Edge-Based QoS Provisioning for Point-to-Set Assured Services

Satish Raghunath, Kartikeya Chandrayana, Shivkumar Kalyanaraman
Department of ECSE, Rensselaer Polytechnic Institute
rsatish, kartikc, shivkuma@networks.ecse.rpi.edu

Abstract

We propose an edge-based architecture for point-to-set services aimed at site-to-site private networks over the Internet. It extends the traditional point-to-point service model to a point-to-set service model, assuming a finite, bounded set of destination sites. An ideal point-to-set service allows a user to have a pool of premium tokens, which could be flexibly assigned to traffic going towards any destination within the set. The architecture concentrates all data-plane and control-plane activity at ISP network edges, coordinated by a management or policy component. The attractiveness of an edge-based point-to-set service is to provide statistical assurances and flexibility to users while allowing providers to get multiplexing gains. The service parameters for a simple point-to-set assured service is developed; and its flexibility (compared to a point-to-point service) and efficiency (multiplexing gain) characteristics are examined. We define a metric to characterize the degree of flexibility allowed by a particular point-to-set assured service. We also propose a dynamic provisioning capability at the edge, to efficiently realize the point-to-set service while minimally trading off service contract parameters. The dynamic provisioning entity tracks the user's bandwidth usage and then changes the provisioning as per the user's traffic profile and current traffic demands. We present simulations to illustrate the basic characteristics of this model.

1 Introduction

The best-effort (non-QoS) traffic in Internet is inherently of the *point-to-anywhere* nature, i.e., sources direct packets to any possible destination. In contrast, traditional quality-of-service (QoS) models set up premium services on a *point-to-point* basis (eg: virtual leased lines[3], frame-relay, ATM services, int-serv [6] etc). Recently, with the advent of IP differentiated services [5, 8, 7] there has been interest in expanding the *spatial granularity* of QoS models. Clark and Feng [8] proposed that a pool of "assured" service tokens could be allocated to a user or site with the flexibility to mark packets sent to any arbitrary destination with such tokens. While such a *"point-to-anywhere"* assured service model is very appealing to users, the large spatial granularity of the service makes efficient admission control and provisioning virtually impossible [7].

This paper develops the building blocks for *"point-to-set"* assured services where the *"set" of destinations is finite and known a priori* (Figure 1). These conditions could be met in a site-to-site private networking (eg VPN) scenario. This service is provisioned over a single provider (or

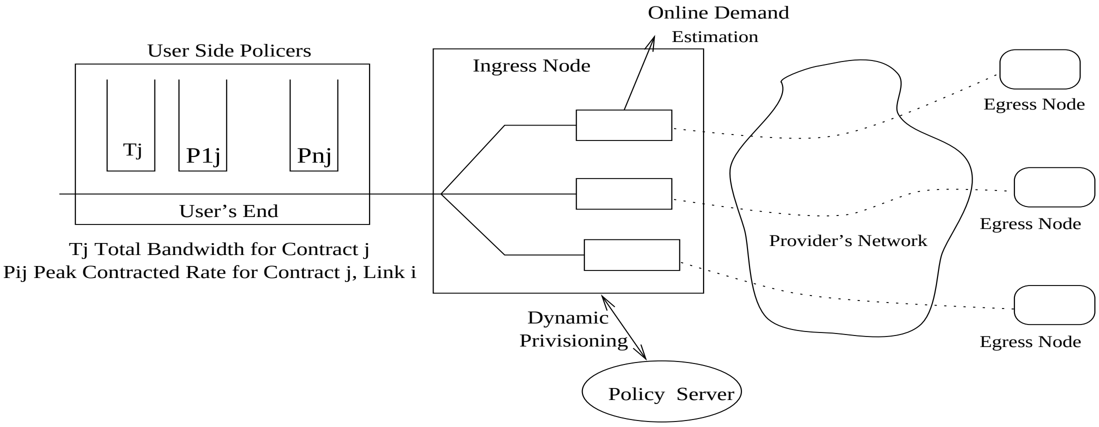

Figure 1: The Point-to-Set Architectural Model

cooperating multi-provider) differentiated services IP inter-network. Further, the paper proposes a model where the service is *completely provisioned and managed at the network edges* using a policy server (or network management) component. For efficiency (i.e. multiplexing gains), we develop *online per-path demand estimation and dynamic provisioning capabilities* which operate on short time-scales and affect provider-side shapers. Point-to-set contracts are admitted using a *simple measurement-based admission control* scheme. These point-to-set contracts are weaker than traditional QoS contracts in several respects. First, they cannot offer strict statistical guarantees and only offer assurances. Second, the system may trigger involuntary peak rate re-negotiation on specific paths on rare occasions when it is unable to dynamically provision for the aggregate demand (eg: due to a concerted and persistent change in traffic patterns). In spite of these limitations, we believe that our point-to-set model is a first step in realizing Clark et al's [8] goal of going beyond point-to-point services and providing flexibility to users, while at the same time realizing multiplexing gains for providers.

