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

Satish Raghunath, Kartikeya Chandrayana, Shivkumar Kalyanaraman
Department of ECSE, Rensselaer Polytechnic Institute, Troy, NY 12180

Abstract— **In this paper we propose an edge-based quality of service (QoS) architecture aimed at site-to-site private networks over the Internet. We extend the traditional point-to-point service model to a point-to-set service model, assuming a finite, bounded set of destination sites. A 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 proposed point-to-set service provides statistical assurances and flexibility to users while allowing providers to obtain multiplexing gains. To realize the point-to-set service model, we introduce edge-based dynamic bandwidth tracking and provisioning schemes. The tracking algorithm predicts demand towards a given destination edge. This information is used to efficiently allocate bandwidth towards the destinations in the set. Simulation results are presented to demonstrate the merits of the proposed architecture in terms of cost savings to the customer and efficient resource utilization to the provider.**

I. 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 QoS models set up premium services on a *point-to-point* basis (eg: virtual leased lines, frame-relay, ATM services, int-serv [5] etc). Recently, with the advent of IP differentiated services [4], [7], [6] there has been interest in expanding the *spatial granularity* of QoS models. Clark and Feng [7] 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 [6]. We consider the subset of this problem by examining assurances to a fixed set of destinations.

Consider a private network of sites A, B, C and D as shown in figure 1. The aggregate traffic from A (called the *point*) is bounded by the "outgoing pipe" purchased from the provider. Site A's peak traffic to any node in the private network, i.e. B, C or D (called the *set*) is bounded by the capacity of the outgoing pipe, "peak". Given the point-to-point allocation model, the site A would require a link with capacity equal to "peak", to each destination in the *set* for an assured service towards the sites in the set. As such, the total purchased capacity from the provider (which is three times the peak here) exceeds "outgoing pipe" leading to wastage of resources. We propose a *point-to-set service* wherein a customer buys a bandwidth *less than or equal* to his "outgoing pipe" (or a given total bandwidth), but is *statistically* assured that his traffic needs to any destination in the set are met (however, note that no apriori assurances are derived in this paper). In other words, the customer buys bandwidth to a set of destinations, instead of purchasing point-to-point links to the destinations. Thus there is a cost saving in that the point-to-point

Supported in part by NSF contract ANI9806660, NSF contract ANI9819112 and grants from Intel Corp. and Nortel Networks Limited

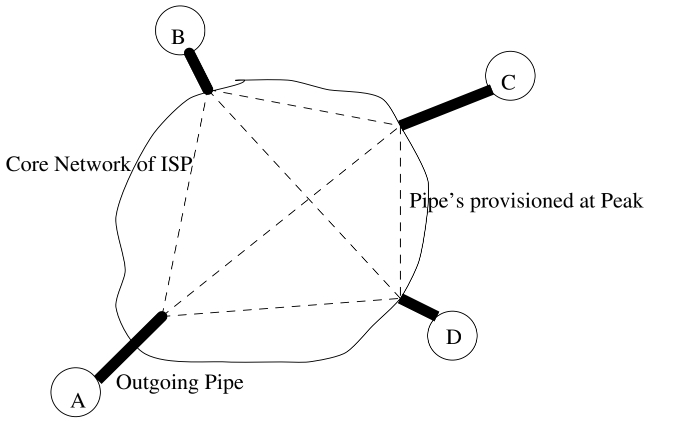

Fig. 1. The Point-to-Set Architecture: Site A transmitting to {B,C,D}

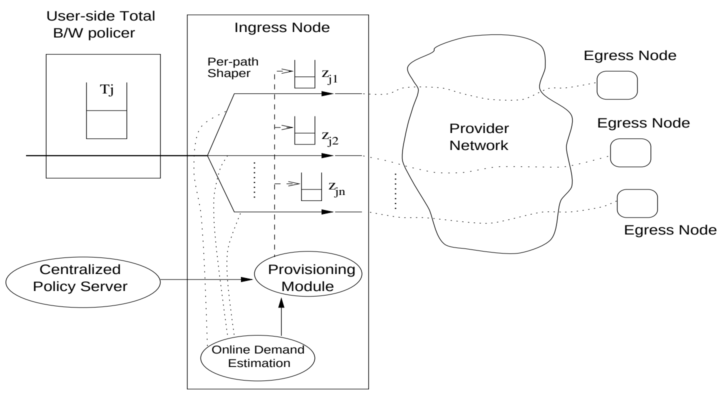

Fig. 2. The Point-to-Set Architectural Model with a Single Customer j

links need not be leased from A to the set. For the provider, the paths connecting edge A to the set $\{$B, C, D$\}$ can be multiplexed with other contracts.

