Statistical Bandwidth Sharing in End-to-End Connectivity Management over Bandwidth-provisioned Networks

Kaliappa Ravindran & Mohammad Rabby
Department of Computer Science
Graduate School and University Center
The City University of New York
New York, NY 10019
Email: ravi@cs.ccny.cuny.edu,mfrabby@yahoo.com

Abstract— **The paper describes a policy-based model for cost-effective 'data connectivity' provisioning between session-level end-points. The connectivity provider (SP) may employ an architecture for end-to-end QoS control between data aggregation points. It involves: i) maintaining multiple *diffserv*-type connections between end-points with parameterizable QoS differentiation between them, and ii) admission control at end-points with *intserv*-type bandwidth management over connections. (ii) aggregates data flows with closely-similar QoS needs over a single end-to-end connection. (i) apportions the available infrastructure bandwidth between various end-to-end connections that carry (aggregated) data flows with distinct QoS levels. Flow aggregation over a connection allows reaping the statistical multiplexing gains in bandwidth, i.e., meets the SP's revenue incentives. Whereas, connection-level bandwidth allocation allows meeting the QoS needs of data flows, i.e., guarantees the end-user's utility. The management functions of SP monitor the changes and/or outages in network bandwidth in a dynamic setting, and maps them onto the connectivity costs incurred for Qos control. Our model allows installing policy functions at end-points that can make the connectivity provisioning cost-optimal.**

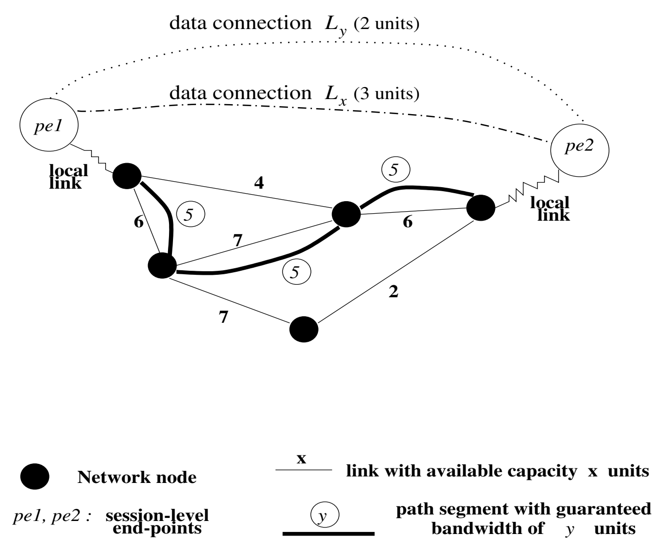


Fig. 1. Bandwidth-controlled connectivity

I. Introduction

The provisioning of end-to-end 'data connectivity' may be viewed as a service offered over the underlying network infrastructure. Clients may in turn build upon a connectivity service to provide higher level information-oriented services (such as image downloads, data mining from remote sites, and video telephony). The connectivity service provider (SP) may set up end-to-end paths between data aggregation points — say, between New York and London. Individual clients may then exchange high volume information over these data paths for sports, business, and entertainment applications. The SP may possibly lease the bandwidth from infrastructure networks (say, telecom companies such as AT&T) for providing a session-level 'data connectivity' between end-points.

The SP is faced with two conflicting goals: reducing the bandwidth costs incurred for data transfers (to maximize the SP's revenues) and allocating enough bandwidth to meet the QoS needs of application sessions (to satisfy the end-user's utility). The SP needs to implement policy mechanisms and management tools that allow balancing these goals. In this paper, we identify the end-point architectures and protocols that enable the SP to attain the revenue and QoS objectives.

Figure 1 shows a session-level data path set up between end-points pe_1 and pe_2. Each segment in the path may be a native communication link between the routers of an IP network or a TCP (or UDP) connection set up between the nodes of an overlay network. Or, the entire path between pe_1 and pe_2 may be a leased line with dedicated bandwidth. Regardless of the network infrastructure, the end-system treats the data path between pe_1 and pe_2 as a single object for the purpose of bandwidth management and admission control.

The SP may employ an architecture based on *data flows* and *path guarantees* (in a statistical sense) to exercise end-to-end QoS control. It involves:

- Maintaining multiple *diffserv*-style data paths between end-points, with parameterizable QoS differentiation between them;
- Admission control at end-points, with *intserv*-style bandwidth management over data paths.

1-4244-1251-X/07/$25.00 2007 IEEE.

The admission control function in an end-system aggregates a large number of data flows with closely-similar QoS needs over a single path. The traffic correlations that exist among such flows allows reaping the statistical multiplexing gains in bandwidth. The path maintenance function in the end-system suitably apportions the available infrastructure bandwidth between the various paths that carry (aggregated) data flows with distinct QoS levels. This bandwidth apportionment allows the SP to enforce per-flow QoS guarantees. Thus, the SP-level mechanisms purport to manage the network infrastructure bandwidth usage from an end-to-end QoS control standpoint.