LIRA [7] is closest to our work in that it considers the problem of large spatial granularity. But it provisions service in terms of resource tokens, rather than absolute bandwidth. The number of resource tokens charged to each in-profile packet is a dynamic function of the path it traverses and the congestion level along the path. In this work we do not measure congestion levels of paths, though that is a future goal [1]. Also we choose to dynamically measure demand and provision bandwidth instead of using resource tokens. Duffield et al [9] also propose some simple QoS models for VPNs where traffic managed on a per-VPN granularity or per-class granularity (CoS). Edge-based management is also an option suggested, but no algorithms are presented. Our initial model considered in this paper differs from the simple assured service model [8] in the following respects: a) we assume a simple FIFO internal network, b) we assume the availability of a network map with bandwidths at a policy server (possible through OSPF extensions [2]), and c) we do not admit any out-of-profile traffic into the internetwork (i.e. all admitted traffic in the network is in-profile and traffic waits at the edge till it obtains tokens).

The rest of the paper is organised as follows. Section 2 outlines the point-to-set architectural model. Section 3 summarizes the building blocks of the model. The building blocks are detailed in Section 3.3 (online demand estimation) Section 3.4 (dynamic provisioning) and Section 3.5 (admission control). In Section 4 we present the simulation results and compare our model with point-to-point allocation resource model. Section 5 summarizes and concludes the paper.

2 Point-to-Set Service Model

The *point-to-set service* model (Figure 1) allows a site j to claim a pool of premium tokens T_j per unit time and flexibly use it for a set of destinations {i}, subject to the limit that on each path ij the *peak assured rate* is p_{ij}. Observe that the user has the flexibility to use upto the peak assured rate p_{ij} on any single path, but pays the provider only based upon the aggregate pool of premium tokens T_j per unit time. This is more economical for the user if $T_j < \sum_i p_{ij}$ because unlike point-to-point leased lines or frame-relay CIR[1] it need not pay for unused committed rates. The essential data plane component is a traffic conditioner which now consists of a set of token buckets: one for the overall pool of tokens with rate T_j and one for each per-path peak assured rate limits p_{ij} (Figure 1). We assume that bucket depths are minimal for smoothness. We assume a network of FIFO queues, which means that no out-of-profile traffic is admitted. Further, by edge-based provisioning, we ensure that all bottlenecks at the ingress edges and not in the internetwork (including potential peering points or international links).

From the provider perspective, however, this scheme does not yield multiplexing gains compared to peak-rate provisioning *unless a dynamic measurement and re-provisioning strategy is used.* That is, in the absence of dynamic re-provisioning, the provider will have to have per-path policers (at the ingress edge block in Figure 1) operating at the peak assured rates p_{ij}. To achieve multiplexing gains, we develop the following components. First, we develop an online demand estimation module which monitors the varying per-path traffic demands (and includes adjustments for queue backlogs at the provider-side per-path policer). It accounts for potential long-range dependence in traffic by exploiting the concept of *correlation horizons* [13], i.e., the existence of an estimation time-scale over which second-order statistics hold. In particular, the module detects the current correlation horizon dynamically and uses the second-order statistics of demand estimated on that horizon.

Second, these per-path demand estimates d_{ij} are used in a dynamic provisioning module. This module also involves the policy server in Figure 1 and operates in two phases: a) per-path provisioning phase and b) per-contract provisioning phase. Finally it enforces per-path rate limits z_{ij} at the provider-side policers for each path ij such that $\sum_i z_{ij}$ is close to T_j. This means that the

[1]CIR = Committed Information Rate parameter of frame-relay service

provider can attain multiplexing gains because $T_j << \sum_i p_{ij} \Rightarrow z_{ij} \leq p_{ij}$. However, the potential random shift of traffic from one path to another implies that each dynamic provisioning decision or admission control decision is subject to error. The queue backlogs due to errors in a provisioning period are accounted in the next period. The rare case of consistent queuing due to errors in admission control and persistent shifts in traffic is dealt with by forced renegotiation of peak assured rates p_{ij} only on the affected paths i. Buffer overflows during such phases lead to packet loss, i.e., degradation of assured service which is a cost of attaining multiplexing gains.

3 Components of the Point-to-Set Architecture

The architecture contains the following components: contract definition, data plane components (user-side and provider-side shapers), online demand estimation component, per-path dynamic provisioning (at the policy server), per-contract dynamic provisioning (at the ingress edge) and an admission control component. These topics are elaborated in the following sub-sections. The specific blocks proposed in this paper could be improved or extended in future work.

3.1 Contract Definition

The contract for user j is defined as:

- The finite, known set of destinations S_j with cardinality N_j.
- The pooled total premium token rate, T_j, which could be used towards any destination $i \in S_j$. If t_{ij} is the traffic generated towards destination $i \in S_j$, then $\sum_i t_{ij} \leq T_j$.
- The per-path peak assured rate, p_{ij} on path ij.
- Duration of the contract. Not explicitly leveraged in this paper.