In this paper we develop an edge-based point-to-set QoS model for a site-to-site private network. Instead of statically provisioning point-to-point links, at peak, to each source-destination pair, this architecture makes better use of the resources by allocating only as much bandwidth as the pair needs with suitable headroom to accommodate bursts. Depending on the traffic profile of the customer, the provider allocates bandwidth towards the various destinations in the set. Thus the provider network would appear as a set of variable-sized links connecting this edge to the destination set. In order to achieve this, we envision the need for demand estimation and provisioning components, which are described in the succeeding sections.

The point-to-set service is provisioned over a single provider (or cooperating multi-provider) IP inter-network. The service is negotiated with the provider as contracts which are admitted using an admission control scheme. The proposed scheme can be *completely provisioned and managed at the network edges* in coordination with a centralized admission control component. The point-to-set contracts are *weaker than traditional QoS contracts as they cannot offer strict guarantees and only offer statistical assurances*. In spite of this limitation, we believe that our point-to-set model is a first step in realizing Clark et al's [7] goal of going beyond point-to-point services and providing flexibility to users, while at the same time realizing multiplexing gains for providers.

LIRA [6] considers the problem of large spatial granularity. But the focus of [6] is not on VPN resource management. Duffield et al [11] propose a framework for VPN resource management. They introduce the idea of a "hose" as a resizeable access link for a VPN node. This is similar to the per-path dynamic provisioning presented in this paper. However, provisioning using the model in [11] as presented by Kumar et al [12] does not consider the problem of edge-based bandwidth provisioning among contracts, instead [12] constructs an optimized tree structure to connect VPN endpoints. Further, [11] uses a fixed measurement window based estimator while we present a new demand estimation algorithm based on correlation horizon of the underlying traffic process (refer section III-A).

The rest of the paper is organized as follows. Section II outlines the point-to-set architectural model. The building blocks are detailed in Section III-A, III-B and III-C which focus on online demand estimation, dynamic provisioning and admission control respectively. In Section IV we present the simulation results and compare our model with point-to-point resource allocation model. Section V summarizes and concludes the paper.

II. Point-to-Set Service Model

The following paragraphs provide an overview of the point-to-set architecture. The point-to-set service is offered by means of contracts. The contract for user j consists of:

- The finite, known set of destinations S_j with cardinality N_j.
- T_j, the aggregate traffic to the set, S_j, i.e. if t_{ji} is the traffic generated towards destination $i \in S_j$, then $\sum_i t_{ji} \leq T_j$.
- The per-path peak rate, p_{ji} on path ji.
- The per-path minimum assured rate, a_{ji} on path ji
- Duration of the contract. Not explicitly leveraged here.

Here a path refers to a set of links connecting a source edge to a destination edge. The per-path peak rates are assumed to have been determined in a characterization phase before the contract is arrived up on. The minimum per-path assured rate ensures that the user can get a minimum provisioning of a_{ji} on path ji at any time. As a simplification, this paper assumes the same value a_{ji} for all paths ji, calculated as T_j/N_j (henceforth, referred to as a_j). This constraint is enforced by admission control. Beyond that, the provisioning depends 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) and packet losses. One can imagine a regime without such a minimum service expectation, which could lead to higher multiplexing gains at the cost of higher provisioning/admission control errors.

The *point-to-set service* model (Figure 2) allows a site j to claim a pool of premium tokens T_j per unit time. The model lets the site j to flexibly use these tokens for the traffic towards it's set, subject to the limit that on each path ji the *peak rate* is less than or equal to p_{ji}. Observe that the user has the flexibility to use up to the peak rate p_{ji} 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 $T_j < \sum_i p_{ji}$ 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 path (Figure 2). We assume that bucket depths are minimal for smoothness. In this paper we consider only premium traffic. However, in a real implementation we can allocate capacity for the premium traffic and leave the rest for the best effort traffic.

From the provider perspective, however, this edge-based 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 require per-path shapers (at the ingress edge in Figure 2) operating at the peak rates p_{ji}.

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. It accounts for potential long-range dependence in traffic by exploiting the concept of *correlation horizons* [8]. In particular, the module detects the current correlation horizon dynamically and uses the mean and deviation of demand estimated on that horizon (Section III-A).