Referring to Figure 1, the available bandwidth between pe_1 and pe_2 is 6 units. QoS-controlled data flows consume 5 units, with the surplus 1 unit allocated to carry, say, 'best-effort' traffic. The 5 units of bandwidth may in turn be split across two data connections, say, 3 units along L_x and 2 units along L_y to carry high resolution and low resolution video traffic respectively. Here, the SP-level control is about estimating the bandwidth of 5 units needed for video traffic and splitting this bandwidth as 3 units and 2 units for L_x and L_y respectively.

The SP may use *policy* functions that prescribe how distinct the flow specs characterizing various data connections are and what cost the per-flow bandwidth apportionment over a data connection incurs. The management functions of SP monitor the changes and/or outages in network bandwidth in a dynamic setting, and maps them onto connectivity costs incurred under a given policy. The SP may also dynamically switch from one policy to another, based on how the costs of bandwidth usage change as the network operating point changes.

Our architecture allows installing a repertoire of policy functions at end-points and selecting the appropriate ones to make the connectivity provisioning cost-optimal. The paper provides the functional mechanisms to realize the policy switching while sustaining a user-transparent connectivity provisioning. These mechanisms are based on our studies on different types QoS-controlled data connections. Overall, our architectural model can be incorporated into the 'telecommunications management' framework (TMN) that has been standardized for network services [1].

The paper is organized as follows. Section II describes a QoS-oriented view of 'data connectivity'. Section III describes how 'data flows' and 'data connections' are managed by the end-system in our architectural model. Section IV motivates the design decisions and functional elements in the model. Section V identifies the end-system mechanisms needed to support the model. Section VI compares our approach with existing methods for scalable connectivity support without per-flow management. Section VII concludes the paper.

II. Bandwidth-provisioned connectivity

In our model, the links that provide the physical connectivity between end-points constitute the 'infrastructure', and the available link capacities in a path chosen to connect peer entities constitute the 'resource'.

A. A management view of connectivity

The management control is exercised on two types of session-level objects: 'data flow' and 'data connection'. A 'data flow' is a sequence of packets transported from the source to receiver entities, subject to a certain end-to-end QoS. A 'data connection' is set up over the transport path between source and receiver entities, with a prescribed amount of bandwidth allocation to carry a group of data flows with a closely-similar QoS characteristics. See Figure 2. A 'data connection' is the object granularity for bandwidth allocation purposes, whereas a 'data flow' is the object granularity for end-to-end admission control[1].

Suppose a flow parameter r captures, at a macro-level, the bandwidth usage. An estimation of the bandwidth needs may be represented as a mapping function:

$$\mathcal{F} \ : \ r \in \mathcal{Q} \ \rightarrow \ b \in (0, \mathcal{W}),$$

where $\mathcal{Q}$ and $\mathcal{W}$ represent the flow parameter space and network capacity respectively (a one-to-one mapping exists between r and b). The SP maps the bandwidth usage to a cost based on the capacity leasing arrangement with the infrastructure. The SP's goal is to reduce the total cost of data connectivity by exploiting the statistical multiplexing gains in bandwidth among data flows.

B. Cost reduction by flow multiplexing

Given a bandwidth allocation $b_r = \mathcal{F}(r)$ for a data flow r, the total bandwidth usage incurred by a 'data connection' C can be transcribed into a cost $\Theta(\sum_{\forall r} b_r)$ for transporting multiple data flows $\{r\}$ over C. The function $\Theta(\cdots)$ maps a bandwidth usage onto a cost — which may include the infrastructure-level tariffs incurred for bandwidth and any fixed cost of maintaining the connection.

The *weakly additive* nature of bandwidth usage by bursty data flows is captured by the *monotonic concavity* of $\mathcal{F}$, denoted as:

$$\begin{aligned} \mathcal{F}(r') &> \mathcal{F}(r'') \\ \mathcal{F}(r'+dr) - \mathcal{F}(r') &< \mathcal{F}(r''+dr) - \mathcal{F}(r''), \end{aligned}$$

for $r' > r''$. Given a cost relation Θ, the above monotonicity condition depicts the cost savings that arise due to the leasing of large bandwidth to carry multiple data flows. Such[2] a quantitative reflection of network bandwidth usage onto the flow parameter space $\mathcal{Q}$ enables the SP to exercise a revenue-oriented control of bandwidth usage.

Consider two different policies $\mathcal{F}$ and $\mathcal{F}'$ that purport to provide data connectivity. That both $\mathcal{F}$ and $\mathcal{F}'$ employ some form of bandwidth allocation allows cost comparisons on a common scale. In general, a measure of the relativistic cost variations of $\mathcal{F}(r)$ and $\mathcal{F}'(r)$ with respect to the flow

[1]Referring to Figure 1, multiple TCP flows between the end-points pe_1 and pe_2 may be bundled in to a single session-level object, namely, a 'data connection' in our model, for bandwidth management purposes.

[2][3] gives guidelines to prescribe the '$>$' relation over $\mathcal{Q}$.