In this paper, we also use a concept of minimum per-path assured service expectation which states that the user will get a minimum provisioning of $\min T_j/N, p_{ij}$ on any path at any time. This constraint is enforced by admission control. Beyond that, the provisioning may depend upon the demand estimation and multiplexing characteristics of the path. Errors in provisioning or admission control in this regime manifest as delays (on which there is no assurance), packet losses, and/or involuntary downward renegotiation of peak rates on specific paths. One can imagine a regime without such a minimum service expectation, which could lead to higher multiplexing gains, but also lead to higher multiplexing costs due to provisioning/admission control errors.

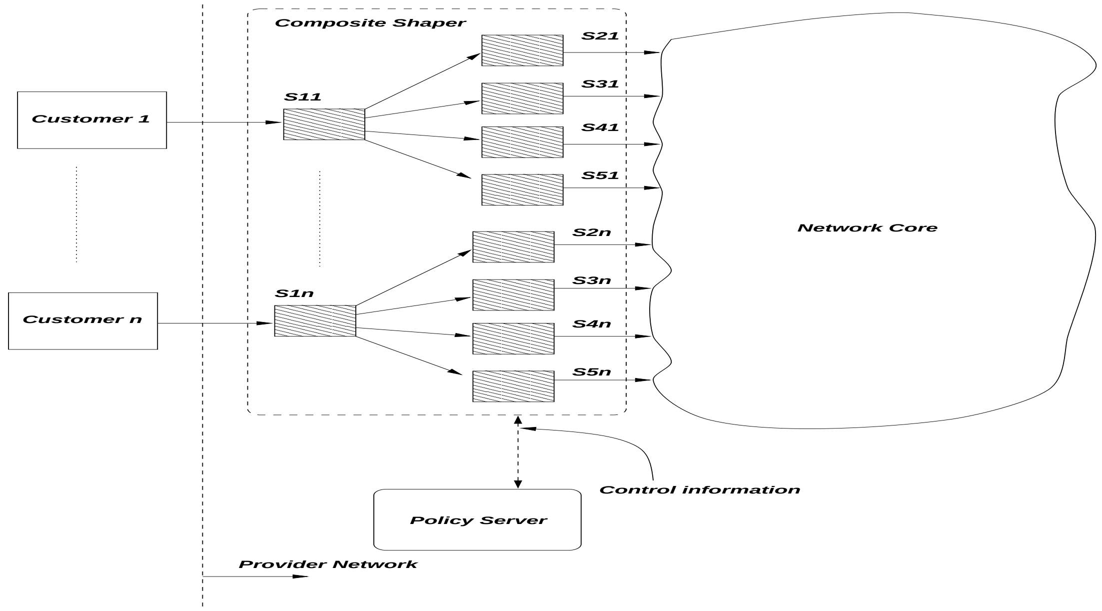

Figure 2: Composite Shaper

3.2 Traffic Conditioners: An Integrated Implementation

Figure 1 had illustrated a simple data-plane model with user-side and provider-side policers/shapers. On each side, the model needs one aggregate shaper per contract and also a shaper per destination in the set specified in the contract. Hence the total number of shapers for contract with user j is $N_j + 1$. Instead of multiple shapers handling this need, a single entity, a *composite shaper*, can be envisioned to achive the same function. Figure 2 illustrates this concept.

3.3 Online Demand Estimation

The online demand estimation component interfaces with the composite shaper discussed in the previous section to compute demand estimates d_{ij} for each path ij. This demand estimation has to also be conducted carefully. Experimental evidence [14] suggests that network traffic exhibits properties of self-similarity and long range dependence (LRD). In order to track demand patterns it is essential to address issues due to long-range dependence.

The immediate question to be answered is that of the amount of correlation information and the appropriate timescale to be considered for demand estimation. One would expect that traditional p-Markov models would not be sufficient in modelling such traffic. However, recent work [13] analysing finite buffer queues with LRD input indicates that only a finite amount of correlation need be considered for certain queueing performance measures (e.g., loss rate). The amount of correlation that needs to be taken into account depends not only on the correlation structure of

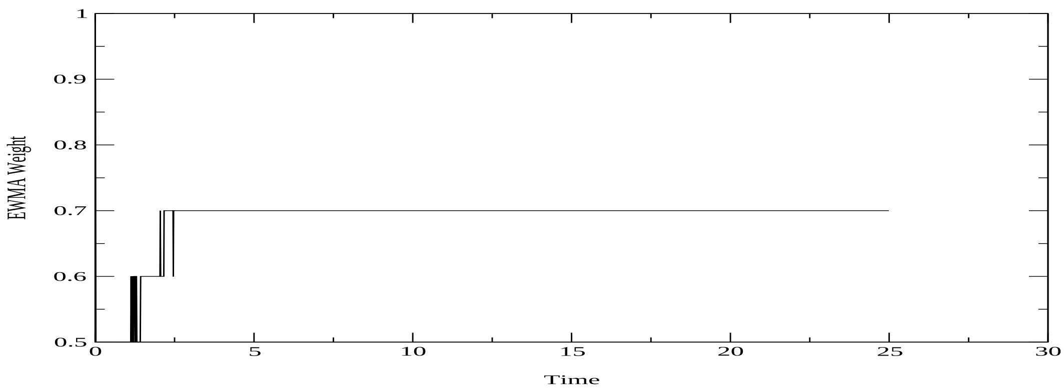

Figure 3: EWMA weight α vs Time

the source traffic, but also on time scales specific to the system under study. However, the paper establishes the existence of a correlation horizon at any point in time. In particular, it is shown that the impact of correlation in the arrival process on loss rates becomes negligible beyond a time-scale, referred to as the *correlation horizon.* This implies that we may choose either self-similar models or models with finite memory as long as it captures correlation upto the correlation horizon for the system.

