

# Proactive Resource Provisioning

Eric Chi, Michael Fu, and Jean Walrand

Department of Electrical Engineering and Computer Sciences

University of California at Berkeley

211 Cory Hall #1772, Berkeley, CA 94720-1772, USA

Tel: 510-643-5885, Fax: 720-293-5129

echi@eecs.berkeley.edu, michaelfu@cal.berkeley.edu, wlr@eecs.berkeley.edu

## Abstract

Allocating resources in networks to QoS flows may require undesirable delays or costs. We consider a dynamic Service Level Agreement (SLA) negotiation scheme between peer autonomous systems (ASes) that implement Diff-Serv per domain behaviors. For concreteness, we tailor our scheme to account for the needs of VoIP transport across multiple ASes. We present a heuristic but computationally simple and distributed scheme that uses traffic statistics to forecast the near-future demand. Our scheme provides a statistical guarantee on the renegotiation frequency, to manage adjustment costs, and blocking probability, to manage penalties for undesirable adjustment delays. Simulations demonstrate that this scheme achieves more efficient usage of the reserved bandwidth than would be accomplished by using the *de facto* static SLA negotiation schemes

## Index Terms

Resource Provisioning, Quality of Service (QoS), Voice over IP (VoIP), MPLS, Service Level Agreement (SLA).

## I. INTRODUCTION

Allocating resources to QoS flows in networks may require undesirable delays or costs. Consider the problem of readjusting wavelength allocation in mesh optical networks employing Wavelength Division Multiplexing as load varies between two peering optical domains. While retunable optical transceivers enable optical networks to adapt to load churn, the option to retune should not be taken as an invitation to retune arbitrarily frequently. An analogous statement can be made for readjusting Label Switched Path (LSP) reservations in peering MPLS domains. In both cases, it makes sense to proactively provision enough resources to absorb a surge in an anticipated demand.

A more basic problem may be formulated from both these scenarios. Consider a two-tiered network supporting QoS flows. Connections arrive as some renewal process and hold for some i.i.d. random times. Bandwidth may be dynamically adjusted to accommodate load changes over of the network. To check overzealous readjustments, there is a cost for readjusting. Additionally there is a cost for being in the saturated reservation state. We do this to minimize the blocking probability of an incoming connection by requesting sufficient bandwidth early enough. Consider first only a single peering relationship. Note that action-space events consist not only of making a request but also the choice in how much more bandwidth to request. The required state to make an optimal decision is the number of active connections and the amount of time till a pending request is effected if one has been made. This results in an uncountable state space. Using a phase-type approximation for the pending time reduces the uncountable state space to a countable one. If we also assume Poisson arrivals and exponential holding times, the resulting optimal control formulation is a Markov Decision Process, and we may attempt to solve the Bellman optimality equation numerically using estimates of the mean arrival rate and holding time. The size of the state and action space, however, scales poorly.

Because of this “curse of dimensionality” we opt to take a heuristic approach relying on simple computations to solve the stated problem. Furthermore, we consider the problem in light of a case study of how to develop a dynamic Service Level Agreement (SLA) negotiation scheme between peer autonomous systems (ASes) that implement Differentiated Services (DiffServ) per domain behaviors. For concreteness, we tailor our scheme to account for the needs of VoIP transport across multiple ASes. We choose VoIP because not only is it a specific instance of the described problem, it is also an important application in itself and our heuristic solution provides a simple yet effective solution to the relevant problem of managing SLA adjustments when VoIP connections are carried between peering ASes.

The rest of the paper is organized as follows. In section II we describe VoIP at the level of detail relevant to the basic problem. In section III, we formulate the VoIP provisioning problem between two peering ASes. In section IV, we discuss features any good solution should meet and with these features in mind construct our scheme. In section V, we apply traditional telephony traffic models to our formulation. We present simulation results in section VI. In section VII, we discuss related work, and in section VIII we conclude with future directions.

## II. VOICE OVER IP

Developments in forwarding behavior such as DiffServ provide the forwarding-plane building blocks necessary for supporting delay-sensitive applications like VoIP [1], [2], [3]. In this paper we consider privately managed IP networks that support, the Expedited Forwarding (EF) packet treatment [4]. While a single privately managed IP network, or Autonomous system (AS), may successfully provide toll-quality voice services over a limited geography, it cannot economically provide that service globally. The more scalable approach is to provide global coverage by peering with other privately managed IP networks.

To ensure desired end-to-end forwarding treatment, each AS has a bandwidth broker [5] which negotiates with brokers of its peer ASes to determine how much of an aggregate class of traffic an AS is willing to carry on behalf of its peer ASes. Such business contracts are called Service Level Agreements.

To date, work on bandwidth brokers has focused more on implementation — signaling protocols and maintaining databases of network state [6], [7], [8]. In accordance with the DiffServ philosophy, these mechanisms are designed for admission control and resource reservation of aggregate flows of traffic. Management mechanisms are essential, but they only ensure the quality of an ongoing session — e.g. low latency — will be adequately maintained. What is often overlooked is that for some QoS applications there are additional QoS requirements which warrant a deeper look into how SLAs should be negotiated.

For VoIP maintaining a low latency connection is necessary, but it does not achieve the QoS standards set by the Public Switched Telephone Network (PSTN). When a user picks up a phone, the user expects not only that the connection request will be granted with near certainty but also that the session will be established in negligible time. In the United States, PSTN standards dictate that 50% of all connections should experience a mean call setup delay of no more than 3.6 seconds and a maximum delay of 3 seconds 95% of the time [9]. The PSTN achieves such high performance with an architecture consisting of devices finely tuned to the sole task of providing voice connections.