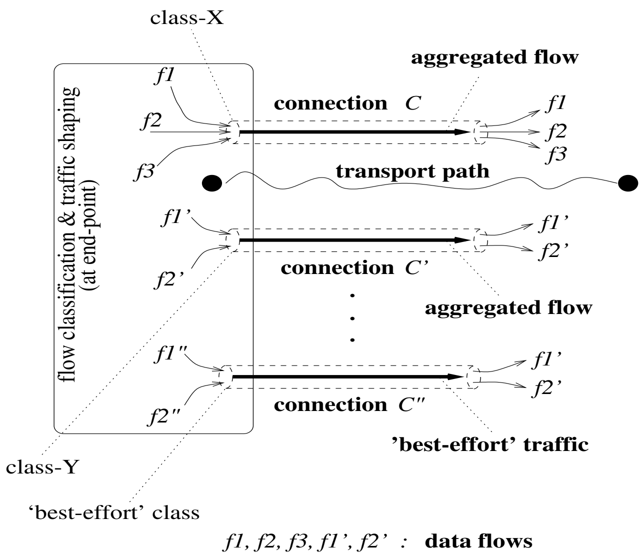


Fig. 2. 'data connection' versus 'data flows'

characteristics of r is needed for cost comparison, so that $\mathcal{F}$ or $\mathcal{F}'$ is chosen to provide the required data connectivity.

C. Application-level flow specs

A policy function $\mathcal{F}$ maps a flow specs $r \in \mathcal{Q}$ to the bandwidth needs b_r at network elements in a data path. In one form, r may be given by a peak rate p, average rate A, loss tolerance limit Δ (specified as a fraction of average rate), delay tolerance limit $\mathcal{D}$, and auto-correlation parameter ζ of data traffic — where $0 < A < p$ and $0.0 < \Delta \ll 1.0$. Note that $\zeta \in (0.0, 1.0)$, with $\zeta \rightarrow 0.0^+$ indicating a totally random flow and $\zeta \rightarrow 1.0^-$ indicating a high degree of statistical dependence of the current data rate on past rates. [2] discusses these parameters from a network engineering standpoint.

Consider a data flow of type $r = (A, p, \zeta, \Delta, \mathcal{D})$ over a network element E. Given[3] a policy function $\mathcal{F}$, the monotonic concavity of $\mathcal{F}$ depicts an *optimistic* allocation of bandwidth over E — which assumes that the peak rate of flow does not persist long enough to backlog packets at the input queue of E to a level where more than a fraction Δ of the packets will miss their deadlines prescribed by $\mathcal{D}$. Such an allocation will have: $[A - \Delta] < \mathcal{F}(r) < p$, with the actual allocation determined by $\mathcal{D}$, duration of peak p, ζ, and input queue length of E. If $\mathcal{F}$ and $\mathcal{F}'$ depict policies such that $\mathcal{F}(r) > \mathcal{F}'(r)$ for some $r \in \mathcal{Q}$, then $\mathcal{F}(r') > \mathcal{F}'(r')|_{\forall r' \in \mathcal{Q}}$ — and we say that $\mathcal{F}'$ is more aggressive than $\mathcal{F}$.

Note that the flow type r may be viewed as a 'traffic class' in an extended form of DiffServ architecture. A connection $C(r)$ is a 'DiffServ' path to carry a group of data flows $\{f_1, f_2, \cdots\}$ of type r. Multiple data connections may be created, say, $C(r)$, $C'(r')$ and $C''(r''$, to carry different groups of data flows, say, $\{f_1, f_2, f_3\}$, $\{f_1', f_2'\}$ and $\{f_1'', f_2''\}$ respectively — all sharing the available infrastructure bandwidth B. Figure 2 shows such a scenario. It depicts a 'proportional differentiation' in the end-to-end scheduling of packets of different traffic classes [5].

III. End-point bandwidth control

We now describe a management view of the mechanisms that exercise the connection and flow objects.

A. Macroscopic estimation of bandwidth

$\mathcal{F}$ encapsulates a bandwidth allocation policy that can be installed at the end-points. Typically, an allocation $b_r = \mathcal{F}(r)$ is such that $[A - \Delta] < b_r < p$, with the constraint that the packet loss over the observation interval T_{obs} is less than Δ. An example of $\mathcal{F}$ is to reserve 10% additional bandwidth relative to that necessary to sustain the average rate A. For $\Delta > \text{link error } e$, packet losses arising from a less-than-peak allocation are indistinguishable from those arising from the infrastructure-level link error characteristics. Typically, the scheduler should visit the packet queue of C for a portion $\frac{b}{\text{CAP}(E)}$ of T_{obs}. For $\Delta \rightarrow 0^+$, the allocation should be more than the estimated bandwidth needs for the flow with $e = 0$, with the additional bandwidth required to meet the application requirement, if any, for recovering lost packets through retransmissions (such as TCP error control).

Packet loss may also arise due to an actual data traffic exhibiting a peak rate higher than that specified. This is despite any traffic shaping by an admission control function that is embodied in $\mathcal{F}$. This however does not depict an incorrect flow specification, since flow specs is only a macroscopic characterization of the traffic behavior. Though a traffic shaper can bound the peak rate (say, with a 'leaky bucket'), the average rate of actual traffic may itself change relative to the specified rate (over a slow time-scale). This difference can skew the bandwidth estimation, resulting in queue overflows.