We therefore choose a simple Exponentially weighted moving average (EWMA) predictor. The weight parameter of the EWMA predictor is changed dynamically to suit the traffic characteristics depending upon the current correlation horizon. This missing piece is to detect the correlation horizon. The demand estimation scheme hence maintains a *vector of EWMA predictors* for different weight parameters ($\alpha = [0.001, 0.9]$, though the pseudo-code below implies a smaller range). At the end of a measurement window, the weight parameter *corresponding to the minimum deviation from the observed sample for the previous interval* is chosen. The minimum deviation here implies that the corresponding weight parameter reflects underlying correlation horizon of the traffic. The details of the algorithm can be found in the pseudo code. The predictor then provides the average (μ) and variance (σ^2) of the tracked process. The estimate, e, is calculated as

$$e = \mu + 2\sigma$$

.

Figure 3 shows a typical graph for the variation of the weight parameter with time, for a specific simulation run. The weight parameter initially varies, but later on stabilizes around a single value. The estimates produced by the predictor are plotted against the actual samples, in Figure 4. The estimates can be seen to be effectively tracking the samples. The final estimate of demand is referred to as d_{ij} in subsequent sections.

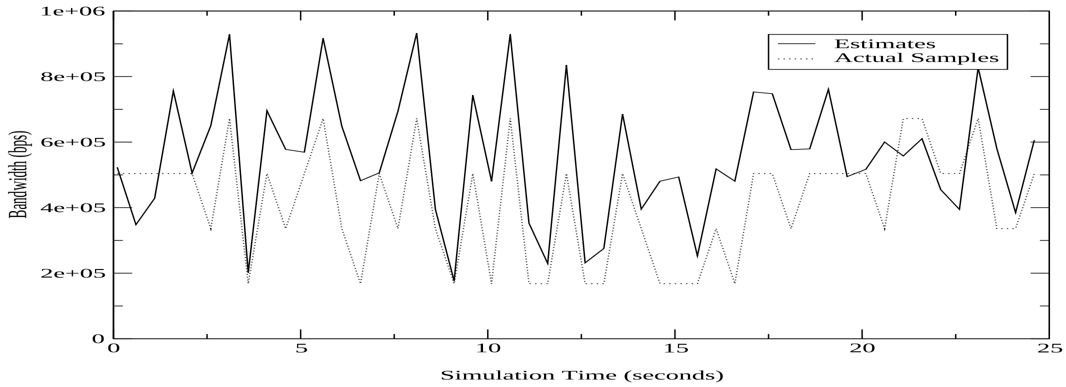

Figure 4: Online Estimates of Traffic vs Time

Algorithm 1 Adaptive calculation of $EWMA$ weight parameter

$mindeviation = MAXNUM$;
$index = 0$;
for $weight = 0.05$ to 0.99 **do**
  {Update the mean deviation using latest sample in arrivals}
  {meanarrivals was used for prediction }
  $meandev[index] = meandev[index] + abs(meanarrivals[index] - arrivals)$;
  {To find the weight with minimum deviation}
  if $meandev[index] < mindeviation$ **then**
    $mindeviation = meandev[index]$;
    $optimalwt = index$;
  end if
  {Update the moving average for arrivals}
  $meanarrivals[index] *= (1 - weight)$;
  $meanarrivals[index] += weight * arrivals$;
  {For calculation of deviation}
  $sqrmeanarrivals[index] *= (1 - weight)$;

  $sqrmeanarrivals[index] += weight * arrivals * arrivals$

  $index++$;
  $weight += 0.05$;
end for
{Using $optimalwt$ calculate prediction}
$std\_deviation = sqrt(sqrmeanarrivals[optimalwt] - (meanarrivals[optimalwt])^2)$;