The peer-to-peer organization of ASes and user expectations on session setup time complicate VoIP admission control and resource provisioning across multiple ASes. The network layer in the protocol stack is aware only of packets; connection state exists only in the application layer. With such a short time frame to establish a connection across multiple ASes, it is impossible to perform call-by-call admission control and resource allocation through the application layer.

Instead of optimizing the hardware, we propose separating admission control and resource provisioning so that they occur asynchronously. Resource provisioning should be done in bulk, but it should be done in anticipation of resources needed in the near future. Admission control should be distributed to meet session initialization requirements. Then, when call requests arrive, the required resources will either be reserved or not and an immediate admission decision may be made upon a request arrival at an ingress into a confederation of peering ASes.

Since connections have in-session QoS requirements, good admission decisions require some mechanism for such a distributed admission control to determine what resources have been provisioned within the network. Ensuring in-session QoS using distributed admission control is not the focus of this paper; our focus is provisioning. Providing a distributed admission control scheme with QoS guarantees on blocking probability requires a proactive provisioning scheme like the one discussed earlier. This paper presents the design and evaluation of a distributed resource provisioning scheme engineered to complement a distributed admission control. While statistics are collected to forecast near-future bandwidth demands, no specific traffic models are inherently integral to our heuristic scheme. Nonetheless, to quantitatively evaluate the performance of our heuristic, we consider our scheme under traditional teletraffic models and estimators for computational convenience. Our primary contribution is a formulation that enables quantifiable tunability in the reprovisioning frequency and not just a specified blocking probability.

### III. VOIP PROBLEM FORMULATION

We assume that EF forwarding is achieved through priority queues [4]. Strict priority forwarding isolates voice traffic from non-voice traffic. Thus, we consider a network that carries only voice traffic. The core consists of a mesh of DiffServ ASes. We use the term DiffServ AS to refer to an AS that provides DiffServ forwarding. Customer stub networks are connected to autonomous systems that serve as ingress into the DiffServ core. Users in one stub network wish to establish voice sessions with users in another stub network. We assume that interdomain routing is fixed.

We assume that the only end-to-end requirement of voice connection is an effective bandwidth with EF forwarding and make the simplification that brokers negotiate the number of connections to carry on behalf of a neighboring AS. This assumes some way of determining or approximating the number of active calls on a link. Many current VoIP realizations follow the H.323 standards which has the option of performing PSTN like link-by-link admission control through entities called gatekeepers [10]. One possibility is to require a VoIP call to register with an AS's broker after it has been admitted into the network and de-register when it leaves. Thus, a connection does not have to make itself known to all the ASes it traverses before it can begin transmitting packets. The broker requires only maintaining the total number of active calls. The count can be incremented and decremented as calls come and go.

A deterministic amount of time  $\tau$  elapses between the time when the bandwidth broker sends its request for more bandwidth and the time when the broker is notified that its request has been activated. We assume  $\tau$  is the same for all SLA transactions.

We do not explore the question of how many connections a transit AS should choose to transport. We assume that an AS employs an admission and intradomain provisioning strategy that ensures that whatever amount of traffic an AS agrees to carry, it will manage to provide the requested service with high likelihood. It is reasonable to assume this since the commonly used over-provisioning strategy will work. Instead, as explained in the section II, we explore the complementary question of how many connections should an AS request its downstream neighbor carry on its behalf.

### IV. ADJUSTING THE SLA

For now, assume the arrival process is stationary; we address time inhomogeneity at the end of this section. We first consider two peering ASes and assume all VoIP calls are initiated in one AS and destined for the other. We consider how the client AS should adjust its SLA with its peering AS. The bi-directional case where the two ASes are transporting each other's traffic is a superposition of two instances of the previous case. The provisioning decisions over an entire network of ASes is the superposition of the provisioning decisions made in a two-peer scenario since inter-AS links are managed asynchronously.

The AS has incentive to request bandwidth in bulk from its neighbors because it is reasonable to assume some fixed cost — processing time and messaging overhead — is incurred every time bandwidth is requested and allocated. If connection requests arrive frequently enough, it makes sense to pre-provision for the future arrivals by requesting resources prior to the arrivals. To be precise, an update should occur no sooner than

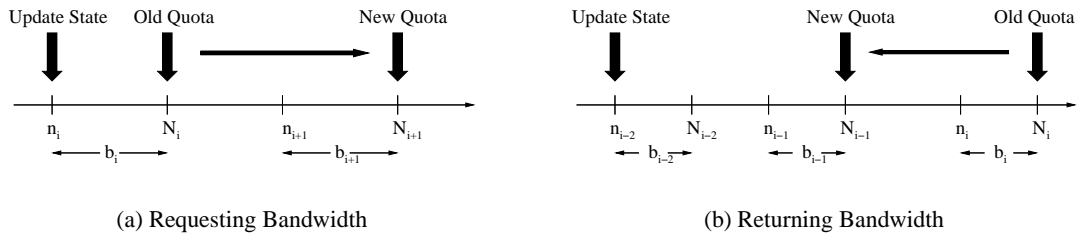


Fig. 1. Schedule

some specified time  $T$  with some specified confidence  $p$ . Thus,  $p$  and  $T$  implicitly specify the amount of processing cost to amortize over several connection admissions. On the other hand, more frequent but smaller bandwidth adjustments give better utilization.

