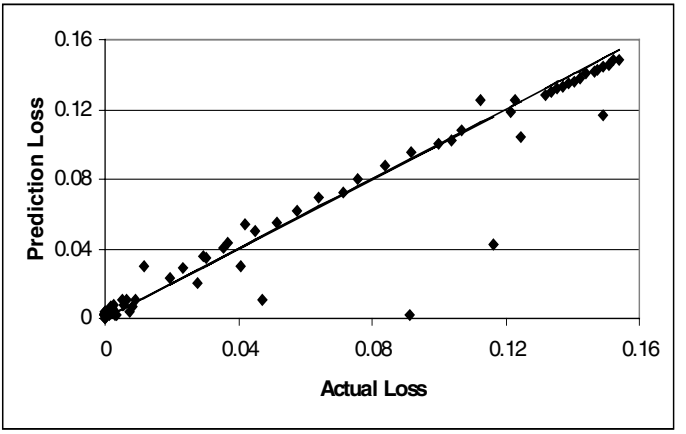

Fig. 6. Prediction error from the AR model on a NetMeeting trace

3.4 CAC Options

We now describe two variations of our proposed CAC approach, which differ in how the historical packet loss observations are calculated. The difference is significant in that the second option requires more information out of the MGWs. In Section 4 we compare the effectiveness of the two alternatives and evaluate whether or not the additional information significantly improves the overall QoS.

CAC-1: In this option, packet loss rate is defined as the percentage of packets that are lost due to network conditions such as buffer overflow, and lost or corrupt connections. The process of determining the packet loss rate for the completed calls is through EWMA using a fixed smoothing factor of 0.8. If at the time of the new call arrival, neither of the predicted one-way packet loss rates exceeds a given threshold (say, 1%), the new call is accepted, otherwise it is rejected. For the calls that are accepted, overall QoS is measured as the ratio of the completed calls for which the actual packet loss *or* jitter delay is less than a given threshold (say 2%). In order to discount the effect of packet loss in the admission process when there are long inter-arrival times between consecutive calls, the packet loss rate is discounted at a rate of say 50% if the update takes place in longer than 10 seconds. Also, in order to account for calls with very long call holding times (longer than say 100 seconds), the packet loss is determined only over the last 100 seconds of the call.

CAC-2: In this option, packet loss rate is defined as the percentage of lost packets due to all of the reasons defined for CAC-1 plus jitter variation. There is clearly an additional cost to acquire jitter measurement. However it is recognized as an important component of overall packet loss at the application layer and many vendors do support it. Note also that if network buffers are large, delay jitter may be the dominant component of packet loss. Our intent with CAC-2 is to ascertain that the use of delay jitter in the calculation of packet loss significantly improves the effectiveness of the CAC algorithm. All other details associated with CAC-2 are identical to CAC-1.

4. Performance of CAC Algorithms

A simulation model is used to compare CAC-1, CAC-2 with two other scenarios. First, a "no CAC" scenario, in which *all* calls are admitted into the network. Second, a "Static CAC" scenario, where the maximum number of calls that can be allowed on a path is predetermined from provisioned bandwidth, and the number of calls in-progress is consulted before a new call is admitted.

In the simulation model calls arrive to a single server queue, representing the network with no other interfering traffic. Each call generates voice packets every 20 ms from a PCM encoded source at 64 Kbps. Call inter-arrival times are exponentially distributed with a rate of 0.42 calls per second. The service rate of the queue corresponds to a transmission link with a capacity of 5 Mbps. The service time of a packet is therefore 1280/5000 = 0.256 ms. The call holding time is also assumed to be exponentially distributed with a mean of 180 seconds. The queue capacity is varied in the simulations. The three QoS measures are: 1) good call rate, where a good call is defined as those for which the packet loss rate over the entire duration of the call does not exceed a threshold (2% is used in the experiments), 2) call blocking rate, defined as the percentage of new call arrivals that were denied admission and, 3) packet loss, measured as the total number of packets lost due to buffer or delay jitter, and the average packet loss is determined as the percentage of packets lost within the duration of a call.

4.1 Simulation Results

Table 1 shows call blocking rate (B), good call rate (Q), and average packet loss (L) for buffer sizes of 200, 400, 600 and infinite (packets). The two entries associated with L represent the packet loss components due to buffer overflow and delay jitter, respectively. Note that since CAC-1 relies only on packet loss due to buffer overflow, the performance for the infinite buffer size case is identical to the "No CAC" case. For the simulations summarized by Table 1, play-out delay was 60 ms. In this scenario the results for Static CAC, independent of the buffer size, are B=6.2%, Q=100% and L=0%. In the case of "No CAC," all calls are allowed to the system, so obviously B=0. As a result, the quality of service as measured by Q and L are the worst compared to the other options. CAC-1 and CAC-2 both improve Q, particularly when the buffer size is small. Note that in that case, network buffer overflow dominates the overall packet loss rate so the difference between CAC-1 and CAC-2 is minimal. For large network buffers, delay jitter contributes to the overall packet loss, and CAC-2 results in better Q values compared to CAC-1, although the gains are modest for these cases. Of course CAC-2 always has higher B values since its predictions for packet loss will always be larger than those of CAC-1.