Second, these per-path demand estimates, d_{ji}, are used in a dynamic provisioning module (d_{ji} is the estimate for site's traffic from j to destination i in the set). This module operates in two phases: a) per-path provisioning phase and b) per-contract provisioning phase. The two phases can be explained as follows. On a given path, there could be many contending contracts. The sum of the demand estimates of all contracts on that path may exceed the path capacity, i.e., $\sum_j d_{ji} > C_{ji}$, where C_{ji} is the capacity of the path ji. The "Per-Path" provisioning module allocates y_{ji} (on the path ji) such that $\sum_j y_{ji} \leq C_{ji}$. Also, the sum of the allocations made at each path, for a particular contract, may exceed the contract's total bandwidth T_j, i.e., $\sum_i y_{ji} > T_j$. "Per-Contract" provisioning is performed to ensure that the final allocations conform to the contracted bandwidth. The resultant per-path rate limits z_{ji} are enforced at the provider-side shapers for each path ji (note that $\sum_i z_{ji} \leq T_j$). The provider can attain multiplexing gains if $T_j < \sum_i p_{ji} \Rightarrow z_{ji} \leq p_{ji}$.

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. Since the re-provisioning occurs periodically, the rate changes within a provisioning time scale are accounted for, only in the next period. Buffer overflows during such phases lead to packet loss, i.e., degradation of assured service which is a cost of attaining multiplexing gains. The case of consistent queuing due to errors in admission control and persistent shifts in traffic can be dealt with by forced re-negotiation of peak assured rates p_{ji} only on

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

the affected paths i. Handling admission control errors and renegotiations are not discussed in this paper.

The choice of provisioning time-scales is important since it decides the responsiveness of the provisioning algorithm to traffic changes. We assume that online measurement intervals (Section III-A) are much more fine-grained than provisioning time-scales. The provisioning timescale is set based on the round-trip times between the edge routers. Our simulations show that provisioning timescales of less than twice the round-trip between the edges suffices the requirements. This is discussed in more detail in Section IV.

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 and meanarrivals as previous 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}
$std\_deviation = sqrt(sqrmeanarrivals[optimalwt] - (meanarrivals[optimalwt])^2)$;
$prediction = meanarrivals[optimalwt] + 2 * std\_deviation$

III. Components of the Point-to-Set Architecture

The architecture contains the following components: data plane components (provider-side shapers), online demand estimation component, per-path and per-contract dynamic provisioning and an admission control component. These topics are elaborated in the following sub-sections.

A. Online Demand Estimation

Experimental evidence [9] 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 modeling such traffic. However, recent work [8] analysing finite buffer

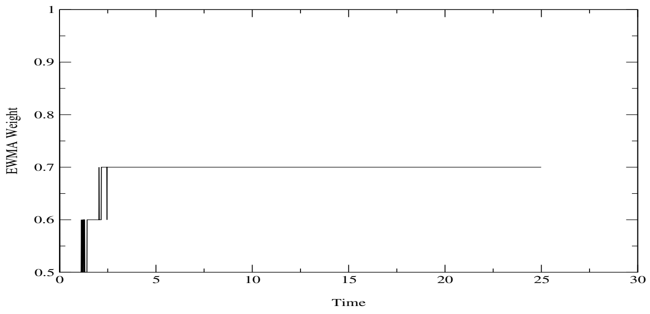

Fig. 3. EWMA weight α vs Time

queues with LRD input indicates that only a finite amount of correlation need be considered for certain queuing 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 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 models with finite memory as long as it captures correlation up to the correlation horizon for the system.

We 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. The demand estimation scheme hence maintains a *vector of EWMA predictors* for different weight parameters ($\alpha \in [0.05, 0.95]$). 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_{ji} in subsequent sections.For details on the demand estimation module the reader is referred to [2].

B. Per-Path Dynamic Provisioning

As mentioned earlier the dynamic provisioning decision is made in two phases, and uses the demand estimate d_{ji} to finally yield the rate limits for the shapers z_{ji}. The per-path decision is made with the knowledge of demand estimates for competing contracts on that path. The decision is then used in the second phase (per-contract provisioning) to calculate shaper settings. Let there be N contracts on the path. Then the per-path provisioning proceeds as follows:

- If the sum of per-path provisioning estimates (over all contracts) on a path ($\sum_j d_{ji}$) is less than or equal to the path capacity C, then allocate $y_{ji} = d_{ji}$ to each shaper. Otherwise:

Algorithm 2 Per-Path Provisioning Decision

For all $d_{ji} \leq a_j$ allocate d_{ji}. Delete these contract from the List. Calculate available resource A, as the difference of allocations made and the path Capacity C.
For the rest of the contract allocate a_j, where a_j is the minimum assured rate for the Contract j. Define $x_{ji} = a_j$ and re-calculate A.
while $A > 0$ **do**
  Define *diff* = Minimum($d_{ji} - x_{ji}$) over all j, M as the number of contracts for which provisioning decision has to be made.
  if *diff* $\geq A/M$ **then**
    Define *increment* = A/M
    Allocate resources equal to *increment* to each contract.
    $A = 0$
  else
    increment = *diff*
    Allocate resources equal to *increment* to each contract.
    Update x_{ji} as x_{ji} += *increment*
    A -= *diff* * M
    Delete the contracts for which ($d_{ji} - x_{ji}$)=0 from the List (of contracts for which provisioning has to be made).
    Update M.
  end if

