Available online at www.sciencedirect.com
SCIENCE DIRECT
Computer Communications 28 (2005) 2070–2081
computer communications
www.elsevier.com/locate/comcom

Capacity requirements of traffic handling schemes in multi-service networks

Towela P.R. Nyirenda-Jere[a], Victor S. Frost[a,*], Nail Akar[b]

[a]Department of Electrical Engineering and Computer Science, Information and Telecommunication Technology Center, University of Kansas, 2335 Irving Hill Road, Lawrence, KS 66045-7612, USA
[b]Department of Electrical and Electronics Engineering, Bilkent University, Ankara, Turkey

Received 29 July 2004; accepted 29 July 2004
Available online 17 September 2004

Abstract

This paper deals with the impact of traffic handling mechanisms on capacity for different network architectures. Three traffic handling models are considered: per-flow, class-based and best-effort (BE). These models can be used to meet service guarantees, the major differences being in their complexity of implementations and in the quantity of network resources that must be provided. In this study, the performance is fixed and the required capacity determined for various combinations of traffic handling architectures for edge-core networks. This study provides a comparison of different QoS architectures. One key result of this work is that on the basis of capacity requirements, there is no significant difference between semi-aggregate traffic handling and per-flow traffic handling. However, best-effort handling requires significantly more capacity as compared to the other methods.

Keywords: Network engineering; Internet QoS; Multimedia networking; Network capacity planning

1. Introduction

The Internet has experienced tremendous growth in both volume and diversity of carried traffic. The engineering philosophy of the Internet was based on the model of a homogeneous community that had common interests [22]. A major tool that was often used to engineer the Internet was over-engineering (often called 'throwing bandwidth at the problem') which refers to providing more bandwidth than the aggregate demand. The recent growth in network traffic coupled with advances in high-speed applications, however, tends to stretch the limits of over-booking. Simultaneous uses have increasing expectations on the service that they receive. Applications with diverse throughput, loss and delay requirements require a network that is capable of supporting different levels of service and this requires changes to network control and traffic handling functions. Control mechanisms allow the user and network to agree on service definitions, identify users that are eligible for a particular type of service and let the network allocate resources appropriately to the different services. Traffic handling mechanisms are used to classify and map packets to the intended service class as well as to control the resources consumed by each class [32,34]. Notable results of the effort to provide Quality of Service (QoS) in the Internet are the definition of Integrated Services and Differentiated Services by the Internet Engineering Task Force (IETF) [2–5,10,19,20] and Asynchronous Transfer Mode (ATM) [1]. There is a trade-off between efficiency in the use of network resources and strictness of QoS guarantees. The simplest and least efficient strategy is to perform no data handling and no signaling while over-provisioning the network. This is the best-effort (BE) approach. On the other end of the scale is the per-flow approach with per-flow signaling and data handling. An intermediate approach is the semi-aggregate, class-based approach that has been proposed for the Differentiated Services (Diffserv) model. An integrated network must

* Corresponding author. Tel.: +90 785 864 4833; fax: +90 785 864 0387.
E-mail address: frost@eecs.ku.edu (V.S. Frost).

0140-3664/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2004.07.035

balance the trade-off between performance, complexity and flexibility. There is also a trade-off to be made between the cost of bandwidth and the cost of extra mechanisms to provide QoS. In one model per-flow traffic handling mechanisms are used in the edge of the network and aggregate mechanisms such as those proposed in the Diffserv model are used in the core [16,26,33,37]. The result is an edge-core network architecture. Research efforts in the last several years have produced considerable literature on how the Internet can be re-engineered and redesigned to provide QoS, for example:

- The question of whether the Internet should retain a BE only architecture or adopt an architecture that supports reservations is addressed in [7]. They consider the incremental bandwidth that is required to make a BE network perform as well as a reservation capable network. Their results indicate that the incremental bandwidth depends on whether the applications are adaptive or non-adaptive. Adaptive applications require less incremental bandwidth. The general conclusion is that providing a definite answer to the choice between reservation and BE depends on the load patterns.
- The work in [14] addresses the issue of levels of aggregation and the main conclusion is that the best QoS is achieved when flows with identical characteristics are aggregated and the division of traffic into two classes, a Real-Time class and a non-Real-Time class, is adequate to meet the stringent delay QoS requirements of the audio and video.
- In [15], the authors compare the 'fat dumb pipe' (best-effort) model with a differentiated services model. The BE model uses over-provisioning to achieve QoS. They differentiate between absolute service differentiation in which the network tries to meet the same goals as the Intserv network but without the use of per-flow state and relative service differentiation in which assurances are based on a relative ordering of each application and its requirements.
- In [21], the authors compare the use of flow aggregation with no aggregation for provision of QoS guarantees. They find that flow aggregation systems require more bandwidth than those with no aggregation but that systems with no aggregation are more complex to administer. They also note that the benefits of aggregation increase with the number of flows and as the number of flows increases, the bandwidth required by the aggregate systems approaches that of the non-aggregation system.
- In [25], a comparison is made between class-level and path-level aggregation. In class-level aggregation, flows belonging to the same class are queued together and a jitter controller (regulator) is used to ensure that all flows within a class experience the same (maximum) delay. In path-level aggregation, flows which share the same end-to-end path are queued together. They find that the multiplexing gain for class-level aggregation is higher but the use of the jitter controller results in increased delays and requires more buffering. They conclude that the better multiplexing gain with the class-level approach is worth the increased delays due to the jitter control.
- In [31], the efficiency due to flow grouping is analyzed based on the observation that aggregation of flows inside the core network will resolve the scalability issues associated with handling numerous flows individually inside the core. Two aspects to aggregation are considered: how should resources be allocated to aggregated flows and which flows should be grouped together. The analysis shows that for homogeneous flows, aggregation requires less resources than handling flows individually. For flows that have different rate and burstyness parameters and the same delay requirements, aggregation requires more resources.