$prediction = meanarrivals[optimalwt] + 2 * std\_deviation$

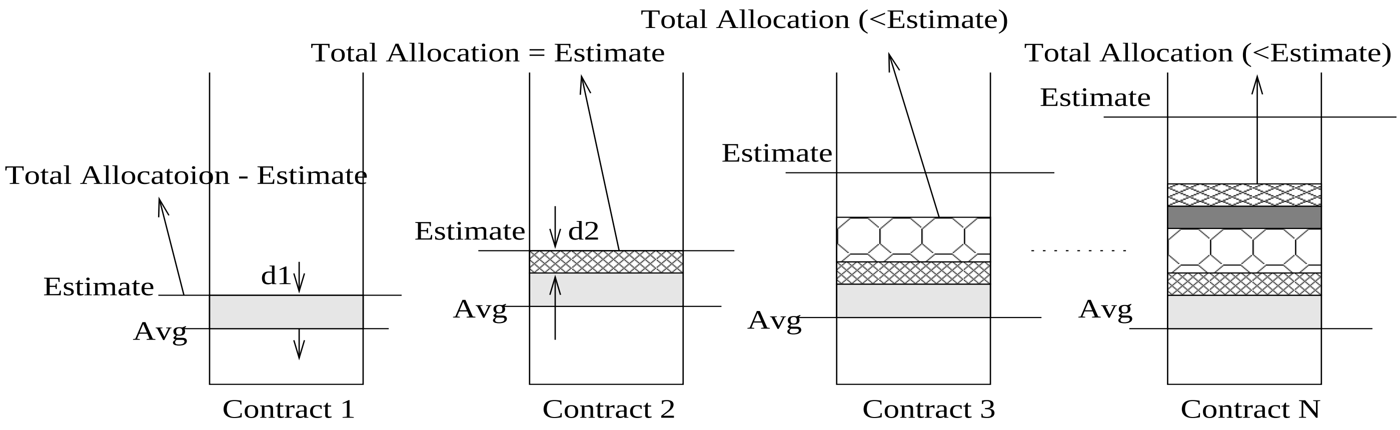

Figure 5: Per-Path Reprovisioning Policy

3.4 Per-Path Dynamic Provisioning

As mentioned earlier the dynamic provisioning decision is made in two phases, and uses the demand estimate d_{ij} to finally yield the rate limits for the shapers z_{ij}. The first phase is the *per-path* provisioning which involves a policy server component. The per-path decision is made with the knowledge of demand estimates for competing contracts on every link in the edge-to-edge path. The decision is then returned to the ingress edge which uses it in the second phase (per-contract provisioning) before finally enforcing the rate-limits.

The policy server is a centralized decision making authority in the point-to-set architecture for admission control and the per-path dynamic provisioning decision (both of which requires network topology information). The policy server is assumed to have a map of the network including link bandwidths (available through OSPF extensions [2]) and a picture of the current set of contracts and demand estimates (d_{ij}). We assume that the network topology remains constant over the timescales of provisioning. The choice of provisioning time-scales is important since frequent updation of the shapers will lead to large control traffic to the policy server. We assume that online measurement intervals (see previous section) are much more fine-grained than provisioning time-scales. The latter is set based upon the available buffer sizes at the edges and known limits on maximum traffic to the policy server. Observe that the policy server interfaces only to the ingress edges, and not to the interior routers or egress edges. For scalability a distributed provisioning architecture like [15] may be considered.

The per-path provisioning proceeds as follows:

- If the sum of per-path provisioning estimates (over all contracts) on a link ($\sum y_{ij}$) is less than the link capacity C, then allocate estimate y_{ij} to each shaper. Otherwise:
- Minimize the given equation, according to constraints stated. Let us define a_j, p_j, d_{ij}, and y_{ij}

as the average rate and peak rate for the Contract j, resource demanded (estimate) for the present timescale, and the final allocation made for Contract j on Link i respectively. Also define that there are N contracts on the link. Then the problem can be stated as:

$$Minimize \sum_{j=1}^{N}(d_{ij} - y_{ij}) \ \ subject \ to \tag{1}$$

$$\sum_{j=1}^{N} y_{ij} \ \leq \ C, \tag{2}$$

$$y_{ij} \ \leq \ d_{ij}, \forall j \tag{3}$$

$$\sum_{j=1}^{N} a_j \ < \ C, \tag{4}$$

$$y_{ij} \ \geq \ 0, \forall j \tag{5}$$

The solution to the above minimization is suggested in Algorithm 2. Assume that the contracts are maintained, in a list, in the ascending order of the difference of estimate and average, i.e., $(d_{ij} - a_j)$.

Figure 5 describes the per-path provisioning policy. As stated, we first allocate resources to all the contracts whose estimate is less than their average and eliminate them from the provisioning list. Figure(5) shows all the remaining contracts. Here we first allocate the corresponding average avg to each contract. Then, the minimum difference between the estimate and the difference is calculated $d1$ and is allocate to each contract, and the available bandwidth on the link is updated. With this allocation, we find that the first contract, Contract 1, demands has been met and subsequently it is removed from the list. Again, the new minimum difference, $d2$ between the estimate and the allocated (till now) is calculated and dispersed amongst all the remaining contracts, eliminating yet another contract from the list. This process is repeated recursively until, we run out of the available bandwidth on the link. This is similar to a max-min algorithm.

3.4.1 Per-Contract Reprovisioning

At the end of the Per-Path Reprovisioning, the allocations conveyed by the Policy Server to the agents on the Ingress edges are y_{ij}. This means that the per-path allocation is y_{ij} to contract j on path i. However, the total suggested allocations for the contract may exceed his total contracted bandwidth T_j. So the agent at the edge needs to do another level of reprovisioning amongst different paths in a contract to obey this SLA and also to conform to the suggested allocations by the Policy Server. This reprovisioning can be again expressed as an optimization problem. Let z_{ij} be the final

Algorithm 2 Per-Path Provisioning Decision

For all $d_{ij} \leq a_j$ allocate d_{ij}. Delete these contract from the List. Calculate available resource A, as the difference of allocations made and the link Capacity C.