Table 2 shows the simulation results for a 20 ms play-out delay and for buffer sizes of 200 and infinity. Comparing Table 1 and Table 2 for a buffer size of 200 packets, we note that the smaller play out delay translates into difference in the performance of CAC-1 and CAC-2. In Table 2, delay jitter is a significant component of the overall packet loss rate, and thus the CAC-2 packet loss predictions will have better fidelity. As a result, both B and Q are higher for CAC-2, and L is appreciably lower. Although Q falls off precipitously from Table 1 to Table 2 for both CAC-1 and CAC-

2, Table 2 still shows that both options improve upon the "No CAC" case. Similar to the previous case, the results for Static CAC, independent of the buffer size, are B=6.2%, Q=100% and L=0%.

Table 1. Performance of CAC options (60 ms play out delay)

Scenario	Buffer Size (Packets)			
	200	400	600	Infinity
No CAC	B: 0% Q: 86.6% L: 2.2%, ~0%	B: 0% Q: 49.9% L: 2.1%, 17.5%	B: 0% Q: 49.9% L: 2.1%, 19.1%	B: 0% Q: 41.4% L: 0%, 34.1%
CAC-1	B: 5.9% Q: 95.1% L: ~0%, ~0%	B: 6.0% Q: 55.9% L: 0.5%, 10.7%	B: 5.9% Q: 53.8% L: 0.5%, 13.2%	B: 0% Q: 41.4% L: 0%, 34.1%
CAC-2	B: 5.9% Q: 95.1% L: ~0%, ~0%	B: 11.4% Q: 61.7% L: 0.2%, 6.9%	B: 12.1% Q: 62.9% L: 0.2%, 8%	B: 12.5% Q: 61.8% L: 0%, 9.8%

Table 2. Performance of CAC options (20 ms play out delay)

Scenario	Buffer Size (Packets)	
	200	Infinite
No CAC	B: 0% Q: 48.7% L: 2.2%, 19%	B: 0% Q: 43.6% L: 1.7%, 31.5%
CAC-1	B: 5.9% Q: 52% L: 0.5%, 12.5%	B: 0% Q: 43.6% L: 1.7%, 31.5%
CAC-2	B: 11.8% Q: 59.7% L: 0.2%, 6.7%	B: 13.3% Q: 60.6% L: 0%, 10.3%

In general, as expected, simulations indicate that both CAC-1 and CAC-2 increase Q compared to the "No CAC" case. What is worth noting however is that the performance of both CAC-1 and CAC-2 are approaching that of the best case "static CAC" when the network buffer and play-out buffers are sized appropriately. We have also noted that the difference between CAC-1 and CAC-2 with respect to Q is modest. However, with respect to L, CAC-2 provides appreciable reductions compared to CAC-1 and significant reductions compared to "No CAC".

5. Summary

In this paper, we presented a distributed edge-to-edge measurement-based approach to QoS support for a VoIP application. The measurements, namely packet loss rates, are available on most media gateways. We compared two statistical prediction methods: AR and EWMA, to predict packet loss from these measurements. It was demonstrated that the less complicated EWMA approach produced similar results under various traffic conditions on a public IP network.

Two measurement based CAC methods were analyzed and compared through simulations with encouraging initial results. Both methods use predicted packet loss, with the difference that the packet loss is measured on different interfaces. CAC-1 uses measurements collected on the network layer and thus only represents packet loss due to various network conditions. CAC-2 uses the measurements collected on

the application layer at the time where the packet play out is to take place. We quantified the magnitude of improvements with CAC2.

These observations help us determine the details of algorithms and identify the required interfaces to the network elements. Future work includes: (i) extension of the simulation study to include data sources and multiple MGWs, (ii) investigation of the nature of packet loss in typical network environments to better differentiate CAC-1 and CAC-2, (iii) development of a revenue model based on QoS measures to facilitate comparison of alternative CAC scenarios, and (iv) investigation of the use of other measures, such as delay, in CAC algorithms.

References

1. R.A. Lakshmi, " The Lucent Technologies Softswitch: Realizing the promise of convergence," Bell Labs Technical Journal V4, 2, APR-JUN, 1999, p 174-195
2. B. Doshi, E. Hernandez-Valencia, K. Sriram, YT Wang and O.C. Yue, "Protocols, Performance, and Controls for Voice over Wide Area Packet Networks," Bell Labs Technical Journal, Vol 3, No. 4, Oct 98.
3. K. Sriram and Y. T. Wang, "Voice over ATM using AAL2 and Bit Dropping: Performance and Call Admission Control," IEEE Journal of Selected Areas in Communications, Vol.17, No.1, 1999, pp.18-28
4. ITU-T Recommendation G.114, One-Way Transmission Time, 2/96
5. M. Grossglauser and D. Tse, "A Framework for Robust Measurement Based Admission Control," IEEE/ACM Transactions on Networking V7, 3, JUN, 1999, p293-309
6. Z. Dziong, M. Ji, and Y.T. Wang, "Learning Algorithm for CAC adjustment in ATM networks", ATM/IP workshop, June 1999
7. P. Aukia, M. Kodialam, P. Koppol, T. Lakshman, H. Sarin, and B. Suter. RATES: A server for MPLS traffic engineering," IEEE Network Magazine, pp. 34--41, March/April 2000
8. M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and Modelling of the Temporal Dependence in Packet Loss," Proceedings of Inforcom99.
9. W.S. Wei, Time Series Analysis, Addison-Wesley Publishing Company, 1990.
10. D.W. Trigg and A.G. Leach, "Exponential smoothing with adaptive response rate," Operations Research Quarterly, Vol.18, Issue 1, 1967, pp.53-59
11. D.Jeske, W.Matragi and B.Samadi, "Adaptive Play-out Algorithms for Voice Packets in a LAN Environment," International Conference on Communications (ICC), 2001.