The SP may install different policy functions $\mathcal{F}, \mathcal{F}', \cdots$ at appropriate control points of a 'data connection'. Our focus here is not on the accuracy in estimating the bandwidth needs itself, but is on the management support to make a reasonable estimate from the traffic-oriented QoS parameters. Note that the IntServ-style bandwidth allocation embodied in our model is exercised only at the end-system[4].

B. State information at end-points

Aggregating multiple flows over a single connection C reduces the scheduling overhead, relative to setting up a separate connection for each data flow. Figure 3 illustrates the state information maintained at end-points to support flow aggregation. The key pieces of state information include the QoS specs that classifies the component flows, the number of flows multiplexed, the policy function to map QoS specs to bandwidth needs, and the available bandwidth on a connection. Since this information is maintained at connection-level, the amount of per-flow state is reduced by $\mathcal{O}(n)$, where n is the

[3]The (p, A, Δ, ζ) tuples may be viewed as prescribing distinct 'virtual link classes' (see [4] in this context). The admission controller then maps an application-generated data flow to one of these 'virtual links'.

[4]The IntServ-type and DiffServ-type of functional elements in our end-system model are inspired by, but are different (both architecturally and in scope) from, the IntServ and DiffServ architectures proposed by IETF for use in core network elements [6], [7].

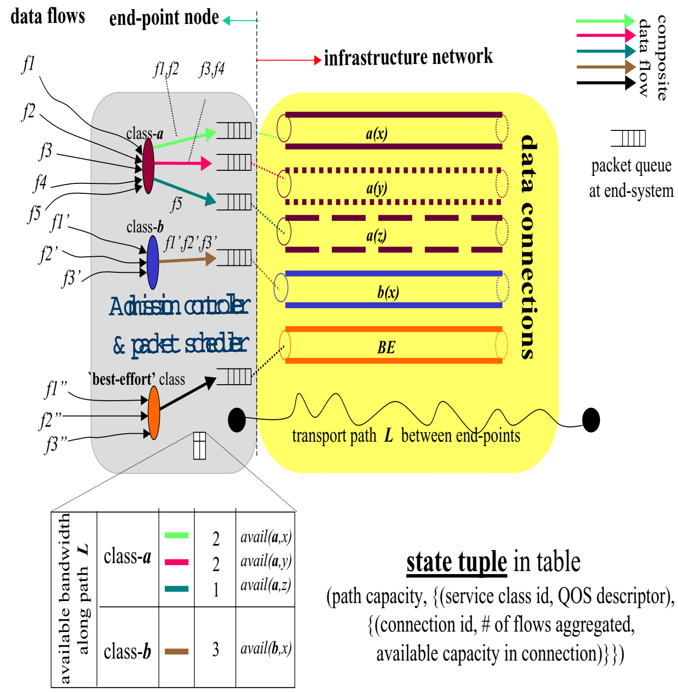


Fig. 3. State maintained at end-point nodes

number of flows aggregated over C. The only per-flow control activity incurred at the admission controller when flows are admitted or removed is to adjust the number of flows n and re-estimate the bandwidth needs using policy functions.

To enable the aggregation of data flows, the session-level manager may assign a unique label $l(C)$ to bind the component flows together, whereupon the admission controller can multiplex them over C. In other words, $l(C)$ is a session-level index to the grouping of data flows that are carried over C. In the example of sensor system, $l(C)$ may be the identifier used to refer to the external phenomenon from which the sensor data get generated in the application. The session-level labeling of connections can be part of a MPLS-based routing [8] over the path set up through the underlying network.

IV. Measurability of Bandwidth Gains

In this section, we describe how a measure of bandwidth gains can be incorporated in the end-point control of data flows over a transport path.

A. Why 'data connections' ??

The multiplexing gains arises from our ability to take advantage of traffic correlations that may exist between data flows at the application level. It allows determining a strategy for 'statistical sharing' of bandwidth, particularly, when the flows are bursty. For example, a strategy may be to allocate 75% of the peak bandwidth demands of the data flows. Can such strategies be effective without the notion of 'data connection' objects ? Our answer is NO.

Current models of QoS control are based on two session-level objects: 'data flows' and 'bandwidth guaranteed data paths' [9]. In contrast, our model stipulates another object, namely, 'data connection', to embody the grouping of closely-similar data flows. Referring to Figure 1, the 'data connections' L_x of 3 units bandwidth and L_y of 2 units bandwidth simply do not exist in the current models. Instead, only a single end-to-end path of 5 units bandwidth is visible to the session-level controller for multiplexing the various data flows.

Referring to Figure 3, an available bandwidth B along the end-to-end data path L can be shared between various data flows $f_1, f_2, \cdots, f_1', \cdots, f_3''$ by simply multiplexing them over L. No doubt, statistical multiplexing gains will accrue here also. However, when the data flows have diverse traffic characteristics, the bandwidth gains accrued therein may not be easily quantifiable. In this light, a 'data connection' offers the right abstraction, namely, the grouping of closely-similar flows, to enforce bandwidth allocation policies for the SP.