For the rest of the contract allocate a_j, where a_j is the average rate for the Contract j. Define $x_{ij} = a_j$ and recalculate A.

while $A > 0$ **do**

  Define *diff* = Minimum($d_{ij} - x_{ij}$) over all j, M as the number of contracts for which provisioning decision has to be made.

  if $\mathit{diff} \geq A/M$ **then**

    Define $\mathit{increment} = A/M$

    Allocate resources equal to *increment* to each contract.

    $A = 0$

  else

    $\mathit{increment} = \mathit{diff}$

    Allocate resources equal to *increment* to each contract.

    Update x_{ij} as x_{ij} += *increment*

    A -= *diff* * M

    Delete the contracts for which $(d_{ij} - x_{ij})$=0 from the List (of contracts for which provisioning has to be made).

    Update M.

  end if

end while

allocation made (by agent) to Contract j on Link i and N be the number of destinations in the Contract j's set. Then the problem is stated as:

$$Minimize \sum_{i=1}^{N}(y_{ij} - z_{ij}) \quad subject\ to \tag{6}$$

$$\sum_{i=1}^{N} z_{ij} \leq T_j, \tag{7}$$

$$z_{ij} \leq y_{ij}, \forall\, i \tag{8}$$

$$z_{ij} \geq 0, \forall\, i \tag{9}$$

The solution to the above equations can again be computed using the algorithm 2 (i.e. a max-min allocation), albiet taking care of the new variables and the constraints.

3.5 Measurement-Based Admission Control

Since the policy server has information about network's current usage and also the aggregate information about the network, it is best equipped to make the admission control decision. The admission control policy is defined as follows. When the user passes his requirement to the policy server, calculate the most constrained (by resources) path. Here a path is defined as the set of link, starting with the edge's link, to a destination in the Set. The path which includes the link (at any edge) with the minimum available resources (bandwidth) is defined as the most constrainted path. The admission decision is based on the most constrained link, i of total capacity C with N contracts currently using that path. Let the new contract needs a Total resource of T_{new} with a peak p_{new} and average $a_{new} = T_{new}/n$ where n is the number of destinations in the contract.

Define a measure flexibility, f_{ik} which characterizes the allowed variation (by the point-to-set service) of traffic profile to the Contract k on Link i. Let C_i be the total capacity on a link and the user's destination set consists of links $i = 1 \ldots n$. Assume at the begining of the provisioning timescale, the estimate for the Contract j on the Link i is d_{ij} and N contracts use the link i and their usage is y_{ij}. The unused capacity U_i on the link i can then be calculated as

$$U_i = C_i - \sum_{j=1, j\neq k}^{N} y_{ij} \tag{10}$$

Then, we define the flexibilty f_{ik} as:

$$f_{ik} = 1\, if\, \frac{U_i}{y_{ik}} \geq 1\, and\, \frac{y_{ik}}{p_j} \leq 1 \tag{11}$$

$$= \frac{U_i}{y_{ik}},\, if\, \frac{U_i}{y_{ik}} \leq 1 \tag{12}$$

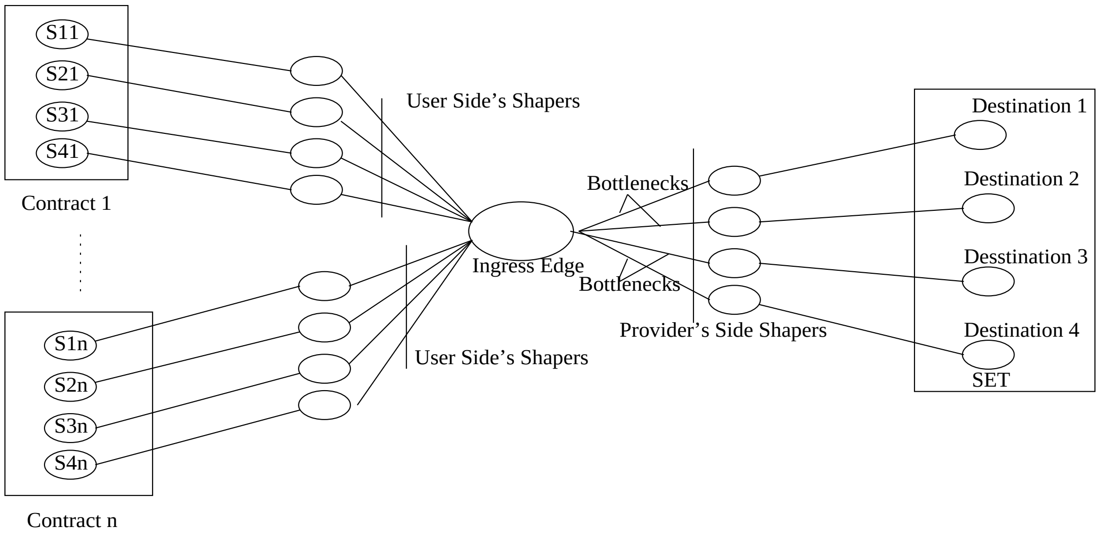

Figure 6: Topology used for Simulation

The total flexibility for the contract (per timescale), F_j is then defined as the geometric mean of the per-link flexibility of all the edge links in the contract's set.