Thus, the strategy should dynamically update a resource request schedule for each bandwidth broker based on anticipated demand. We call the observed number of active connections at which a bandwidth request is made an *update state*<sup>1</sup>. When the number of active calls equals *update state* then *new quota* calls are allocated, if that bandwidth is available. This quota assignment rule holds true whether the number of active calls reaches *update state* from below or from above. This schedule should be such that it simultaneously minimizes processing time per connection and maximizes allocated bandwidth utilization.

The schedule should also account for  $\tau$ . An AS should request more bandwidth early enough so that there is low likelihood of running out of bandwidth before its neighbors can grant more bandwidth. To be precise, the schedule should be such that the AS requests for more bandwidth when the current number of admitted connections is such that in time  $\tau$  with some specified confidence  $q$  the number of connection requests will not exceed the current number of allocated connections.

Figure 1 (a) graphically represents a request schedule. The sequence of allocated quotas of bandwidths are denoted by  $\{N_i\}$ . The  $i + 1^{\text{th}}$  quota  $N_{i+1}$  is requested when the number of admitted connections reaches the  $i^{\text{th}}$  update state which is denoted  $n_i$ . Note there is a buffer bandwidth,  $b_i$ , between the  $i^{\text{th}}$  update state and the  $i^{\text{th}}$  quota to account for the delay  $\tau$  in making an SLA adjustment.

To minimize underutilization of allocated bandwidth, we also require a strategy for releasing any unused bandwidth. If bandwidth is freed prematurely and enough new calls arrive, recently returned bandwidth will have to be re-requested. Recall we already have an update frequency defined by the parameters  $p$  and  $T$ . We choose update states at which to release bandwidth in a manner consistent with the desired update frequency.

Figure 1 (b) graphically represents the schedule for returning bandwidth. If the last connection quota is  $N_i$ , we do not decrement the quota to  $N_{i-1}$  unless the number of connections falls to  $n_{i-2}$ . Recall  $n_{i-2}$  denotes the update point to request a new quota of  $N_{i-1}$ . The next update state is  $n_{i-1}$  and the quota requested for that state is  $N_i$ . Recall  $n_{i-1}$  was chosen so that starting from  $n_{i-2}$  we will reach  $n_{i-1}$  no sooner than in time  $T$ , with confidence  $p$ . Thus, if the number of connections drops to  $n_{i-2}$  but begins to increase again, we will not request for bandwidth that was returned no sooner than in time  $T$  with confidence  $p$ . We refer to update states that correspond to an increase in quota to be *upgrade states* and update states that correspond to a decrease in quota to be *downgrade states*.

To account for time inhomogeneity in the call arrival process we maintain a “history stack” of all downgrade states and corresponding downgrade quotas to use if the call arrival process were to stop. Note we do not keep the entire history. In the example stack shown in Figure 2 the current number of active calls is between 21 and 39. If the number of active calls reaches the current upgrade state of 40 calls, then a new upgrade state and corresponding quota is computed based on the current arrival statistics. This pair is pushed onto the top of the stack and the rest of the state is appropriately re-designated. If the number of active connections drops to the current downgrade state, the current upgrade state and corresponding upgrade quota are popped off of the stack, and the remaining state is re-designated. Thus in moving down the “history stack” the schedule re-applies previously computed update states and quotas. New information is incorporated in

<sup>1</sup>We use the word “state” for convenience; the true state of the system is not completely specified by the number of active connections.

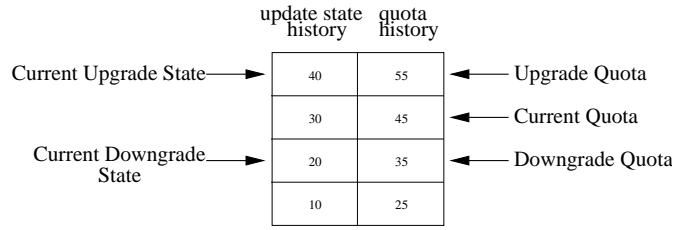


Fig. 2. History Stack

computing the next upgrade state and quota for only upgrade events. The five pointers shown in Figure 2 always maintain the same relative positions with respect to each other. The current upgrade state and quota always point to the entry at the top of the stack.

We decided to reuse previously computed update/quota pairs for downgrades based off of empirically measured daily demand profile of voice traffic which tends to be low in the morning; peak mid-day; and then decline thereafter. Using previous computations for downgrades yields a more conservative approach in returning bandwidth since the true arrival rate at the time of a downgrade is likely to be less than the arrival rate used to compute the subsequent update state after the downgrade. Upgrade events, on the other hand, probably indicate the arrival rate is increasing, and so the next update state should be computed based on the most current estimate of the arrival rate.

## V. A MODEL FOR CALL SOURCES

Denote the number of active connections at time  $t$  as  $X_t$ . We model the connection arrival process as a Poisson arrival process with rate  $\lambda$ . We assume that although  $\lambda$  may vary with time that it does not do so very rapidly and proceed to compute the schedule assuming that  $\lambda$  is fixed.

We wish to determine how much bandwidth to request to ensure a specified blocking probability and specified update frequency probability. We do not focus on blocks due to saturated downstream link capacity. That is a network dimensioning concern. We wish to minimize avoidable blocks — blocks that occur due to poorly scheduled update requests.

We model holding times as i.i.d. exponential random variables with rate  $\mu$ . Thus,  $X_t$  evolves as an  $M/M/\infty$  queue. We assume both  $\lambda$  and  $\mu$  vary slowly enough that they may be estimated accurately in deriving the schedule. In our simulations we use the ML estimator over small time intervals.