B. Determination of bandwidth savings

The bandwidth allocation over a shared connection satisfies weak additivity, indicated as:

$$\mathcal{F}(f_i)|_{i=1,2,\cdots,n} < \quad \mathcal{F}(f_1 \oplus f_2 \oplus \cdots \oplus f_n) \leq \mathcal{F}(f_1) + \mathcal{F}(f_2) + \cdots + \mathcal{F}(f_n),$$

where $\mathcal{F}(f_i) > A_i$. This relation captures the possible savings due to sharing of connection-level bandwidth across various flows, with the actual gains determined by the cross-correlation parameter associated with these flows. When there is no connection-level sharing, the inability to map the traffic correlation onto the packet scheduling exercised on various data flows forces the end-system to determine the bandwidth needs independently for each of the flows. So, the total allocation is $\sum_{i=1}^{n} \mathcal{F}(f_i)$. This in turn precludes bandwidth savings that may otherwise be feasible due to shared allocation driven by a traffic cross-correlation across the data flows (i.e., savings $=$ $\sum_{i=1}^{n} \mathcal{F}(f_i) - \mathcal{F}(\oplus\{f_1, \cdots, f_n\})$. Figure 4 illustrates the end-point admission control function.

C. Determination of policy functions

In a general form, the per-flow bandwidth allocation may be given as:

$$\mathcal{R}_{bw}(n) = \frac{\mathcal{F}(f_1 \oplus f_2 \oplus \cdots \oplus f_n)}{n},$$

with the monotonicity condition being: $\mathcal{R}_{bw}(n) < \mathcal{R}_{bw}(n')$ for $n > n'$. Figure 5 illustrates how a policy function $\mathcal{F}$ may capture these gains, so that it can be plugged in by the SP at appropriate control points[5]. The study is based on subjecting the packet flows generated from the video traffic traces of a *JurassicPark* movie segment to our policy-based bandwidth allocations. Three policies are employed: A, B and C — as

[5]It is not the mechanism of 'statistical multiplexing' that we focus in the paper. Rather, it is how we can quantitatively represent the policies that reap 'statistical multiplexing' gains, so that these gains can be factored into the flow admission decisions by the SP.

ADMIT_FLOW $(f_{N+1},\ q,\ C)$

additional bandwidth $X := F(q,N+1) - F(q,N)$;
if $[X < avail(C)]$
  admit new flow f_{N+1}
  $avail(C) := avail(C) - X$;
else
  reject f_{N+1}

q: flow descriptor
F : bandwidth estimator
b: estimated bandwidth need

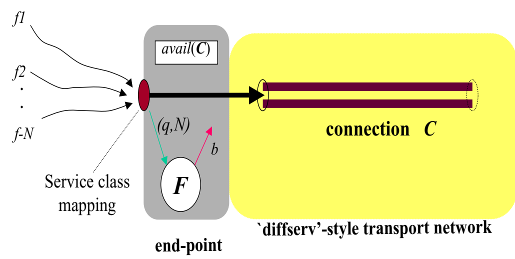
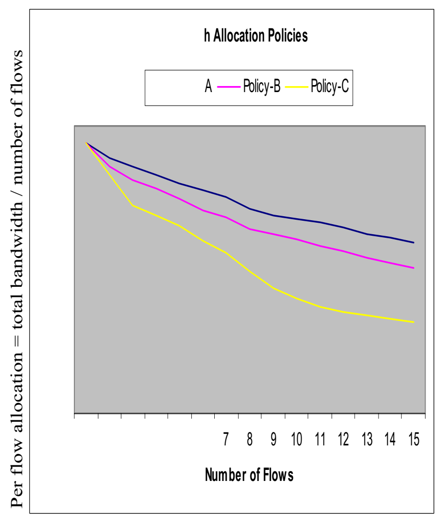


Fig. 4. Admission control on shared connection

per-flow bandwidth allocation (in *mbps*)

Number of Flow		Policy-B	Policy-C
1	3.3	3.3	3.3
2	3.1	3	2.9
3	3	2.85	2.54
4	2.9	2.73	2.41
5	2.8	2.62	2.29
6	2.71	2.48	2.1
7	2.63	2.38	1.95
8	2.5	2.25	1.72
9	2.41	2.18	1.53
10	2.37	2.12	1.41
11	2.33	2.04	1.29
12	2.26	1.97	1.24
13	2.19	1.9	1.2
14	2.14	1.83	1.15
15	2.08	1.77	1.11



policy C is more aggressive than policy B;
policy B is more aggressive than policy A

Fig. 5. Bandwidth allocation policies for shared connections

indicated in the tables. Policy C incurs the least amount of bandwidth allocation, policy B incurs the most, and policy B is in-between. The tables are pre-computed based on a traffic analysis of the traces, namely estimating the burstiness and average rate parameters from the packet size distributions.