Providing QoS in the Internet requires providers to re-evaluate the mechanisms that are used for traffic engineering and management. Over-engineering is an attractive option because it is a simple solution [18]. Recent proposals are calling for more active traffic management in the Internet that will result in more efficient use of resources while allowing providers to offer varying levels of service. These range from simple admission policies to complex queuing and scheduling mechanisms within routers and switches. There are several alternatives for providing QoS. These are: (1) inefficient use of network bandwidth with no traffic management; (2) moderately efficient use of network bandwidth with simple traffic management; (3) efficient use of network bandwidth with complex traffic management. The issue of finding the network capacity at which these approaches to traffic management become equivalent is considered here. Specifically, we determine the network capacity required to achieve equivalent levels of performance under a variety of traffic management schemes. This paper is organized as follows. In Section 2, we provide an overview of an analytic method to determine worst-case bounds on end-to-end delays in networks. In Section 3, we then show how this analysis can be used to compare the capacity requirements of different traffic handling mechanisms. Section 4 provides numerical results and conclusions are presented in Section 5.

2. Edge-core network capacity analysis using network calculus

The network calculus is deterministic and does not depend on probabilistic descriptions of traffic. For most of the research discussed in Section 1, the results depended on the assumed traffic models. Network calculus is used with envelope bounded traffic models to provide worst-case analysis on network performance. We have chosen this method because it allows us to obtain results that can be

applied to any type of traffic provided we can bound the traffic process at the input to the network. This method is especially appealing since many services defined by the ATM Forum and the IETF are based on traffic that is regulated before it enters the network [1,19]. One of the concerns with the network calculus approach is that it provides worst-case bounds on end-to-end delay, which may underestimate the utilization. Since we are primarily concerned with the ratio of the performance between the different QoS architectures, we do not expect this to have a significant impact on the conclusions. Network calculus has been applied to a variety of network problems [8,11–13,17,24,28,29,30]. Most of the literature focuses on simple network topologies with at most two classes of service. Here we apply this approach to large networks carrying a diverse mix of traffic as well as in addressing an aspect of network performance evaluation that has received limited attention. Thus the results presented here are not sensitive to specific traffic assumptions. Further network calculus also provides a framework to obtain the relative magnitude of the capacity requirements for the different edge-core architectures; the precise determination of required capacity is not the objective, rather the analysis is aimed at determining the relative scale of the required network capacity. This work shows that there is no significant difference in total network capacity for architectures using semi-aggregate traffic handling and per-flow traffic handling. However, BE handling requires several orders of magnitude greater capacity as compared to the other options. Previous analysis has shown that with all other parameters constant, traffic burstyness grows exponentially with the number of network hops. However, in this analysis, the capacity is allowed to change as necessary to keep the performance, i.e. delay, constant. This analysis shows that in this case, the growth in capacity and burstyness is linear. Allowing the capacity to change while fixing the number of hops changes the increase in burstyness from exponential to linear.

2.1. Framework

We use two classes of traffic with voice and video belonging to the Real-Time (RT) traffic class and email and WWW traffic belonging to the Non-Real-Time (NRT) traffic class. These applications are representative of current network usage and they provide diversity in their attributes and QoS. The QoS metric that is used here is end-to-end delay. We recognize that email and WWW traffic are considered to be adaptive applications that do not have strict delay requirements. The delay objectives for email and WWW used here are an order of magnitude higher than those of voice and video to reflect the fact that while they may be adaptive, users of email and WWW applications have certain expectations on delay and may require bandwidth guarantees to prevent starvation. For characterization of the traffic sources, we used the burstyness constraint model of Cruz [11] in which traffic is characterized by two parameters, a burstyness parameter σ and an average rate parameter ρ. In this analysis, we assume that the average rate on any link is less than its capacity. We assume that the network uses regulator elements to ensure that the traffic entering it conforms to these parameters. The IETF and ATM Forum have defined network elements which can convert an arbitrary traffic process into a process that is bounded in this way [1,19]. The QoS metric that we use is end-to-end queuing delay. The parameters for each traffic type are shown in Table 1. The rate and packet size parameters are based on values quoted in the literature. The burst parameters were chosen based on the literature and our own judgment. We classified traffic handling mechanisms as simple, intermediate and complex depending on whether they are used for total aggregation (BE), semi-aggregation (class-based) or per-flow handling, respectively. We identified four candidate traffic handling mechanisms: First-In-First-Out (FIFO) handling for BE handling, Class-based Queuing (CBQ) and Priority Queuing (PQ) for semi-aggregate handling and Weighted Fair Queueing (WFQ) for per-flow handling. In WFQ, each flow is assigned its own guaranteed rate. The CBQ and PQ schedulers have two classes: a Real-Time (RT) class for voice and video and a non-Real-Time (NRT) class for email and www traffic. In CBQ, each class is assigned a guaranteed rate, while for PQ there is no guaranteed bandwidth and service is strictly based on priority with the RT class having highest priority. In FIFO, all flows share the same queue and there is also no per-flow guaranteed bandwidth. We use the subscript k to refer to a traffic type such as voice or video and the subscript p to refer to a traffic class or priority level. In addition, the subscript 'class p' is used to refer to a traffic class without reference to the traffic type. Using delay as an example, D_k is the delay of traffic of type k, $D_{k,p}$ is the delay of traffic of type k when it is assigned to class p and $D_{\text{class } p}$ is the delay of class p. The superscript (m) refers to a node in the network and parameters for an aggregation of flows have a bar over them. Using this notation and with burstyness as an example, $\sigma_k^{(m)}$ would refer to the burstyness of a flow of type k at node m, $\bar{\sigma_k}^{(m)}$ would refer to the aggregate burstyness of flows of type k at node m and $\bar{\sigma}^{(m)}$ would refer to the aggregate burstyness of all flows at node m. For each scheme, the maximum end-to-end delay will depend on the traffic handling scheme. Let $D_{k,p}^X$ be the per-node delay for traffic of type k when it is assigned to class p using scheme X. For WFQ, each flow constitutes a class while with FIFO