We continue with the simple case where the core DiffServ network consists of two peering ASes. We wish to calculate a schedule for the ingress AS to request and return bandwidth from its downstream neighbor given the traffic parameters  $\mu$  and  $\lambda$  and the given parameters  $T$ ,  $\tau$ ,  $p$ , and  $q$  as defined in Section IV.

We recursively calculate the update states  $\{n_i\}$  with  $n_0 = 0$ . Denote the first passage time to state  $m$  as  $T_m$ .

$$T_m = \inf\{t > 0 : X_t = m\}. \quad (1)$$

We want  $n_k$  to be the smallest state such that when we start at the last update state  $n_{k-1}$ ,  $T_{n_k}$  is greater than  $T$  with confidence  $p$ . For notational convenience let  $P_k(A) = P(A|X_0 = k)$ .

Then

$$n_k = \min\{m > n_{k-1} : P_{n_{k-1}}(T_m > T) \geq p\}. \quad (2)$$

We take into consideration that updating an SLA takes non-negligible time  $\tau$ . When we start in state  $X_0 = n_k$  we seek the smallest state  $n_k + b_k$  such that  $T_{n_k + b_k}$  is greater than  $\tau$ , with confidence  $q$ .

$$b_k = \min\{m > 0 : P_{n_k}(T_{m+n_k} > \tau) \geq q\}. \quad (3)$$

The schedule of update states is given by  $\{n_i\}$ . The schedule of bandwidth quotas is given by  $\{N_i = n_i + b_i\}$ .

Using the reflection principle when  $p$  and  $q$  are close to unity, we approximate the hitting time probability with the transition probabilities of an  $M/M/\infty$  queue [11].

We again recursively calculate the update states  $\{n_i\}$  with  $n_0 = 0$ , using transition probabilities. We want  $n_k$  to be the smallest state such that when we start at the last update state  $n_{k-1}$ , with confidence  $p$ ,  $X_T$  is strictly less than  $n_k$ .

$$n_k = \min\{m : P(X_T < m \mid X_0 = n_{k-1}) \geq p\}. \quad (4)$$

We choose the buffer bandwidth  $b_k$  for the  $k^{\text{th}}$  update state to be the smallest state such that when we start at the update state  $n_k$ , with confidence  $q$ ,  $X_\tau$  is strictly less than  $n_k + b_k$ .

$$b_k = \min\{m : P(X_\tau < m + n_k \mid X_0 = n_k) \geq q\}. \quad (5)$$

To perform the above computations, we require the transition probabilities of an  $M/M/\infty$  queue. We seek to compute the transition probability  $p_{ij}(t)$ :

$$p_{ij}(t) = P(X_{t'+t} = j \mid X_{t'} = i). \quad (6)$$

Given the number of active calls  $X_{t'}$  at time  $t'$ , we can express the number of active calls at time  $t + t'$  as the sum of two independent random variables,  $N(t, t')$  and  $Y(t, t')$ .  $N(t, t')$  denotes the number of the calls out of the  $X_{t'}$  that are still active after the time  $t$ , and  $Y(t, t')$  denotes the number of calls that arrived between the time of  $t'$  and  $t + t'$  that are still active at time  $t' + t$ .  $N(t, t')$  is distributed binomially with parameters  $X_{t'}$  and  $e^{-\mu t}$ .  $Y(t, t')$  is Poisson with rate  $\lambda/\mu(1 - e^{-\mu t})$ .

To simplify calculations we approximate both the Binomial and Poisson components with Gaussian random variables. Furthermore, we take  $\tau$  to be orders of magnitude smaller than  $\mu^{-1}$ , thus we keep only the first order terms of the Taylor expansion of the terms  $e^{-\mu\tau}$ . Applying these simplifications we obtain the following recursive update schedule.

$$\begin{aligned} \mu_Y &= \frac{\lambda}{\mu}(1 - e^{-\mu T}) \\ \mu_{n_k} &= n_k(1 - \mu\tau) \\ \sigma_{n_k}^2 &= n_k\mu\tau(1 - \mu\tau) \\ n_{k+1} &= \lceil \mu_Y + \mu_{n_k} + Q^{-1}(1 - p)(\mu_Y + \sigma_{n_k}^2)^{\frac{1}{2}} \rceil \\ b_k &= \lceil \lambda\tau - n_k\mu\tau + Q^{-1}(1 - q)(\lambda\tau + n_k\mu\tau)^{\frac{1}{2}} \rceil. \end{aligned} \quad (7)$$

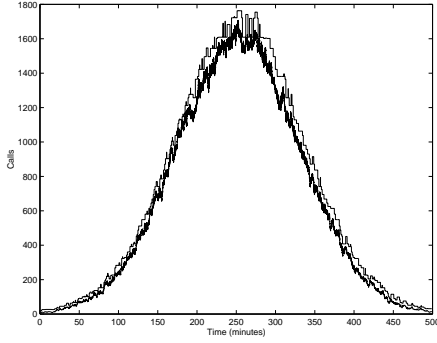
As stated earlier, it is a simplification to assume that the arrival process is stationary. From an estimated arrival rate an upgrade point could be computed adaptively as the estimated arrival rate changes in time as described in Section IV. We shall refer to the scheduling algorithm as proactive resource provisioning (PRP) for the remainder of the paper.