To determine the bandwidth gains, the SP should be able to estimate the combined QoS parameters of the aggregated flows at end-system elements. Since a set of flows can be replaced by an equivalent combined flow for packet scheduling purposes, the parameters $(A, p, \Delta, \zeta, \mathcal{D})$ are definable at arbitrary flow granularities. We are interested in aggregating only the flows $\{f\}$ that are *closely-similar* in characteristics — such as the muliple video data flows generated from the *JurassicPark* movie segments in our study. An actual prescription of policy functions (such as the computation of allocation tables in Figure 5) is itself outside the scope of our paper.

In summary, flow aggregation allows a macroscopic and quantifiable management of connection-level data traffic for revenue-oriented decisions. We now describe the end-system control mechanisms for this management function.

V. End-system control mechanisms

The control mechanisms are built around 'packet scheduling' over data connections, weighted by their bandwidth allocations. We assume a weighted packet scheduling across connections to enforce connection-level bandwidth allocations (there is no per-flow state tracking at the infrastructure level)[6].

A. Packet delay checks

Packet-level delay checks are made against flow-specific delay tolerances. However, a 'connection' is the object granularity seen at the scheduler level. The scheduler may use the connection id (cid) carried in packets to index them into appropriate queues and exercise packet scheduling therefrom. Since only flows with similar characteristics are multiplexed over a connection, delay constraint checks at connection-level can provide information about packets meeting flow-specific delay tolerances. Note that an excessively delayed packet is deemed as a lost packet for end-to-end control purposes.

We have studied, by simulation, an agent-based implementation of the monitor for packet loss/delays. IETF RTCP is used for the agent-level signaling of packet loss information from the receiver to the source. Here, the congestion on a data connection C may arise because of a possible inability of the admission controller to determine the exact bandwidth needs for a set of flows multiplexed over C.

B. Optimal level of multiplexing

The multiplexing of data flows over a connection C is more susceptible to failures due to a possibility of excessive levels of path sharing and sustained higher rates in many of the data flows. Also, the intra-connection scheduling overhead on packets — which is another form of cost (besides bandwidth cost) — is higher with large number of flows multiplexed on a shared connection, to ensure that the acceptable loss rate and delay requirements are met. The work in [10] has shown that the queuing delay of packets is a monotonically increasing function of the number of flows n that feed packets into the queue. Figure 6 corroborates this delay behavior based on our simulation studies of policy functions A, B and C — c.f. Figure 5. Policy C incurs longer packet delays than policy B, and the latter incurs longer delays than policy A. Thus, beyond a certain level of sharing (say, for $n > n''$), the end-to-end delay of packets may increase to a level where the client-prescribed loss tolerance limits are not met.

[6]We assume FIFO based intra-connection scheduling across the component flows. This ensures the scalability of our mechanisms by avoiding the need for per-flow state.

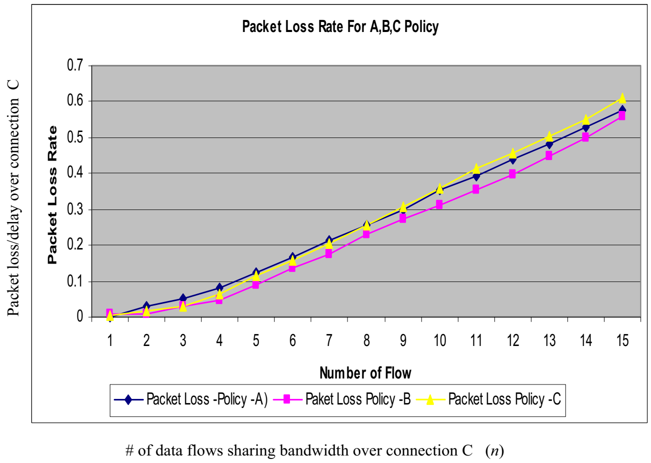


Fig. 6. Intra-connection scheduling costs

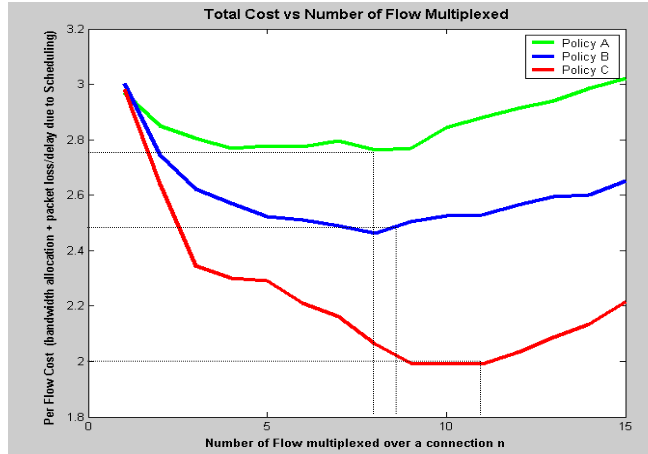


Fig. 7. Cost optimality behavior observed in our simulation study

A behavior similar to the non-guarantee of delay constraints holds for packet loss, represented as:

$$\mathcal{R}_{loss}(n) = \frac{\sum_{i=1}^{n} p_i - \mathcal{F}(f_1 \oplus f_2 \oplus \cdots \oplus f_n)}{n},$$

where p_i is the peak rate of flow f_i. The monotonicity condition is: $\mathcal{R}_{loss}(n) > \mathcal{R}_{loss}(n')$ for $n > n'$.