Table 1
Traffic class parameters

Type	Rate, ρ (Mbps)	Burstyness, σ (Bytes)	Packet size (Bytes)	Delay (ms)
Voice	0.064	64	64	20
Video	1.5	8000	512	50
Email	0.128	3072	512	500
WWW	1.0	40,960	1500	500

there is only one class so that the delay for each flow will be the minimum over all specified delays. With CBQ and PQ, we have two classes and the delay seen by a given traffic flow will be the minimum over all traffic flows in its class. Thus we have:

$$D_k^{\mathrm{WFQ}} = D_k, \quad \forall\, k, \quad D_k^{\mathrm{FIFO}} = \min_k\{D_k\} = D_{\min},$$
$$\forall\, k, \quad D_{k,p}^{\mathrm{CBQ/PQ}} = \min_{j\in \text{class } p}\{D_j\}$$

2.2. End-to-end delay

Here network calculus is used to calculate maximum end-to-end delay. The general approach to computing end-to-end delays in a multi-node network is to sum the individual delays at each node. For a class of servers known as Latency-Rate (LR) servers [35] which provide a guaranteed rate to each connection and which can offer a bounded delay, a tighter bound can be obtained by using the 'pay bursts once' principle [23]. The pay bursts once principle uses the approach of considering the entire network as a whole rather than looking at individual network elements in isolation. Using the pay bursts once principle for a network of LR-servers, the end-to-end delay for a flow k over a network of M servers in series is given by [35]

$$D_{\mathrm{E2E}_k} = \frac{\sigma_k}{\min_m(g_k^{(m)})} + \sum_{m=1}^{M}\theta_k^{(m)} \tag{1}$$

where $\theta_k^{(m)}$ is the latency experienced by connection k at server m and $g_k^{(m)}$ is the guaranteed rate for connection k at server m. Examples of LR schedulers are Generalized Processor Sharing, Weighted Fair Queueing, Self-Clocked Fair Queueing, Weighted Round Robin. For a Generalized Processor Sharing (GPS) scheduler, the delay is $\theta_k^{(m)} = L_{\max}/C$ while for Packet Generalized Processor Sharing (Weighted Fair Queueing) the delay is [28]:

$$\theta_k^{(m)} = \frac{L_{\max}}{C^{(m)}} + \frac{L_k}{g_k^{(m)}} \tag{2}$$

where L_k is the maximum packet size for connection k, $L_{\max}$ is the maximum packet size in the network and $C^{(m)}$ is the capacity of the link at the mth server.

The end-to-end delay with WFQ is thus:[1]

$$D_{\mathrm{E2E}_k}^{\mathrm{WFQ}} = \frac{\sigma_k}{\min_m(g_k^{(m)})} + \sum_{m=1}^{M}\left(\frac{L_k}{g_k^{(m)}} + \frac{L_{\max}}{C^{(m)}}\right) \tag{3}$$

For non-LR schedulers, the end-to-end delay is given by

$$D_{\mathrm{E2E}_k} = \sum_{m=1}^{M}\theta_k^{(m)} \tag{4}$$

[1] Tighter bounds can be obtained by omitting the factor L_k/g_k in the last node. Refer to [29] for details.

where $\theta_k^{(m)}$ is the maximum delay for connection k at server m. For a FIFO queue, $\theta_k^{(m)} = \theta^{(m)}$ and

$$\theta^{(m)} = \frac{\bar{\sigma}^{(m)}}{C^{(m)}}, \quad \bar{\sigma}^{(m)} = \sum_{k\in N(m)}\sigma_k^{(m)}$$

where $\sigma_k^{(m)}$ is the accumulated burstyness of a flow of type k at node m, $\bar{\sigma}^{(m)}$ is the aggregate burstyness of all flows at node m, $C^{(m)}$ is the link capacity at node m and $N(m)$ is the set of connections that flow through server m. Thus for a series of M FIFO servers, we have the end-to-end delay given by:

$$D_{\mathrm{E2E}}^{\mathrm{FIFO}} = \sum_{m=1}^{M}\frac{\bar{\sigma}^{(m)}}{C^{(m)}} \tag{5}$$

For a static priority system (PQ), the latency experienced by a connection of priority p at node m is

$$\theta_p^{(m)} = \frac{\sigma_H^{(m)}(p) + L_{\max}(p)}{C^{(m)} - \rho_H^{(m)}(p)}$$

where $C^{(m)}$ is the link capacity at node m and:

$$\rho_H(p)^{(m)} = \sum_{j=1}^{p-1}\rho_j^{(m)}, \quad \sigma_H(p)^{(m)} = \sum_{j=1}^{p}\sigma_j^{(m)},$$

$$L_{\max}(p) = \max_{j\geq p}\{L_j\}$$

In the above equations $\rho_j^{(m)}$ is the aggregate average rate of priority level j, $\sigma_j^{(m)}$ is the aggregate burstyness and L_j is the maximum packet size for priority j. Thus, for a series network of M static priority servers, we have the end-to-end delay for traffic of priority level p given by:

$$D_{\mathrm{E2E}_p}^{\mathrm{PQ}} = \sum_{m=1}^{M}\frac{\sigma_H^{(m)}(p) + L_{\max}(p)}{C^{(m)} - \rho_H^{(m)}(p)} \tag{6}$$

For a Class-Based Queueing system with P classes, we assume separate FIFO queues for each class with WFQ service between the queues so that each queue p gets a guaranteed rate g_p. If we assume that the network routing is such that the set of flows in a class share the same path end-to-end and do not merge with other flows (even if they are of the same class), then we can use the pay bursts once approach to calculate the end-to-end delays as was done for WFQ. This type of flow grouping has been referred to as path-level aggregation in [25] and has also been studied in [31]. Here the more general case where flows in a class do not share the same end-to-end path so that the composition of flows belonging to a class varies at each node is considered. The latency at the mth node is

$$\theta_p^{(m)} = \frac{\sigma_p^{(m)} + L_p^{(m)}}{g_p^{(m)}} + \frac{L_{\max}}{C^{(m)}}$$

where $\sigma_p^{(m)}$ is the aggregate burstyness for class p, $g_p^{(m)}$ is the guaranteed rate for class p, L_p is the maximum packet size for flows in class p, $L_{\max}$ is the maximum packet size over

all flows and $C^{(m)}$ is the capacity of the link at node m. Then the end-to-end delay for connections of class p going through a series of M CBQ servers is given by:

$$D_{\mathrm{E2E}_p}^{\mathrm{CBQ}} = \sum_{m=1}^{M} \frac{\sigma_p^{(m)} + L_p^{(m)}}{g_p^{(m)}} + \frac{L_{\max}}{C^{(m)}} \tag{7}$$

Next the capacity requirements for edge-core networks for fixed delays is calculated.