In some networks, setup delay is not a significant concern. For instance blocking probability is not a typical performance metric in optical networks. In those cases where only the tradeoff between adjustment frequency and utilization is important only the update states  $n_k$  need to be computed.

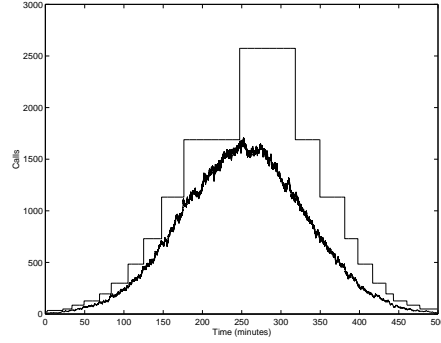
## VI. SIMULATION

We describe the simulation scenarios used to evaluate the performance of PRP. Call durations are modeled as i.i.d. exponential random variables with a mean of 2 minutes. The call arrival process for each source-destination stub pair was modeled as a Poisson arrival process with a right continuous step intensity function,  $\lambda(t)$ . We present three sets of simulations.

The first set considers provisioning a single OC-3 link between two peer ASes which each serve one stub network. All calls are initiated in one stub and are destined for the other. The arrival rate increases and then decreases in a manner that mimics empirically measured arrival rates. The update frequency and blocking rate statistics are examined, and the time average utilization is compared to that which would have been observed if provisioning is done statically applying the Erlang B formula to the busiest hour.



(a) Single Link:  $T = 1$  minute



(b) Single Link:  $T = 5$  minutes

Fig. 3. Single Link

TABLE I

SINGLE LINK: UPDATE AND UTILIZATION STATISTICS

$T$ (min.)	$\hat{P}(T_{obs} > T)$	$\hat{E}T_{obs}$ (min.)	$\hat{U}$	$P\hat{B}P$	$\hat{B}P$
1	0.997	1.74	0.908	0.048	$5.6 \times 10^{-5}$
5	0.996	22.8	0.654	0.011	$4.9 \times 10^{-5}$
10	0.996	25.5	0.633	0.012	$5.5 \times 10^{-5}$
$\infty$	-	-	0.194	-	0.01

The second set applies PRP to a network of six ASes and five links each consisting of two OC-3 links. A naive distributed admission control strategy is used in conjunction with the provisioning provided by PRP. The purpose of this simulation set is to determine how much of the average utilization gained in using PRP over static dimensioning is lost when global knowledge of the link provisioning is unavailable at the admission points into the core network.

In the first two sets we find that PRP achieves higher link utilization than static schemes while maintaining prescribed stochastic bounds on update frequencies.

In the third set we consider arrival rates that change abruptly to quantify PRP's blocking rate performance to sudden surges in traffic along a single OC-3 link. The purpose of this set is to quantify PRP's sensitivity to estimation error.

For all simulations we took  $\tau$  as defined in Section IV to be 1 minute. Parameters  $p$  and  $q$  were taken to be 0.05 and 0.01 respectively in all simulations except for the multi-link simulation where  $p$  was taken to be 0.01. The arrival rate is estimated with the MLE over a sliding five-minute window. The mean holding time is assumed to be known.

1) *Single Link*: Figure 3(a) and 3(b) depict typical realizations of PRP provisioning for traffic for a prescribed threshold update period  $T$  of 1 minute and 5 minutes respectively. The daily variability used here is consistent with measurements of the number of calls placed at Stanford University's telephone exchange during every hour of a work day averaged over a period of six months in 1995 [12]. To the eye PRP successfully adjusts resource allocation to match demand. The same trial was repeated 1000 times for different  $T$ . The relevant statistics are shown in Table I.

The random variable  $T_{obs}$  is the time interval between quota update request events.  $\hat{P}(T_{obs} > T)$  and  $\hat{E}T_{obs}$  are the empirical tail distribution and sample mean of  $T_{obs}$  respectively.  $\hat{U}$  is the time average utilization of the provisioned resources  $P\hat{B}P$  is the sample mean of maximum of observed ratios of blocked calls to arrived calls within a  $\tau$  time interval over a single simulation run, and  $\hat{B}P$  is the sample mean of the observed ratio of blocked calls to arrived calls over an entire simulation run.

The last row in Table I, corresponding to  $T = \infty$ , was calculated analytically. Using the Erlang B

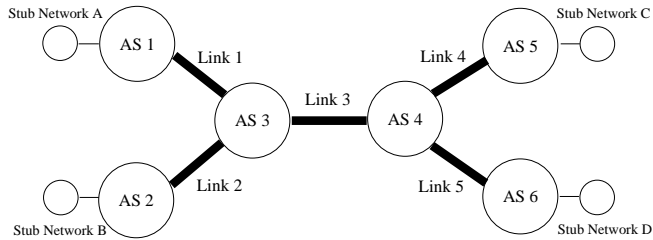


Fig. 4. Multiple Links: Topology

formula, we calculated the provisioning required to achieve a blocking probability of 0.01, which equals our prescribed parameter  $q$  for the simulations, for the peak arrival rate prescribed over the 500 simulated minutes. Note all prescribed inequalities are satisfied while the average utilization of provisioned resources compares favorably to the average utilization achieved through static worst-hour provisioning.

Not surprisingly, decreasing the value of  $T$  increases bandwidth efficiency. The sharp drop in average utilization shown in Table I when  $T$  is increased beyond 1 minute can be explained by PRP's conservative return bandwidth policy. In Figure 3(b) at about 250 minutes into the simulation the bandwidth quota is increased perhaps a bit too proactively, and PRP waits over an hour for the demand to drop significantly prior to returning the bandwidth.