The Policy Server maintains the flexibility information for each contract (j) on each link (i) as an EWMA of the contract's per-link flexibility f_{ij}. This metric is called f_j, as the contract flexibility. Given all this information, the admission policy is stated as:

If a_j, f_{ij} are the average rate and the flexibility of a Contract j on the constrained path i, then admit a new contract if,

$$\sum_{i=1}^{N} a_i + B/n \ \leq \beta\, C \ , where\, \beta \leq 1 \; and \tag{13}$$

$$(f_1 * f_2 * \ldots * f_N)^{1/N} \ \geq \ F_{min} \tag{14}$$

where F_{min} is the minimum flexibility needed on any link. This parameter F_{min}, is defined by the provider to maintain a specified grade of service.

4 Results

The topology shown in Figure 6 was simulated on NS [12]. All the contracts have an equal total contracted bandwidth, T, of 2 Mbps, a peak rate of 0.75 Mbps and have a common set, of 4 destinations. The constrained links are of 25 Mbps and the one way end-to-end propagation delay is 30ms. All the links other than the constrained links are of 10 Mbps. The sources are shaped by static shapers to ensure conformity with the contract. The provider network features the dynamically provisioned shapers which adapt to the user's traffic distribution with respect to his destinations.

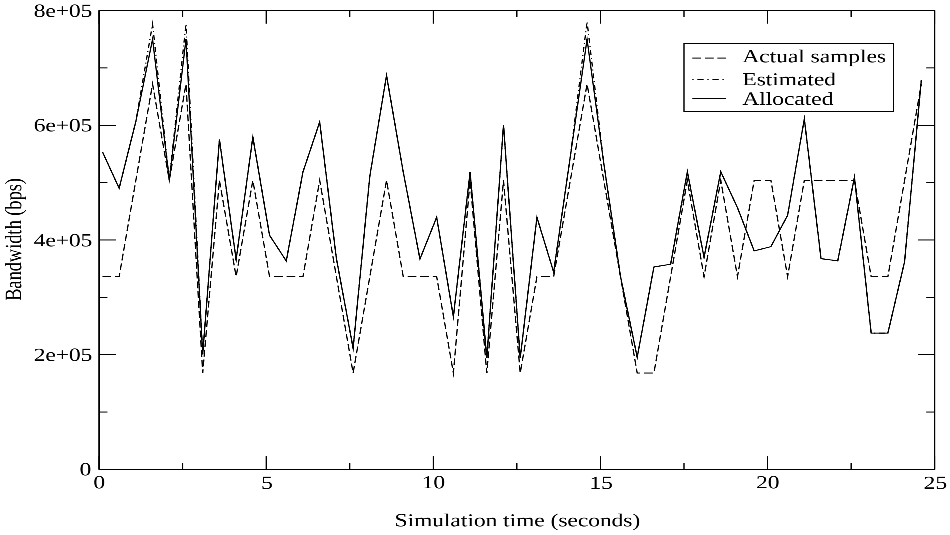

Figure 7: Allocated, Estimated bandwidth and Actual Traffic Samples Vs Time

The point-to-set model was tested with 1 to 50 contracts. The sources are NS UDP sources with CBR agents attached, but randomly changing rates. At fixed time intervals, the rates of the CBR agents were reassigned with randomly generated numbers. The sources generate traffic such that it is distributed uniformly with respect to the destinations. That is, a particular rate being seen towards any destination is an equally likely event. This is a generic traffic profile, which does not assume any bias in the way a contract chooses to send its traffic towards its destination set. It is a simple extension to define specific profiles for test purposes. The following sections explain the results.

4.1 Performance of the provisioning scheme

Section 3.4 we presented a strategy to handle per path bandwidth provisioning. The perfomance of this strategy decides the costs to the customer. Figure 7 shows the plots for the actual allocation made for a contract along with the traffic samples and estimates versus time. The Allocation curve is almost coincident with the Estimate curve. This is indicative of the fact that most of the time the contract was allocated what it demanded. Since the customer's traffic is loosely patrolled, his peak rate at some time instants exceed the negotiated peak, i.e. 0.75MBps and in such cases the allocation is clipped to the contract's peak. In effect the plot shows that the contract requirements are met.