2.3. Comparison of capacity requirements

We consider a network architecture that has two distinct hierarchical layers—an edge and a core—as shown in Fig. 1. The core is the backbone of the network and consists of high-speed elements. The edge portion of the network provides access to the core network and serves to aggregate traffic into the network core. The parameters and notation we use to describe a topology are:

- K, number of traffic types
- P, number of classes or priority levels
- N_{core}, number of core nodes
- N_{edge}, number of edge nodes per core node
- C_{edge}, reference capacity of edge links (assumed here to be homogeneous across the network)
- *Core Connectivity Matrix A*. This matrix describes the way in which the core nodes are connected. For N_{core} core nodes, the matrix is $N_{\mathrm{core}} \times N_{\mathrm{core}}$ and has elements a_{ij} where $a_{ij} = 1$ if a link exists from core node i to core node j and 0 otherwise. Note that links are unidirectional.

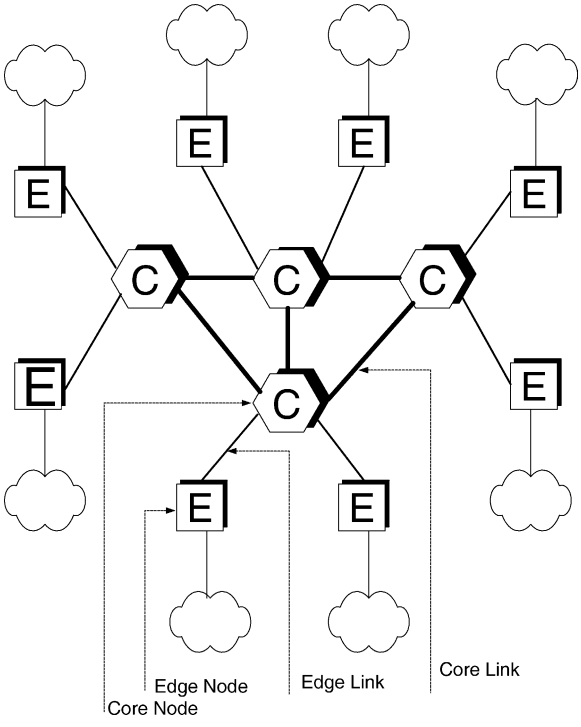

Fig. 1. Network topology.

- M_{T}, maximum number of nodes traversed by any flow in the network
- M, maximum number of core nodes traversed by any flow in the network
- D_k^{E2E}, maximum end-to-end delay for traffic of type k
- D_k, maximum delay per node for traffic of type k. We assume that the delays are uniformly distributed over the nodes to give:

$$D_k = \frac{D_k^{\mathrm{E2E}}}{M_{\mathrm{T}}}$$

- *Core Traffic Distribution Matrix* T_k. This matrix describes how the traffic belonging to a particular type k, incident at a core node is distributed to the destination core nodes in the network. There will be one matrix for each traffic class and each matrix will be of size $N_{\mathrm{core}} \times N_{\mathrm{core}}$. The elements of the matrix will be fractions of the per-class traffic destined for a specific core node. For example, let $N_{\mathrm{core}} = 3$ and suppose τ_k is the distribution matrix for voice traffic. Then $\tau_k^{12} = 0.5$ means 50% of voice traffic incident at core node 1 is destined for core node 2, etc.
- *Traffic allocation*. The traffic allocation for each traffic type denoted w_k is specified with respect to each edge link so that it measures how much bandwidth on a single edge-link has been allocated to a specific type. More precisely, the allocation is the fraction of the edge link capacity assigned with $0 \le w_k < 1$ and $\sum_k w_k < 1$. It is used together with the guaranteed rate to determine how many sources of each type may be supported on one edge link. The total allocation is denoted w_{T} and is equal to $\sum w_k$.

Other parameters used are:

- $D_{k,p}$, maximum delay of traffic of type k when it is assigned to class p
- $D_{\mathrm{class}\ p}$, delay of traffic in class p. We use this notation when we do not need specific reference to the traffic type.
- $D_{\min}$, minimum delay in the network given by $D_{\min} = \min_k\{D_k\}$

In performing the analysis, we can either study each node separately and accumulate delays [11,12] or we can analyze the end-to-end path as a whole [28,29]. The first approach can lead to very loose bounds on delay while the second approach is applicable only when all the nodes in the network are rate-guaranteeing and a minimum guaranteed rate can be identified. Since we are using traffic handling schemes that vary in their ability to provide rate guarantees, the analysis of the end-to-end path will be done in the following manner [6,38,39]: (1) divide the path into regions containing rate-guaranteeing and non-rate-guaranteeing segments; (2) analyze the non-rate-guaranteeing segments by accumulating delays additively;

(3) analyze the rate-guaranteeing segments using the path approach; (4) sum the delays in the rate-guaranteeing and non-rate-guaranteeing segments to obtain the end-to-end delays.

Since the goal is to compare the network capacity required by different QoS architectures, we must ensure that there is a uniform basis for comparison. We use a network with WFQ in both the edge and core as the reference. We will refer to this network as the reference network and use the notation WFQ* to distinguish it from other networks. The approach is thus to calculate the amount of traffic that can be supported in a WFQ* network and compare the capacity required to support the same traffic for various combinations of traffic handling schemes in the edge and core.