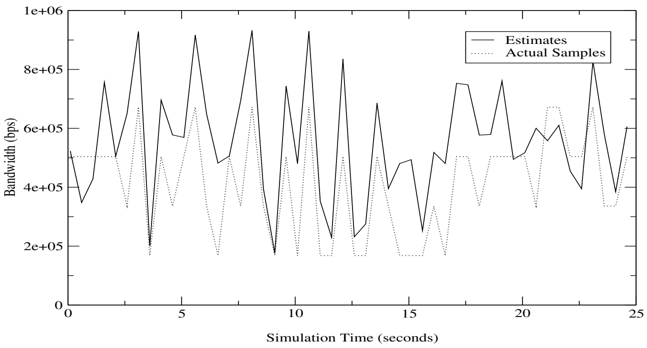

Fig. 4. Online Estimates of Traffic vs Time

- For all the contracts where $d_{ji} \leq a_j$ allocate d_{ji}, i.e., $y_{ji} = d_{ji}$. Set

$$C' = C - \sum_{\{j: d_{ji} \leq a_j\}} d_{ji}$$

for every allocation.
- For the remaining contracts use a max-min strategy to arrive at the allocations y_{ji} subject to the following constraints:

$$\sum_{\{j: d_{ji} > a_j\}} y_{ji} = C', \quad (1)$$

$$y_{ji} \leq d_{ji}, \forall j \quad (2)$$

$$y_{ji} \geq a_j, \forall j : d_{ji} > a_j \quad (3)$$

The solution to the above problem is suggested in Algorithm 2. In the case where, $d_{ji} \leq a_j$, another allocation strategy could have been always provisioning a_j. Though this assures a contract of a minimum guaranteed bandwidth, it will be at the cost

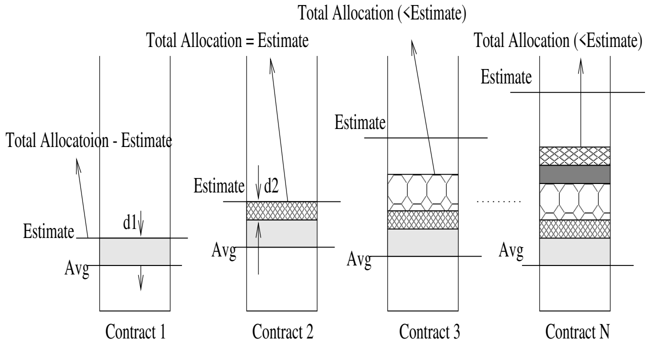

Fig. 5. Per-Path Re-provisioning Policy

of greater losses for some other contract, which makes excursions to peak. Assuming that a contract will make excursions above and below the contracted minimum assured rate (a_j), by allocating bandwidth only equal to the estimate (even though it is below a_j) we distribute losses uniformly over time. We now describe the per-path provisioning policy. We first allocate resources to all the contracts whose estimate is less than the minimum assured rate and eliminate them from the provisioning list. Figure(5) shows all the remaining contracts. Here we first allocate the corresponding assured rate *avg* to each contract. Then, the minimum difference between the estimate and the difference, $d1$, is calculated and is allocated to each contract, and the available bandwidth on the path is updated. With this allocation, we find that the first contract's demand has been met, hence 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 path.

B.1 Per-Contract Re-provisioning

At the end of the Per-Path Re-provisioning, the allocations y_{ji} are obtained. This means that the per-path allocation is y_{ji} to contract j on path ji. However, the total suggested allocations for the contract may exceed his total contracted bandwidth T_j. So another level of re-provisioning amongst different paths in a contract is needed. This re-provisioning can be again expressed as an optimization problem. Let z_{ji} be the final allocation made (by agent) to Contract j on path ji and N_j be the number of destinations in the Contract j's set. Then a max-min strategy can be used to arrive at z_{ji} subject to the following constraints:

$$\sum_{i=1}^{N_j} z_{ji} = T_j, \quad (4)$$

$$z_{ji} \leq y_{ji}, \forall i \quad (5)$$

$$z_{ji} \geq min(y_{ji}, a_j), \forall i \quad (6)$$

The solution to the above problem can again be computed using the algorithm 2 albeit taking care of the new variables and the constraints.