In general, the per-flow bandwidth cost $\mathcal{R}_{bw}(n)$ on a connection can be reduced by increasing the number of flows sharing this path. The lower bandwidth usage may however be counteracted by increased packet loss $\mathcal{R}_{loss}(n)$ arising from scheduling delays. Accordingly, the management module should ensure that the number of flows admitted into C does not exceed a threshold n_{opt} that may cause connection failures due to excessive packet loss. To determine this optimal point at run-time, the SP prescribes a cost function of the form:

$$\Theta(n) = a.\mathcal{R}_{bw}(n) + b.\mathcal{R}_{loss}(n)$$

for use by the SP, where a and b are normalization constants. There is a unique global optimal point n_{opt} for each allocation policy. Figure 7 shows this cost optimality behavior in our experimental study using video traffic traces under policies A, B and C. An aggessive policy has a higher n_{opt}, yielding a lower cost minimum — such as the policy-B over policy-A.

C. Checking for cost minimality

Since there is no closed-form analytical relation between $\Theta(n)$ and n for a connection C, the optimal value n_{opt} needs to be determined dynamically by measurements of packet loss experienced over C at run-time. That $\mathcal{R}_{bw}(n)$ and $\mathcal{R}_{loss}(n)$ exhibit monotonicity properties (as illustrated in Figures 5 and 6 respectively) ensures that the $\Theta(n)$-versus-n relation has a single global minimum, which in turn allows a measurement-based dynamic determination of n_{opt} by an iterative search process. For this purpose, we empirically identify how the multiplexing levels can change per-flow costs over C and how different policies can impact these costs.

Where there are multiple policy functions, empirically relating them allows the SP to dynamically switch from one policy to another. If $\mathcal{F}$ and $\mathcal{F}'$ depict optimistic policies such that $\mathcal{F}(f) > \mathcal{F}'(f)$ for some $f \in \mathcal{Q}$, then:

$$[\mathcal{F}(f_1 \oplus \cdots \oplus f_n \oplus f_{n+1}) - \mathcal{F}'(f_1 \oplus \cdots \oplus f_n \oplus f_{n+1})] > [\mathcal{F}(f_1 \oplus \cdots \oplus f_n) - \mathcal{F}'(f_1 \oplus \cdots \oplus f_n)]$$

for $n \geq 1$. The SP needs to determine the optimal multiplexing level n based on cost minimality considerations. Referring to Figure 7, the optimum n is 5 for policy A, 8 for policy B, and 11 for policy C. After such a determination of the optimal point, the set of connections required can be identified.

The SP needs to monitor the grouping of data flows at chosen points in time, and then estimate the service-level costs using the $\Theta(n)$ relation. The decision as to when a re-grouping of data flows into distinct connections should be undertaken may be based on a policy function[7] that interprets the variations in connection state n.

Our cost analysis is based on a relativistic measure of resource usage, rather than an absolute measure. This suffices to compare and evaluate policy functions on a relative scale.

VI. Related Works

There have been works that attempt to get the advantages of ”IntServ” world, namely, flexibility and fair QoS support and that of ”DiffServ” world, namely, robustness and scalability. We compare these works with our approach, with an emphasis on SP-level revenue incentives.

[12] proposes that packets carry the require per-flow state, instead of having the core network maintain the state. The approach requires the ingress and egress routers to manipulate a ’dynamic packet state’ (DPS), with the core routers functioning only based on packet classification and scheduling. [13] proposes a ’link-based fair aggregation’ (LBFA) technique that provides for class-based flow aggregation and fair queue

[7]Current methods to quantify network resource allocations (such as those described in [11]) can be incorporated in policy functions.

scheduling at the ingress and egress routers and for intra-class FIFO scheduling at the core routers. Both the DPS and LBFA techniques resort to the per-flow fair queuing in the edge routers as part of packet scheduling protocols. [9] provides an architecture in which end-points probe the network for bandwidth availability and admit a flow only when there is no congestion in the network. Though the architecture suggests a grouping of flows for admission control purposes, it argues against a fair queuing discipline for reasons of 'bandwidth stealing' by new flows from the flows already admitted.

In contrast to the above works, our connectivity model evaluates the fair queuing effects at a macroscopic level while consciously allowing statistical multiplexing among the flows sharing a bandwidth allocation. In other words, the end-point nodes in our model do not track the fairness in queuing at the flow granularity (despite flow-level traffic additions/removals to/from an end-point packet queue). Even when traffic policing is needed, the per-flow tracking is done by the end-system only in the event of congestion in the underlying transport connection. Overall, we take the SP's revenue incentives also into account when evaluating the per-flow guarantees.

Though our model also employs a a combination of the DiffServ and IntServ notions, our approach is different in that it anchors these architectural notions to the end-system operations — and not to the router operations. This is quite different from the context and scope of DiffServ and IntServ initially formulated by the IETF community and the subsequent implementation proposals [6], [7]. Furthermore, our approach allows finer levels of flow classification, as determined by applications. And, the flow classification need not be based only on closed-form traffic descriptions.