3. Analysis

For rate-guaranteeing schemes, there are two possibilities for bandwidth allocation in the network: core nodes allocate the same bandwidth as edge nodes to each connection or core nodes allocate more bandwidth than edge nodes. We will use the first approach for simplicity which will mean that the delays in the core are the same as in the edge. The second approach may be more useful when optimizing the delay budget since we can reduce the delays in the core by allocating more bandwidth. Using Eq. (3), we calculate the guaranteed rate $g_k^{\mathrm{WFQ*}}$ as

$$g_k^{\mathrm{WFQ*}} = \max\left\{\rho_k, \frac{\frac{\sigma_k}{M_{\mathrm{T}}} + L_k}{D_k}\right\} \tag{8}$$

where we have assumed that the factor $(L_{\max}/C)$ in Eq. (3) is negligible. The number of sources of each type that can be supported at an edge node using WFQ* is then given by

$$N_k = \left\lfloor \frac{w_k C_{\mathrm{edge}}}{g_k^{\mathrm{WFQ*}}} \right\rfloor \tag{9}$$

where $\lfloor x \rfloor$ is x rounded down, C_{edge} is the edge link capacity and w_k is the proportion of capacity on the edge link allocated to traffic of class k. Next the edge and core capacity required to support this traffic is determined.

For WFQ, the edge capacity is given by:

$$C_{\mathrm{edge}}^{\mathrm{WFQ*}} = \sum_{k=1}^{K} N_k g_k^{\mathrm{WFQ*}} = \sum_{k=1}^{K} w_k C_{\mathrm{edge}} \tag{10}$$

For CBQ and PQ with P classes/levels we have:

$$C_{\mathrm{edge}}^{\mathrm{CBQ}} = \sum_{p=1}^{P} \sum_{k \in p} \frac{N_k \sigma_k + L_p}{D_{\mathrm{class}\ \ p}}, \quad p = 1, 2, \ldots, P \tag{11}$$

$$C_{\mathrm{edge}}^{\mathrm{PQ}} = \max_{p=1,\ldots,P} \left\{ \sum_{j=1}^{p} \sum_{k \in \mathrm{class}\ \ j} \frac{N_k \sigma_k + L_{\max}(p)}{D_{\mathrm{class}\ \ p}} + \sum_{j=1}^{p-1} \sum_{k \in \mathrm{class}\ \ j} N_k \rho_k \right\} \tag{12}$$

For FIFO, the capacity is given by:

$$C_{\mathrm{edge}}^{\mathrm{FIFO}} = \sum_{k=1}^{K} \frac{N_k \sigma_k}{D_{\min}} \tag{13}$$

The capacity required in the core for each scheme has the same general form regardless of the mechanism in the edge. The key differentiating factor is the change in burstyness for a given traffic type induced by the delay in the edge. Since edge capacities are calculated to meet the prescribed QoS objectives for each class, we can assume that the edge delay will be bounded by the per-node maximum delay D_k. For a WFQ core, using the traffic distribution matrices T^k, the minimum required capacity on the link $l(i,j)$ between the core node-pair (i,j) is given by

$$C_{\mathrm{core}(i,j)}^{\mathrm{WFQ}} = N_{\mathrm{edge}} \sum_{k=1}^{K} \sum_{(x,y)} \tau_k^{(x,y)} N_k^x g_k, \quad \{(x,y) : l(i,j) \in \mathrm{Path}(x,y)\} \tag{14}$$

where $\tau_k^{(x,y)}$ represents the distribution factors of flows between core nodes (x,y) whose path $\mathrm{Path}(x,y)$ includes the link $l(i,j)$ and N_k^x is the number of sources of class k whose edge node is attached to core node x. The value of g_k will depend on the traffic handling in the edge: when the edge uses WFQ, g_k will be the same as $g_k^{\mathrm{WFQ*}}$ in Eq. (8) and when the edge uses any other mechanism considered here, it will be

$$g_k = \max\left\{\rho_k, \frac{\frac{\sigma'_k}{M} + L_k}{D_k}\right\} \tag{15}$$

where M is the number of core nodes and σ'_k is the burstyness after passing through the edge portion of the network which is given by:

$$\sigma'_k = \sigma_k + \rho_k D_k^{\mathrm{edge}},$$

$$D_k^{\mathrm{edge}} = \begin{cases} D_k, & \text{for WFQ} \\ \min_{j \in \mathrm{class}\ \ p}\{D_j\} k \in \mathrm{class}\ p, & \text{for CBQ, PQ} \\ \min_k\{D_k\}, & \text{for FIFO} \end{cases}$$

To calculate the core capacity with CBQ, PQ and FIFO, for each link $l(i,j)$ let

$$\bar{\rho}_k^{(i,j)} = \sum_{(x,y)} \tau_k^{(x,y)} N_k^x \rho_k \tag{16}$$

$$\bar{\sigma}_k^{(i,j)} = \sum_{(x,y)} \tau_k^{(x,y)} N_k^x \sigma_k^{h(x,y)} \tag{17}$$

where $\{(x,y)$: $l(i,j) \in \text{Path}(x,y)\}$. Note that $h(x,y)$ is the number of hops traversed by the traffic before reaching link $l(i,j)$ and $\sigma_k^{h(x,y)}$ is the associated burstyness which will depend in part on the traffic handling mechanism used in the edge. To be more specific, $\sigma_k^{h(x,y)}$ will be given by

$$\sigma_k^{h(x,y)} = \begin{cases} \sigma'_k + h(x,y)\rho_k D_{\text{class}\ \ p}, & \forall\, k \in p, \quad \text{for CBQ/PQ} \\ \sigma'_k + h(x,y)\rho_k D_{\min}, & \text{for FIFO} \end{cases}$$

with σ'_k defined as before.

Then by inverting Eqs. (5)–(7), the core bandwidth required on link $l(i,j)$ for each scheme is calculated as follows

$$C_{\text{core}(i,j)}^{\text{CBQ}} = N_{\text{edge}} \sum_{p=1}^{P} \sum_{k \in p} \frac{\bar{\sigma_k}^{(i,j)} + L_p}{D_{\text{class}\ \ p}} \tag{18}$$

$$C_{\text{core}(i,j)}^{\text{PQ}} = N_{\text{edge}} \max_{p=1,\ldots,P} \left\{ \sum_{m-1}^{p} \sum_{k \in \text{class}\ m} \left(\frac{\bar{\sigma_k}^{(i,j)}}{D_{\text{class}\ p}} + \frac{L_{\max}(p)}{D_{\text{class}\ p}} \right) + \sum_{m=1}^{p-1} \sum_{k \in \text{class}\ m} \bar{\rho}_{(i,j)}^{k} \right\} \tag{19}$$

$$C_{\text{core}(i,j)}^{\text{FIFO}} = N_{\text{edge}} \sum_{K=1}^{k} \frac{\bar{\sigma}_k^{(i,j)}}{D_{\min}} \tag{20}$$

where we again assume the term $(L_{\max}/C)$ in Eq. (7) is negligible. This analysis provides a common framework to compare the capacity requirements of different edge-core architectures given constant performance, in the case a delay bound. Additional details of the analysis can be found in [27].