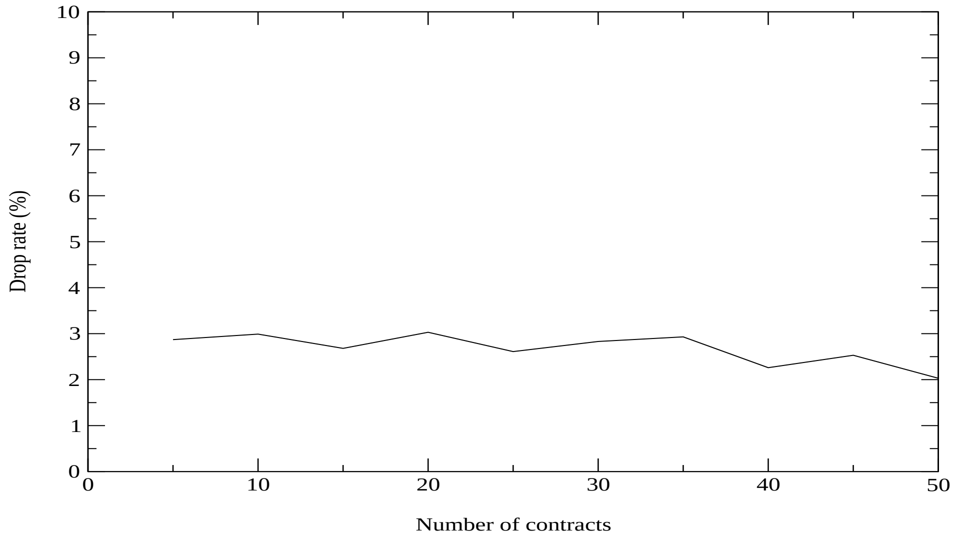

Figure 8: Drop Rate versus Number of Contracts

4.2 Multiplexing Gains

The proposed model exploits the fact that customer does not offer peak load towards every destination. As against a point-to-point model, where the per-path peak would have been reserved on each path, this model only needs the contracted per path average to be set aside. This means, the sum of peaks of contracts on a link can add up to a number greater than the link capacity. The advantage of this is that the provider can offer a cheaper and more flexible service. If the user buys a total bandwidth (towards the set of destinations) that is smaller than the sum of peaks (towards these destinations) he stands to gain. This would typically satisfy the needs of a customer whose average usage is low as compared to the peak usage. The trade off here is the statistical nature of the assurances. Unlike the point-to-point model, there is a cost in terms of dropped packets. Here we consider the cost of multiplexing contracts in the simulation scenario described above in terms of drop rates. The Figure 8 illustrates the drop rate for a contract as the total number contracts increase. The drop rates were obtained as averages over multiple runs of simulations for the particular number of contracts. The plot shows that the drop rates remain approximately the same. Thus the provider and customer avail of multiplexing gains with an approximately fixed drop rate as the cost. The reason for the drops is partly due to the traffic profile of other contracts and also due to the limitations of the estimation module.

5 Summary and Conclusions

This paper developed the building blocks for an efficient implementation of point-to-set assured service capabilities. The contributions include the development of a simple architecture, contract model and dynamic demand estimation, dynamic provisioning and simple admission control techniques. Future work will focus on making the framework robust and to quantify the errors in admission control, provisioning etc. Also integration with edge-to-edge congestion control building blocks [1] is contemplated.

References

[1] D. Harrison, S. Kalyanaraman, S. Ramakrishnan, "Overlay Bandwidth Services: Basic Framework and an Edge-to-Edge Closed-Loop Building Block," preprint. http://www.ecse.rpi.edu/Homepages/shivkuma/OverlayQoS.ps

[2] G.Apostolopoulos, R. Guerin, S. Kamat, A. Orda, T. Przygienda, D. Williams, "QoS Routing Mechanisms and OSPF Extensions,"*RFC 2676,* August 1999

[3] K. Nichols, V. Jacobson, L. Zhang, " An approach to service allocation in the Internet," *Internet Draft,* November 1997.

[4] D. Clark,J Wroclawski, " An approach to service allocation in the Internet", *Internet Draft,* November 1997.

[5] S Blake etal, "An Architecture for Differentiated Services", *IETF RFC 2475,* December 1998.

[6] R. Braden, D. Clark, Scott Shenker, "Integrated Services in the Internet Architecture: An Overview", *IETF RFC 1663,* June 1994.

[7] Ion Stoica, Hui Zhang, " LIRA: An Approach for Service Differentiation in the Internet",*Proceedings of NOSSDAV, Cambridge,* England, pp. 115-128d, July 1998.

[8] D. Clark, W. Fang, "Explicit Allocation of Best-Effort Packet Delivery Service", *IEEE/ACM Trans. on Networking,* 6 (4), pp. 362-373, August 1998.

[9] N. Duffield, etal, "A Performance Oriented Services Interface for Virtual Private Networks", *IETF Draft,* November 1998.

[10] Floyd, S., and Jacobson, V. "Random Early Detection gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking,* Vol.1 No.4, p. 397-413, August 1993.

[11] G. Iannaccone, M. May, C. Diot. "Aggregate Traffic performance with Active Queue Management and Drop from Tail", *Sprint Technical Report TR01-ATL-012501,* Sprint ATL, January 2001.

[12] UCB/LBLN/VINT "Network Simulator - ns (version 2)", http://www.isi.edu/nsnam/ns, 1997.

[13] Matthias Grossglauser, Jean-Chrysostome Bolot. "On the Relevance of Long-Range Dependence in Network Traffic," *SIGCOMM'96,* August 1996.

[14] Will E Leland, Murad S Taqqu, Walter Willinger and Daniel Wilson. "On the self-similar nature of Ethernet traffic (Extended version)," *IEEE/ACM Trans. on Networking*, 2(1):1-15, February 1994

[15] Chuah C.N., Subramanian L., Katz R.H., and Joseph A.D., "Qos provisioning using a clearing house architecture," *IwQoS'00.*