C. Admission Control

A centralized Policy Server makes all the Admission Control decisions. The Policy Server is assumed to have a map of

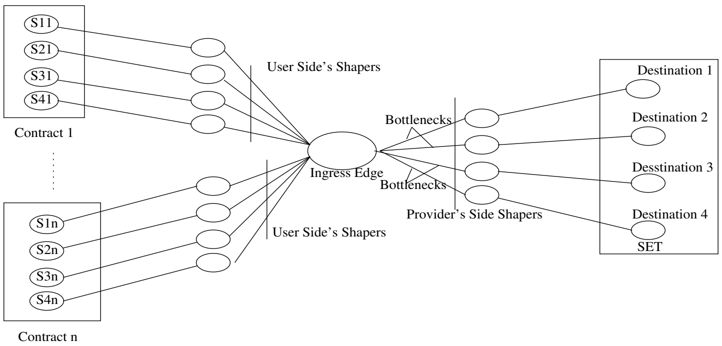

Fig. 6. Single Ingress Topology

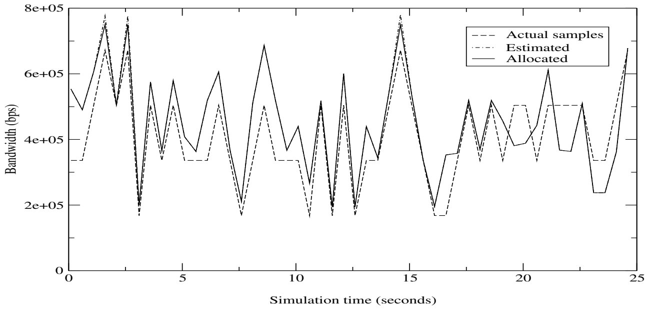

Fig. 7. Allocated, Estimated bandwidth and Actual Traffic Samples Vs Time

the network including link bandwidths (available through OSPF extensions [3]) and information regarding the current set of contracts. On the arrival of a new customer, the following steps are undertaken by the policy server. First, the contract is mapped to a set of paths leading from the source edge to the destination edges. Second, the decision whether to admit this contract is taken. This is done by computing the available bandwidth on every path ji and ensuring that it is greater than the contract's minimum assured rate a_{ji}, for each path ji. The available bandwidth is the path capacity reduced by the minimum assured rates a_{ji} of all existing contracts (i.e., $C_i - \sum_j a_{ji}$).

The following simple strategy is proposed to compute path capacities. For each link calculate the number of paths passing through it. Divide the link capacity by this number and consider this new capacity as the "effective" capacity available to each path passing through this link. For each path, the link with the least "effective" capacity decides the path capacity. Routing changes require that the path capacities be re-computed and may also require contracts to be re-examined. In this paper it is assumed that routing is stable over longer timescales and hence the path capacity remains constant throughout the lifetime of the contract. For a measurement-based admission control strategy the reader is referred to [1].

IV. Results and Discussion

In this section we present the simulation results. We evaluate the point-to-set model with a single ingress topology and a multiple ingress topology. With the single ingress topology, the performance of the provisioning algorithm is first examined and then, the customer and provider gains are presented. For the multiple ingress topology we evaluate the customer gains. Finally, we examine the effect of the provisioning timescale on the performance of the model.

A. *Single Ingress Topology*

The topology shown in Figure 6 (single ingress) was simulated on NS.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 paths are of 25 Mbps and the RTT is 60ms.All the paths other than the constrained paths are of 10 Mbps. The source traffic is shaped by static (customer side) shapers to ensure conformity with the contract, in terms of the peak rates. The provider network features the dynamically provisioned shapers which adapt to the user's traffic distribution with respect to his destinations.

The point-to-set model was tested in the following settings:

- **Traffic simulated by agents provided by NS:** Constant Bit Rate traffic sources (CBR) with rates varying randomly with time were employed. A *uniform* random variable was sampled to obtain the rates for the CBR agents every 0.15 seconds. The traffic was generated such that it is distributed uniformly across destinations. Thus it is equally likely that a particular rate is observed towards a destination. Note that the dynamically provisioned shapers adapt to the traffic profile of the customer and are not in anyway aware of this arrangement.
- **Trace driven simulation:** MPEG encoding of the Star Wars movie, converted into the NS trace format [10], was employed to generate video traffic. The source was attached to the NS UDP agent and it picks a random start point in the trace file.

A.1 Performance of the provisioning scheme

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 policed, 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.

A.2 Customer gain with simulated traffic

To examine the customer gain, we employ the ratio of the per-path peak, p_{ji}, to the total contracted bandwidth for the set of destinations, T_j. Observe that the customer is paying for T_j, that is T_j/N towards each destination in the set, while being allowed to send p_{ji} towards destination j. Thus, higher the ratio p_{ji}/T_j, higher the gain for the customer. However, it is important to note that a higher value of this ratio implies *greater strain* on the provisioning algorithm and a higher trade-off in terms of drops. This is because, the provider has to maintain a "zero sum game" in terms of the bandwidth allocated to this contract towards the destinations in its set. Also, the tracking algorithm has to deal with higher variability of offered load towards a destination.