4. Results

4.1. Topology construction

Network topologies were constructed by varying the number of core nodes and the number of core links per node. For a given core node value N_{core}, the number of links per core node n_{link} was varied according to:

$$n_{\text{link}} = 2, 3, \ldots, N_{\text{core}} - 1$$

This resulted in $(N_{\text{core}} - 2)$ different topologies per core node value. Note that the case when the number of links per core node is equal to $(N_{\text{core}} - 1)$ is a full-mesh topology. For each topology, we used the same external load on the core network by fixing the total number of edge nodes $N_{\text{edge}}^{\text{total}}$ and setting the number of edge nodes per core node to $N_{\text{edge}} = N_{\text{edge}}^{\text{total}}/N_{\text{core}}$. We used a value of $N_{\text{edge}}^{\text{total}} = 60$. Once a topology had been constructed, routes were set up within the core using Dijkstra's shortest path algorithm [36]. Traffic within the core was distributed symmetrically[2] so that each core node sent an equal amount of traffic to every other core node and the traffic distribution matrix was specified by:

$$T_{ij} = \begin{cases} \dfrac{1}{(N_{\text{core}} - 1)}, & i \neq j \\ 0, & i = j \end{cases}$$

We used a maximum load on each edge link (w_{T}) of 90%. The capacity required in the edge and core for all possible combinations of edge-core traffic handling schemes was calculated for different topologies with the number of core nodes ranging from 3 to 20. All of the results presented here are based on the analyses given in the previous sections (no discrete event simulation was employed). We present our results in the form of bar graphs that show the mean core link capacity stacked on top of the mean edge capacity to better illustrate the differences between the four schemes. To distinguish between the edge and core results, the core capacity is shaded according to the value of voice load. The bar-graphs are plotted on two separate y-axes to allow for better observation of the WFQ, CBQ and PQ results.

4.2. Capacity requirements with symmetric traffic distribution

The purpose of this analysis was to determine how the bandwidth requirements of the four schemes are affected by changes in the voice load on the edge links. We used three values of video load (0,0.1,0.2) and in one case we varied the voice load while in the other case we varied the WWW load within the limits of the maximum load w_{T}. The remaining edge bandwidth was equally divided between email and WWW for the first case and email and voice for the second. For each load setting, we used the reference all-WFQ network to establish how many flows of each type could be handled by the edge nodes and then calculated the core capacity. We then calculated the capacity required to support the same flows using different combinations of edge and core traffic handling schemes. Three different aspects of network design were considered: the impact of edge traffic handling, the impact of the network diameter and the impact of network connectivity.

4.2.1. Impact of edge traffic handling

Here the specific case of 20 core nodes in a full-mesh topology with a video load of 0.1 is used. For this case, each core node supports three edge nodes and each core node sends 1/19 of the total incoming traffic to every other core node. For the reference WFQ network, we thus expect the core link bandwidth to be at least (3×OC3)/19 which is

[2] We analyzed some cases with random distribution of traffic in the core but this did not have a significant impact on the results obtained.

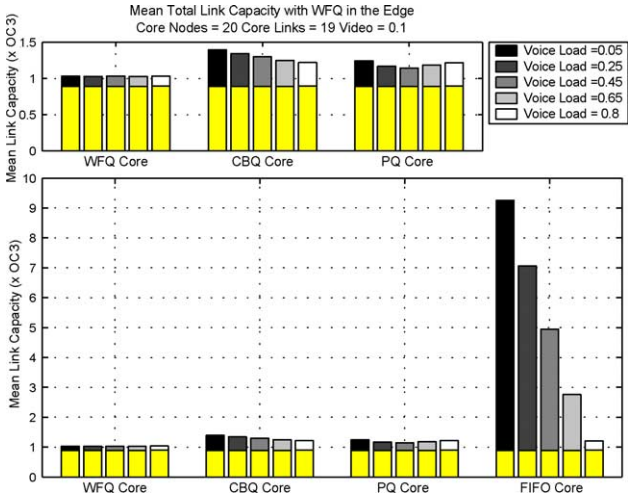

Fig. 2. Edge and core capacity with WFQ in the edge.

15.8% of an OC-3 link. Fig. 2 shows the capacity required when WFQ is used in the edge and voice load is varied.

With a WFQ core, the bulk of the capacity is in the edge (approximately 1 OC-3) and the mean link capacity of a WFQ core is 0.15 of an OC-3 which agrees with our expectations. The core capacity increases marginally with increasing voice load. With CBQ and PQ, slightly more capacity is needed in the core than with a WFQ core but it is still less than an OC-3. For a FIFO core, the bulk of the capacity is in the core for lower voice loads, ranging from 9 OC-3s when the voice load is 0.05 to less than 1 OC-3 when the voice load is 0.85. We note that the FIFO core capacity depends on the voice load; when the voice load is at its maximum, the capacity with FIFO is comparable to the other three schemes. The difference in capacity between FIFO and the other schemes is attributed to the fact that with FIFO, the performance of all traffic types must be equalized to that of the most stringent delay QoS and thus more capacity is required to achieve this when the voice load is low and the email and WWW loads are high. Fig. 3 shows

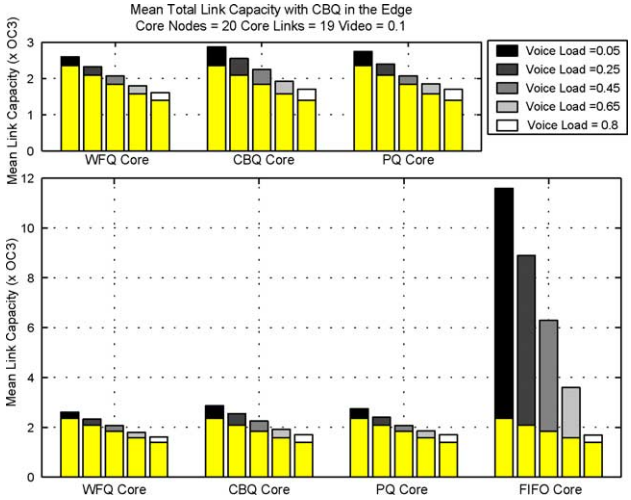

Fig. 3. Edge and core capacity with CBQ in the edge.