2) *Multiple links*: PRP specifies only a resource provisioning strategy. An admission control scheme is not prescribed here but is necessary to complement PRP. Recall distributed admission control at the edges of the DiffServ core expedites admission decisions. Here we demonstrate PRP's effectiveness in a multi-link network where spare capacity of all inter-AS links are not known at all times to admission control points at the edge of the DiffServ core. Each AS applies PRP provisioning on each of its outgoing links in the network shown in Figure 4.

Each AS broadcasts to all its neighbors its spare capacity to a given stub network. An AS receiving the broadcast from an upstream AS knowing its own spare capacity on the inter-AS link from which it received the broadcast takes the minimum of that spare capacity and the spare capacity in the broadcast message. It then broadcasts this minimum to all inter-AS edges other than the one it received the broadcast. This is reminiscent of distance vector routing where nodes announce their distances to a given sink but not the routes. Broadcasts are made both periodically with a specified inter-broadcast time and whenever an update in quota occurs on any link in the network. We acknowledge this scheme fails to scale, but we require some distributed admission control scheme to evaluate PRP in the network. More scalable distributed admission control schemes, such as the ones described in [13] and [14] which use Explicit Congestion Notification, do exist and may be used with PRP.

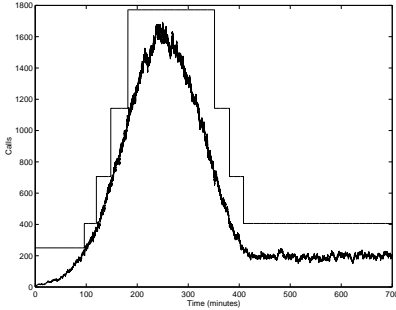
Broadcasts occur every 5 minutes.  $T$  is taken to be 10 minutes. Calls arrive for two different source-destination pairs — stub A to stub C and stub B to stub D. The arrival rate for stub A to stub C increases to a peak rate at 250 minutes and then decreases symmetrically before settling down to a constant rate of 100 calls per minute. The arrival rate for stub B to stub C is the reflection about the 350 minute mark of the arrival rate for stub A to stub C calls. Note that the second goal stated at the end of Section III may be violated when using a distributed admission control policy. Here the broadcast frequency was chosen high enough so that at no time did the number of active calls exceed the allocated quota on any link for all 1000

TABLE II  
MULTIPLE LINK NETWORK: UPDATE AND UTILIZATION STATISTICS

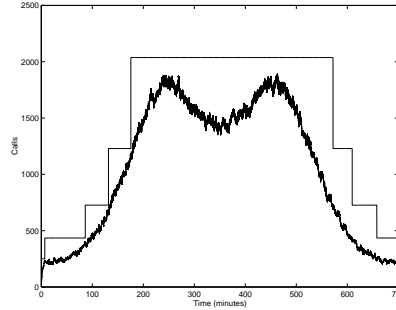
Link	$\hat{P}(T_{obs} > 10 \text{ min.})$	$\hat{E}T_{obs}$ (min.)	$\hat{U}$	$\hat{U}_{static}$
1	1.0	57.4	0.622	0.161
2	0.99	84.6	0.585	0.162
3	0.95	75.0	0.617	0.287

TABLE III  
MULTIPLE LINK NETWORK: BLOCKING RATE STATISTICS

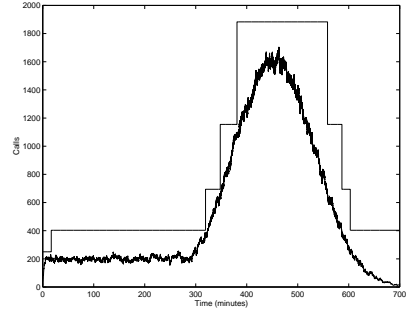
Source/Dest. Stub pairs	$P\hat{B}P$	$\hat{B}P$
$AC$	0.006	$5.0 \times 10^{-5}$
$BD$	0.004	$3.2 \times 10^{-5}$
$\infty$	-	0.01



(a) Link 1 and Link 4



(b) Link 3



(c) Link 2 and Link 5

Fig. 5. Multiple Inter-AS Links

runs.

Update and utilization statistics are shown in the first three columns of Table II; blocking rate statistics are shown in Table III. Again the specified stochastic bounds are met. The  $\hat{U}_{static}$  in the last column of Table II is the average utilization achieved by statically provisioning for the worse case arrival rate with the prescribed 0.01 probability of call blocking.

Figure 5(a), 5(b), and 5(c) depict a sample realization of provisioning for each of the links. We see that PRP still manages to meet performance specifications and maintain competitive utilization when a simple yet aggressive distributed admission control scheme is used.

While this multi-link network may unrealistic, it still provides a useful evaluation of the performance of PRP. This simple multi-link network demonstrates that good performance and utilization is still very possible when provisioning is not explicitly coordinated between ASes or any centralized admission control. By dynamically resizing peering links, PRP creates an opportunity for good network management; intelligent admission control consummates it. We acknowledge that if a better distributed admission control algorithm were used, more complex scenarios could be simulated and a more complete evaluation of PRP can be made. While PRP runs asynchronously with admission control, performance is clearly very dependent on the admission control chosen. Nonetheless, our work presumes adequate admission control exists and focuses on how to provision. We present a scenario where that hypothesis is met and show that PRP enhances the performance.