In the simulations, p_{ji} values were assumed to be equal (say, P_j) Figure 8 and table I depict the drop rate with increasing number of contracts for various values of P_j/T_j, for simulated traffic. For a given number of contracts the drop rate increases with the ratio P_j/T_j. However, we observe from the graph (and from table I) that the drop rates are within 0.8% even for a peak

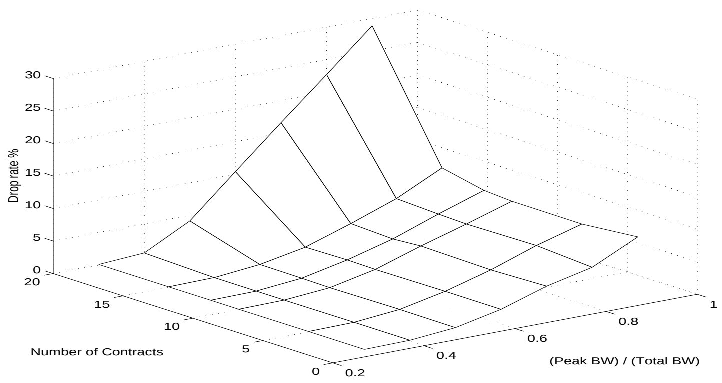

Fig. 8. Drop Rates (%) against number of contracts and Peak/Total ratio

Num Contracts	$p_j/T_j = 0.5$	$p_j/T_j = 0.6$	$p_j/T_j = 0.7$	$p_j/T_j = 0.8$
10	0.7425	2.2410	4.5084	6.5784
12	0.8202	2.3092	4.2187	6.7317
15	0.8812	2.2225	4.5198	7.0512

TABLE I

SIMULATED TRAFFIC: DROP RATES (%), WITH 25M CONSTRAINED LINK, TOTAL BANDWIDTH PER CONTRACT = 5M AND PER PATH PEAK VARYING FROM 2.5M TO 4M

to total ratio of 0.5. This implies that with a low drop rate, the customer can offer a peak rate of $N\Sigma_i p_{ji}$, (N=4 in the simulations) while paying for $2P_j$ (assuming the same peak towards all destinations, $P_j = p_{ji}$ for all i).

A.3 Customer gain with trace-driven simulations

In this section we present the Customer gains with the trace of MPEG encoding of Star Wars movie. In figure 9 the drop rates (%) are plotted against the number of contracts and the ratio p_j/T_j. The plot and the data in table II indicate that even with a peak to total ratio of 0.6 the drop rates are within 0.8%. This reinforces the fact that there is cost saving for the customer in opting for the point to set model, if the specified error rates are tolerable.

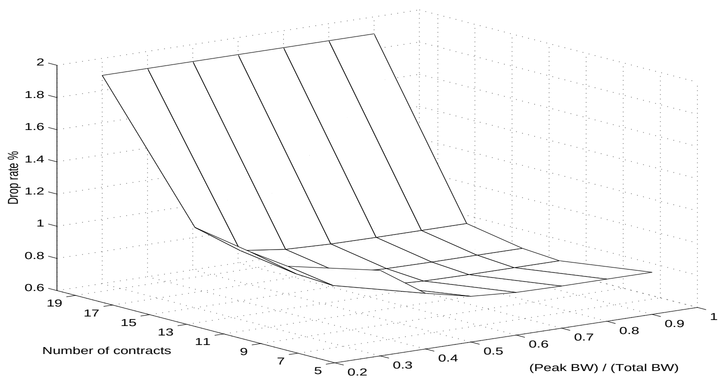

Fig. 9. Star Wars trace: Drop Rates (%) against number of contracts and Peak/Total ratio

Num Contracts	$p_j/T_j = 0.5$	$p_j/T_j = 0.6$	$p_j/T_j = 0.7$	$p_j/T_j = 0.8$
10	0.8007	0.7863	0.7834	0.7833
12	0.8167	0.8042	0.8013	0.8012
15	0.8769	0.8676	0.8654	0.8654

TABLE II

STAR WARS TRACE: DROP RATES (%), WITH 25M CONSTRAINED LINK, TOTAL BANDWIDTH PER CONTRACT = 5M AND PER PATH PEAK VARYING FROM 2.5M TO 4M