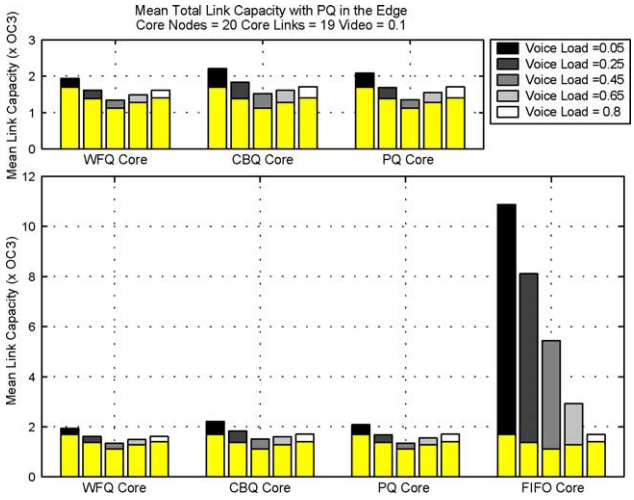

Fig. 4. Edge and core capacity with PQ in the edge.

the results with CBQ in the edge. Here the edge capacity is dependent on the voice load, being larger for smaller values of voice load, which is attributed to the correspondingly higher values of email and WWW load. The edge capacity with CBQ is about twice that with a WFQ edge. The core capacity with WFQ, CBQ and PQ is less than or equal to 1 OC-3, with CBQ having the larger core capacity. The FIFO core capacity follows the same trend as with the WFQ edge, ranging from 10 to 1 OC-3.

With PQ in the edge, the variation in capacity for varying voice load is shown in Fig. 4. The key difference between having CBQ in the edge versus PQ in the edge is the variation in edge capacity with voice load. We note that the variation is non-monotonic. When FIFO is used in the edge, the network capacity is dominated by the edge capacity as shown in Fig. 5.

We observe that the edge capacity depends on the voice load and ranges from 40 OC-3s when voice load is 0.05–2 OC-3s when the voice load is 0.8. With WFQ, CBQ and PQ cores, the core capacity is of the order of 1 OC-3

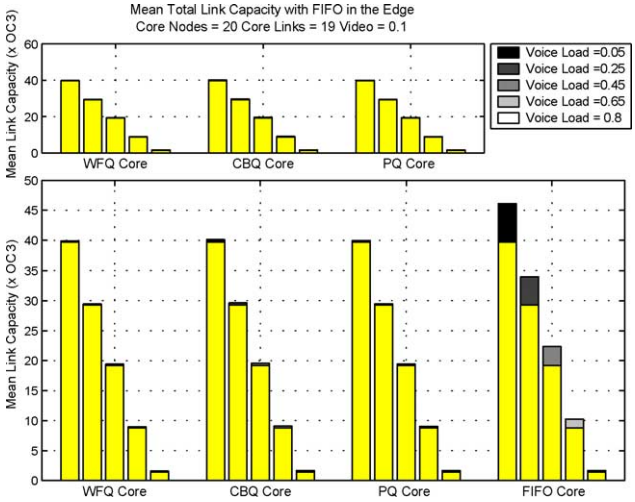

Fig. 5. Edge and core capacity with FIFO in the edge.

Table 2
Network capacity for 20 node full-mesh network (in equivalent OC-3 links)

Edge traffic handling	Core traffic handling			
	WFQ	CBQ	PQ	FIFO
WFQ	107	201	144	1497
CBQ	191	256	195	1818
PQ	146	210	149	1700
FIFO	1212	1269	1224	2318

whereas with a FIFO core the core capacity ranges from 8 to 1 OC-3.

To illustrate the implications of these results in a network-wide sense, we calculated the total network capacity for the 20 core node network in a full-mesh configuration for the specific case of voice load of 0.45 on the edge links. The results are shown in Table 2 where the capacity is in multiples of OC-3 capacity. A significant conclusion to be drawn from these results is that any combination of WFQ, CBQ and PQ in the edge and core will require capacity of the same order of magnitude.

4.2.2. Impact of network diameter

In this section, we consider how the network diameter, measured in terms of the maximum core hops traversed by a flow, affects the network capacity. We use the cases of WFQ and FIFO in the edge for illustration since the CBQ and PQ results are comparable to WFQ. In Table 3, we show the WFQ capacity and the ratios of CBQ, PQ and FIFO capacity to WFQ capacity for the case of WFQ in the edge with 20 core nodes. For all four schemes, the capacity increases with network diameter although the rate of increase is not the same. For instance, with a WFQ core, the capacity required by a 10-hop network is 5.2 times that of a 1-hop (full-mesh) network while for CBQ the factor is 11.5, for PQ it is 12.4 and for FIFO it is 13.6. When FIFO is used in the edge, the results obtained are shown in Table 4. We note that for the same hop-count, the capacities with FIFO in the edge are greater than with WFQ in the edge. Another way to look at the impact of the network diameter is to consider the utilization in the core. We define the core utilization as

$$\mu = \frac{N_{\text{edge}} \sum_{k=1}^{K} N_k \rho_k}{C_{\text{core}}}$$

Table 3
Core capacity as a function of network diameter for WFQ edge

Max hops	C^{WFQ} (xOC-3)	$C^{\text{CBQ}}/C^{\text{WFQ}}$	$C^{\text{PQ}}/C^{\text{WFQ}}$	$C^{\text{FIFO}}/C^{\text{WFQ}}$
1	54	2.8	1.72	28.5
2	85	3.52	2.17	40.08
3	102	3.96	2.24	40.21
4	113	4.0	2.47	46.06
5	156	4.66	2.8	51.8
7	198	5.27	3.36	62.6
10	281	6.17	4.11	74.6