3) *Sensitivity to time variation in arrival rate:* The last simulations explore model-based PRP's sensitivity to estimation error. Two ASes each serve a stub network peer. The two ASes are connected by a single OC-3 link. All calls are initiated in one stub network and are destined for the other. We simulate the onset of a busy cycle on this link. The initial arrival rate is 500 calls per minute. The arrival rate is then increased linearly over a five minute window. The resulting blocking rates over this window are shown in Table IV. Recall that the prescribed blocking probability is 0.01. While PRP is not so fragile that it cannot handle an unexpected increase in the arrival rate, it is clear, nonetheless, that PRP eventually fails to meet prescribed blocking probabilities with a sufficiently large surge in the arrival rate. Nevertheless, we maintain PRP is still a significant improvement over more static schemes.

TABLE IV  
SINGLE LINK: TIME VARYING ARRIVAL RATE

Ending arrival rate	Peak Blocking Rate
600	$1.7 \times 10^{-4}$
650	0.029
700	0.063
750	0.104

## VII. RELATED WORK

We are not the first to attempt to bilaterally concatenate SLAs and dynamically adjust their allocations [15], [16], [17], [18], [19], [20]. In these works, however, the decision of when and how much bandwidth to request does not take into account any QoS requirements. PRP is motivated in part by the observation that the blocking probability of connections carried by these SLAs should influence how these SLAs should be adjusted. On the other hand there are schemes that take into consideration application-level blocks but do not have statistical guarantees on the renegotiation frequency [21].

With regards to allocation schemes that account for application-level blocks, PRP is similar to the scheme presented in [22]. That scheme, however, makes adjustments at regular time intervals; it does not allow for a dynamically adaptive update frequency. By relaxing a deterministic update frequency constraint to a statistical one, we allow occasional violations in a bound on the update frequency to accommodate atypical surges in demand while still limiting excessively frequent updates in the long run.

The idea of sizing and timing bandwidth adjustments to account for blocking probabilities, however, can be readily found in many dynamic allocation schemes in ATM literature [23], [24], [25], [26]. In retrospect PRP is an adaptation of these ATM ideas to an IP framework where control is far less coordinated. PRP answers same questions about how to adjust bandwidth but in a best effort manner – update frequencies are guaranteed only statistically and allow occasional violations in a target frequency. Peers adjust their SLAs asynchronously. We do not rely on ATM-like stringent control, but as the simulation results suggests more than adequate dynamic resource provisioning is still achievable.

## VIII. CONCLUSION

Our key contributions are: 1) provisioning resources to meet a statistical guarantee on inter-peer reservation renegotiation frequency, in addition to connection blocking probabilities; 2) demonstrating that our peering pair scheme running asynchronously on all peering pairs in a multi-peer network efficiently uses reserved resources.

We emphasize that PRP and distributed admission control complement and do not take the place of the other. PRP as it is presented here does not announce what resources have been provisioned. It is up to the admission control scheme to determine what resources have been provisioned within the network by PRP. At the same time, distributed admission control schemes at best only ensure in-session QoS requirements such as low latency but have no control over call blocking probability. The latter QoS requirement can be met only by judicious SLA provisioning.

Additionally, PRP enables any given peer to make provisioning adjustments independently of the states and actions of all other peers. The downside of having each peering pair renegotiate asynchronously is that no end-to-end guarantees can be made on end-to-end resource availability. In this sense PRP provides best-effort end-to-end resource allocation. The simulation results, however, suggest that stringent coordination of provisioning of peering links is not necessary. Indeed, despite a lack of end-to-end coordination among peers, increases and decreases along particular end-to-end paths are correlated since reservation adjustments are made based on measured load levels. Although each peering pair makes provisioning adjustments asynchronously with respect to other pairs, end-to-end like provisioning is roughly met.

Furthermore, these adjustments do not require flooding the network with control messages. Allocations occur independently and moreover asynchronously without need for peers to broadcast allocation decisions to each other. By design, changes in traffic between peers initiate an adjustment only along that peering link which limits the communication overhead making PRP a scalable provisioning strategy. Together with a good distributed admission control scheme, PRP requires marginal overhead within the core of the network.

Given how encouraging the simulation studies in multi-peer networks were, developing models that predict the multi-peer simulation results is a natural next step that we plan to pursue. We feel, however, that the most interesting future direction is to make PRP robust against modeling error. PRP could be fashioned to observe the objectives of interest — update frequency, utilization, and blocking rate — and make adjustments that move the observed objectives in the desired direction. This approach presents new challenges. While utilization and readjustment frequency may be monitored locally at a peer, blocking probability is only observable at the edges of a network where admission decisions are made. Adapting PRP to make decisions locally at an AS based on information from the edges is critical to making PRP practically applicable to all delay-sensitive applications with session initialization requirements.

#### ACKNOWLEDGMENT

The authors thank Dragan Petrović for helpful discussions in the early development of this work as well as reviewers for their constructive comments.