A.4 Provider gain with trace-driven simulations

With dynamic provisioning of resources, the provider sets aside as much bandwidth as is demanded by the customer traffic. If the average demand seen for a contract j is $\bar{d}_{ji}$ towards destination i, then the ratio of peak P_j to $\bar{d}_{ji}$ quantifies the average gain for the provider. Taking note of this fact, the table III presents relevant data for a particular path. Each row of the table gives data for a particular range of gain. Thus in the simulation with 12 contracts, there were 3 contracts for which the gain was in the range $(1.0, 1.5)$ and the average gain was 1.14. From table III we see, that higher gains (for the provider) are

Num Contracts	Gain Range	Avg Gain $P_j/\bar{d}_{ji}$	Drop Rate (%)	Num Contracts seeing gain
12	(1.0,1.5)	1.14	0.04	3
	(1.5,2.0)	1.65	0.66	3
	(2.0,2.5)	2.03	0.40	1
	(2.5,3.0)	2.86	1.40	2
	(3.5,4.0)	3.15	1.00	2
	(4.0,4.5)	4.45	2.20	1
15	(1.0,1.5)	1.09	0.13	3
	(1.5,2.0)	1.70	0.65	4
	(2.0,2.5)	2.39	0.40	1
	(2.5,3.0)	2.77	0.50	1
	(3.5,4.0)	3.53	1.47	3
	(4.0,5.5)	4.80	1.60	3
20	(1.0,1.5)	1.15	0.88	5
	(1.5,2.0)	1.69	1.26	6
	(2.0,2.5)	2.19	1.00	1
	(3.5,4.0)	3.30	2.84	5
	(4.0,4.5)	4.42	3.47	3

TABLE III

STAR WARS TRACE: PROVIDER GAIN WITH 25M CONSTRAINED PATH, TOTAL BANDWIDTH PER CONTRACT = 5M AND PER PATH PEAK = 2.5M

accompanied by a higher drop rate (for the customer). But the average gain with drop rates within 0.65% was 1.14. This means the provider can increase his resource utilization by about 39% $(1 - \bar{d}_{ji}/P_j)$ while keeping drop rates within 0.65%.

It is important to observe here that with 15 contracts, although the sum of the peak rates (37.5M) exceeds the path capacity the drop rates are low. This means that as compared to static provisioning, point-to-set service delivers considerable gains to the provider. However if the number of contracts being multiplexed increased to 20, the drop rates go beyond 1% since the sources are more likely to overwhelm the path capacity.

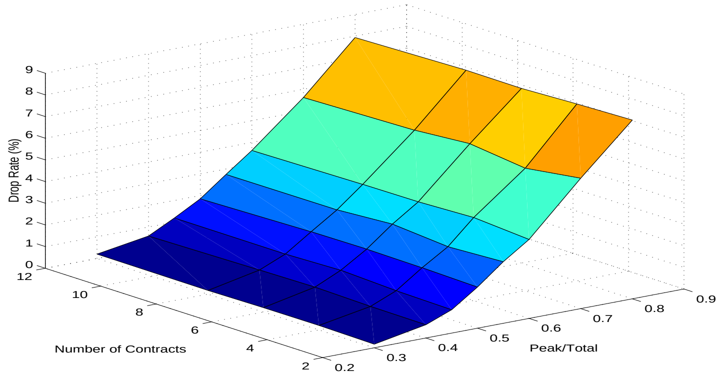

Fig. 10. Simulated Traffic: Drop Rates (%) against number of contracts and Peak/Total ratio for multiple ingress topology