Besides the differences in architectural notions, our model of admission control allows incorporating statistical multiplexing gains as part of a cost assignment policy to application-level flows. Here, it is not that our model provides better statistical multiplexing gains compared to existing traffic management approaches (such as [14], [15]). It simply is that our model offers a better means of quantifying and estimating the bandwidth gains to enable revenue-oriented decision-making by connectivity SP's. We believe that bandwidth gains which cannot be quantified or measured are less meaningful for revenue-conscious SP's — however large the gains are. Our end-point architectural model reflects this paradigm of cost-effective connectivity provisioning.

VII. Conclusions

The paper described a new model of session-level connectivity provisioning for use by QoS-sensitive networked applications. The model is based on creating a variety of diffserv-type of 'data connections' with QoS differentiation and apportioning the available bandwidth across these connections using intserv-type of end-point admission control. The goal is a cost-effective provisioning of data connectivity.

The connectivity provider (SP) may employ policy functions to map the application-prescribed flow specs onto the resource needs of connections carrying data flows. The model allows dynamic switching from one policy function to another, based on a notion of cost associated with the infrastructure bandwidth usage, for a given level of QoS support. The strategy is to reduce the per-flow cost incurred by multiplexing many closely-similar data flows on a single connection. The multiplexing brings in two benefits to the SP, without compromising the QoS needs of applications. First, it reduces the per-flow resource allocation due to the gains accrued from a statistical sharing of connection resources. Second, it amortizes the connection-level overhead across many flows. The level of cost reduction, and hence the revenue accrual, can be controlled by the SP using a range of policy functions that take into account the burstiness and loss/delay tolerance of data flows.

Our model accommodates the above strategy through a management-oriented interface that allows the SP to maintain a repertoire of policy functions and choose one therefrom for providing an appropriate level of 'data connectivity' to the client applications. The paper described the functional mechanisms to monitor the end-to-end QoS and adjust the connection operating points to maximize the SP's revenue without compromising the user-level QoS needs. The studies indicate that our model can be employed in large network settings, alleviating scalability concerns.

References

[1] M. Subramanian. **Telecommunications Management Network**. Chap. 11, *Network Management: Principles and Practice*, Addison-Wesley Publ. Co., 2000.

[2] S. Keshav. **Scheduling**. In Chap. 9, *An Engineering Approach to Computer Networking*, Addison-Wesley Publ. Co., pp.209-260, 1996.

[3] J. Wroclawski. **Specification of the Controlled-load Network Element Service**. In *Internet RFC 2211*, 1997.

[4] S. Floyd and V. Jacobson. **Link-sharing and Resource Management Models for Packet Networks**. In *IEEE/ACM Transactions on Networking*, vol.3, no.4, Aug. 1995.

[5] C. Dovrolis and P. Ramanathan. **Proportional Differentiated Services, Part-II: Loss Rate Differentiation and Packet Dropping**. In *Intl. Workshop on Quality of Service*, IWQoS'00, Pittsburgh (PA), June 2000.

[6] S. Berson and S. Vincent. **Aggregation Internet Integrated Services State**. In *Internet RFC*, 1997.

[7] K. Nichols, V, Jacobsen, and L. Zhang. **Two-bit Differentiated Services Architecture for Internet**. In *Internet RFC 2638*, 1999.

[8] U. Black. **MPLS and Label Switching Networks**. 2nd Ed., Prentice-Hall Publ. Co., 2002.

[9] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang. **End-point Admission Control: Architectural Issues and Performance**. in proc. *ACM SIGCOMM'00*, Stockholm, 2000.

[10] R. Guerin and A. Orda. **QoS Routing in Networks with Inaccurate Information: Theory and Algorithms**. In *IEEE/ACM Transactions on Networking*, vol.7, no.3, June 1999.

[11] H. M. Mason and H. R. Varian. **Pricing Congestible Network Resources**. In *IEEE Journal on Selected Areas in Communications*, vol.13, no.7, pp.1141-1149, Sept. 1995.

[12] I. Stoica and H. Zhang. **Providing Guaranteed Services Without Per-flow Management**. In proc. ACM SIGCOMM'99, Cambridge (MA), Aug. 1999.

[13] Y. Jiang. **Link-based Fair Aggregation: a Simple Approach to Scalable Support of Per-Flow Service Guarantees**. in *Tech. Report*, Norwegian Inst. of Technology, 2004.

[14] D. N. C. Tse, R. G. Gallagher, and J. N. Tsitsiklis. **Statistical Multiplexing of Multiple Time-scale Markov Streams**. *IEEE Journal on Sel. Areas in Comm.*, vol.13, no.6, pp.1028-1038, Aug. 1995.

[15] L. Boroczky, A. Y. Ngai, and E. C. Westermann. **Statistical Multiplexing Using MPEG-2 Video Encoders**. *IBM Tech. Journal*, vol.43, no.4, 1999.