#### REFERENCES

- [1] Blake, S.; D. Black; M. Carlson; E. Davies; Z. Wang; W. Weiss. 1998. "An Architecture for Differentiated Services." RFC 2475, IETF, Dec.
- [2] Brim, S.; B. Carpenter, F. Le Faucheur. 2000. "Per Hop Behavior Identification Codes." RFC 2836, IETF, May.
- [3] Nichols, K.; S. Blake; F. Baker; D. Black. 1998. "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers." RFC 2474, IETF, Dec.
- [4] Jacobson, V., K. Nichols; K. Poduri. 1999. "An Expedited Forwarding PHB." RFC 2598, IETF, Jun.
- [5] Nichols, K.; V. Jacobson; L. Zhang. 1997. "A Two-bit Differentiated Services Architecture for the Internet." Internet Draft, IETF, Nov.
- [6] Engel, T.; H. Granzer; B.F. Koch; M. Winter; P. Sampatakos; I.S. Venieris; H. Hussmann; F. Ricciato; S. Salsano. 2003. "AQUILA: Adaptive Resource Control for QoS Using an IP-based Layered Architecture." *IEEE Communications Magazine*. Vol. 41, No. 1, Jan:46-53.
- [7] Cortese, G.; R. Fiutem; P. Cremonese; S. D'antonio; M. Esposito; S.P. Romano; A. Diaconescu. 2003. "CADENUS: Creation and Deployment of End-User Services in Premium IP Networks." *IEEE Communications Magazine*. Vol. 41, No. 1, Jan:54-60.
- [8] Flegkas, P.; P. Trimintzios; G. Pavlou. 2002 "A Policy-Based Quality of Service Management System for IP DiffServ Networks." *IEEE Network*. Vol. 16, No. 2, March-April:50-6.
- [9] Lin, H.; T. Seth; A. Broscius; C. Huitema. 1999. "VoIP Signaling Performance Requirements and Expectations." Internet Draft, IETF, Jun.
- [10] Open H323 Project: <http://www.openh323.org>
- [11] Chi, E. Proactive Resource Provisioning for VoIP. UC Berkeley, Department of EECS M.S. Thesis (Dec. 2001).
- [12] Lam, D.; J. Jannink; D.C. Cox; J. Widom. 1996. "Modeling Location Management in Personal Communications Services." In *Proceedings of the IEEE 5th International Conference on Universal Personal Communication*, (Cambridge, MA, Sep.).IEEE, New York, NY, 596-601.
- [13] Tung, T. and J. Walrand. 2003. "Providing QoS for Real-time Applications." In *Proceedings of the 2nd IASTED International Conference on COMMUNICATIONS, INTERNET, & INFORMATION TECHNOLOGY* (Scottsdale, AZ, Nov.) ACTA Press, Anaheim/Calgary/Zurich, 462-468.
- [14] Kelly, F.P.; P.B. Key; S. Zachary. 2000. "Distributed Admission Control." *IEEE Journal on Selected Areas in Communications*, Vol. 18., No. 12, Dec:2617-28.
- [15] Chuah, C.; L. Subramanian; R. Katz; A. Joseph. 2000. "QoS Provisioning Using a Clearing House Architecture." In *Proceedings of IEEE/IFIP Eighth International Workshop on Quality of Service* (Pittsburgh, PA, Jun.).IEEE,Piscataway, NJ, 115-24.
- [16] Kamienski C. and D. Sadok. 2001. "Chameleon: An Architecture for Advanced End-to-End Services." In *Proceedings of the Second IEEE Latin American Network Operations and Management Symposium* (Belo Horizonte - MG, Brazil, Aug.).
- [17] Günter M. and T. Braun. 1999. "Evaluation of bandwidth Broker Signaling." In *Proceedings of IEEE Seventh International Conference on Network Protocols*. (Toronto, Canada, Oct.).IEEE Computer Society,Los Alamitos, CA, 145-52.
- [18] Xiao, X. and L.M. Ni. 1999. "Internet QoS: A Big Picture." *IEEE Network* Vol. 13, No.2. March-April:8-18.
- [19] Reichmeyer, F.; L. Ong; A. Terzis; L. Zhang; R. Yavatkar. 1998. "A two-tier resource management model for Differentiated Services networks." Internet Draft, IETF, Nov.
- [20] Dermier, G.; M. Günter; T. Braun; B. Stiller. 2000. "Towards a Scalable System for Per-flow Charging in the Internet." In *Proceedings of the Applied Telecommunication Symposium* (Washington D.C., Apr. 17-19.),SCS, San Diego, CA, 182-7.
- [21] Whitt, W. 1999. "Dynamic Staffing in a Telephone Call Center Aiming to Immediately Answer All Calls." *Operations Research Letters* No. 24, No. 5, June:205-212.
- [22] Hampshire, R.; W. Massey; D. Mitra; Q. Wang. 2002. "Provisioning of Bandwidth Sharing and Exchange." In *Telecommunications Network Design and Economics and Management: Selected Proceedings of the 6th INFORMS Telecommunications Conferences* (Boca Raton, FL, Mar. 10-13). Kluwer Academic Publishers, Boston/Dordrecht/London, 207-226.

- [23] Friesen V.J., Harms J.J., Wong J.W., "Resource Management with Virtual Paths in ATM networks", IEEE Network, vol 10 no 5, September/October 1996.
- [24] Z. Dziong, Y. Xiong, L.G. Mason, "Virtual Network Concept and its applications for resource management in ATM based networks", International IFIP/IEEE Conference on Broadband Communications, Chapman & Hall, 1996.
- [25] Luo Z., Bigham J., Cuthbert L.G., Hayzelden A.L.G., "Traffic Control and Resource Management using a Multi-Agent System", 5th International Conference on Broadband Communications, Hong Kong (China), November 1999.
- [26] Sven-Olof Larsson, Ake Arvidsson, "An Adaptive Local Method for VPC Capacity Management", ITC 16, Key and Smith (Eds.), Elsevier Science B.V., 1999.