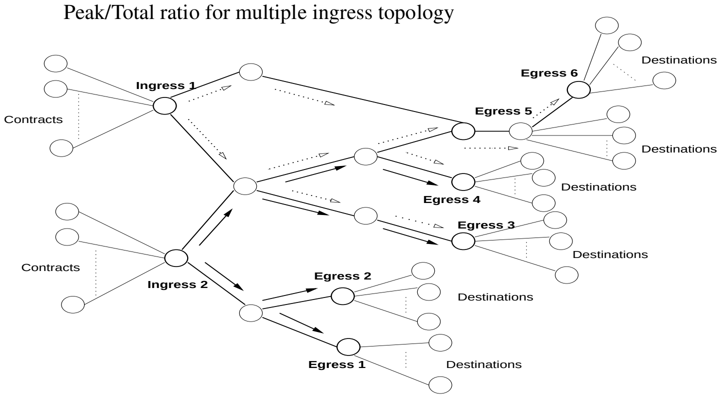

Fig. 11. Multiple Ingress Topology

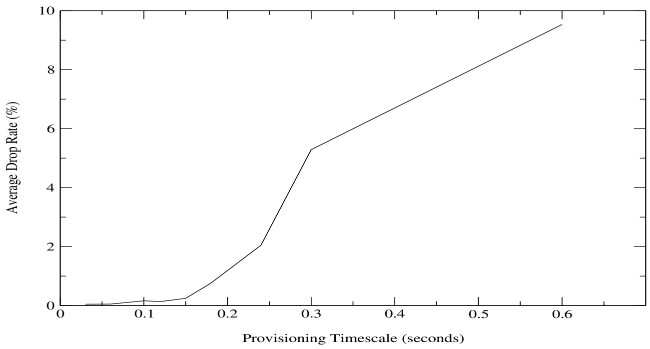

Fig. 12. Drop Rates (%) against different provisioning timescales

B. Multiple Ingress Topology

In this section we evaluate the point-to-set architecture with a multiple ingress topology. Figure 11 shows the topology (a graph derived from the well know MCI backbone topology) with the ingress and egress nodes in bold. The sources generate traffic as described in Section IV-A. The core of the network has links of capacity 75Mbps and are depicted by thick lines. The access links which input traffic for each contract, were chosen to be 10Mbps. The total (T_j) for each contract was 10Mbps.The minimum assured rate was thus 2.5Mbps. For simplicity, the per-path peak values were assumed equal. The traffic from *Ingress 1* is marked by dotted arrows, while that from *Ingress 2* is marked with bold arrows. Traffic from each of the ingresses reach four egresses. Due to the way the traffic is routed, the path connecting *Ingress 2* to *Egress 4* and those linking *Ingress 1* to *Egress 5 and 6* share a link. Thus this link has three paths through it. In the simulation a simplistic strategy of dividing the link capacity by three was used to decide the capacity of each of these paths. Similarly the other path capacities are decided.

The drop rates observed with simulated traffic and multiple edge topology is seen in figure(10). The drop rates for a peak to total ratio of 0.45 is about 1.2%. With the single ingress topology, the observed drop rate for p_j/T_j set to 0.5 is around 0.8%. Thus the multiplexing of paths in the topology caused a degradation in the service. But still the customer sees a gain of about 44.4% with a modest drop rate of 1.2%.

C. Effect of provisioning timescale on performance

The provider-side shapers are dynamically re-provisioned periodically every T seconds. The choice of T affects the performance of the provisioning algorithm. This is because, T decides the responsiveness of the decision process to changes in the input traffic profile. If the response is quick, the drop rates are reduced. In the above simulations the timescale was set to 1.5 times the edge to edge propagation delay (0.1 seconds). The effect of varying T (in multiples of RTT) on drop rate is shown in figure 12. As the provisioning timescale increases with respect to the edge to edge propagation delay, the drop rates increase.

V. 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, online demand estimation, dynamic provisioning and suggestions for a simple admission control technique. Simulation results were presented to demonstrate the gains to the customer and provider due to the scheme. The scheme was shown to deliver gains to the provider in terms of resource utilization while keeping the drop rates low. The customers on the other hand stand to gain in terms of cost savings in that only the required total bandwidth needs to be purchased. Future work will focus on handling routing changes, re-negotiation of contracts and quantifying the errors in admission control, estimation and provisioning.

References

[1] S. Raghunath, K. Chandrayana, S. Kalyanaraman, "Edge-Based QoS Provisioning for Point-to-Set Assured Services," Techreport. http://networks.ecse.rpi.edu/˜rsatish/icc_techreport.ps

[2] S. Raghunath, S. Kalyanaraman, "Dynamic Provisioning for Differentiated Services on the Internet," Techreport. http://networks.ecse.rpi.edu/˜rsatish/dynaprov.ps

[3] G.Apostolopoulos et al, "QoS Routing Mechanisms and OSPF Extensions,"*RFC 2676,* August 1999

[4] S Blake et al, "An Architecture for Differentiated Services", *IETF RFC 2475,* December 1998.

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

[6] Ion Stoica, Hui Zhang, " LIRA: An Approach for Service Differentiation in the Internet",*Proc. of NOSSDAV,* England, pp. 115-128d, July'98.

[7] D. Clark, W. Fang, "Explicit Allocation of Best-Effort Packet Delivery Service", *IEEE/ACM Trans. on Networking,* 6 (4), pp. 362-373, Aug'98.

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

[9] W. E. Leland et al, "On the self-similar nature of Ethernet traffic (Extended version)," *IEEE/ACM Trans. on Networking*, 2(1):1-15, February 1994

[10] "Traffic Trace for ns", http://www.research.att.com/˜breslau/vint/trace.html

[11] N.G. Duffield et al, "A Flexible Model for Resource Management in Virtual Private Networks", *SIGCOMM'99*.

[12] A. Kumar et al, "Algorithms for Provisioning Virtual Private Networks in the Hose Model", *SIGCOMM'01*