Table 4
Core capacity as a function of network diameter for FIFO edge

Max hops	C^{WFQ} (xOC3)	$C^{\text{CBQ}}/C^{\text{WFQ}}$	$C^{\text{PQ}}/C^{\text{WFQ}}$	$C^{\text{FIFO}}/C^{\text{WFQ}}$
1	77	1.74	1.15	15.5
2	99	2.5	1.57	22.12
3	117	3.13	1.83	28.65
4	145	3.8	2.15	34.4
5	171	4.4	2.4	39.8
7	213	5.3	2.84	49.9
10	297	6.67	3.58	62.54

where N_{edge} is the total number of edge nodes, N_k is the number of flows of type k, ρ_k is the average rate of flows of type k and C_{core} is the total core capacity. We will use the results in Table 3 to discuss how network diameter affects the utilization. For WFQ, the utilization ranges from 0.73 in a full-mesh network to 0.14 in a 10-hop network. For CBQ, the range is 0.25–0.02, for PQ it is 0.4–0.03 and for FIFO it is 0.025–0.001. In general, the utilization decreases with increasing hop count. This is because with a fixed end-to-end delay budget, increasing the number of hops reduces the maximum delay per node, which results in more capacity per link to support the same external load. For CBQ, PQ and FIFO, there is the added effect of accumulation in burstyness, which increases the capacity requirements as the network increases in diameter. For the topologies considered here, small hop counts are achieved by having more links per core node and intuitively one would expect to have lower utilization when there are more links in the network. The difference comes about because the capacity per link is much higher in networks with more hops which makes the total network capacity exceed that of

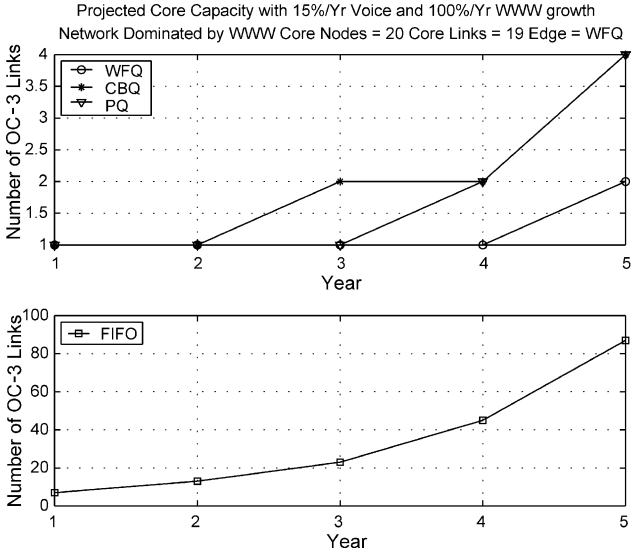

Fig. 6. Projected core capacity for 20 node network with WFQ in the edge.

Table 5
Capacity as a function of voice delay with WFQ edge

Voice delay (s)	C^{WFQ} (Mbps)	C^{CBQ}/C^{WFQ}	C^{PQ}/C^{WFQ}	C^{FIFO}/C^{WFQ}
0.01	81.73	4.04	3.0	55.16
0.015	81.73	3.28	2.2	37.01
0.02	81.73	2.93	1.89	27.9
0.025	81.73	2.75	1.7	22.57
0.03	81.73	2.65	1.6	18.9

networks with smaller hops, leading to reduced utilization. Results for different topologies can be found in [27].

4.3. Effect of projections on traffic growth

Voice traffic is predicted to grow at a rate of 10–15% per year while data traffic is predicted to grow at a rate of 100–125% per year [9]. We investigate the impact of projected annual growth of 15% in voice traffic and 100% in WWW traffic on core capacity over a period of 5 years. We use a network with 20 core nodes for illustration. Fig. 6 shows the capacity (rounded to the number of required OC-3s) needed with WFQ in the edge for the case of initial voice load of 40%, video 10%, email 15% and WWW 15%.

We note that the WFQ capacity increases by a factor of 2 to two OC-3 links after the 5 year period while CBQ and PQ both increase by a factor of 4 although the CBQ capacity increases slightly faster than PQ capacity between the second and third years. The FIFO capacity increases the most by a factor of almost 9.

4.4. Effect of delay ratios

In this section, we look at the effect of voice delay bounds on required core capacity. We use a full-mesh network with 10 core nodes for illustration with a load of 40% voice, 10% video, 15% email and 15% WWW. Results are presented in terms of the ratio of CBQ, PQ and FIFO core capacity to WFQ core capacity. Table 5 shows the effect of the voice delay bound on core capacity when WFQ is used in the edge. The WFQ capacity is not impacted by the voice delay bound while for CBQ, PQ and FIFO, the capacity decreases with increasing voice delay. For CBQ and PQ, the decrease in capacity is because a larger value of voice delay requires less capacity to support the RT queue. For FIFO, a larger value of voice delay also reduces the capacity requirements of the entire queue. Note that for FIFO, the relationship is linear in that when the delay is increased by a factor of 3, the capacity reduces by a factor 3. When FIFO is used in the edge, Table 6 shows that the WFQ core capacity increases with increasing voice delay. This is because as the voice delay increases, there is a corresponding increase in the burstyness of traffic through the FIFO edge and this requires more guaranteed bandwidth in the WFQ core. For CBQ, PQ and FIFO, the capacity decreases with increasing voice delay as before although the capacity requirements are now less than with a WFQ edge.

Table 6
Capacity as a function of voice delay with FIFO edge

Voice delay (s)	C^{WFQ} (Mbps)	C^{CBQ}/C^{WFQ}	C^{PQ}/C^{WFQ}	C^{FIFO}/C^{WFQ}
0.01	110	2.47	1.89	31.02
0.015	113	1.99	1.42	20.4
0.02	116	1.76	1.2	15
0.025	119	1.63	1.08	11.96
0.03	122	1.55	1.01	9.89

5. Conclusion

This analysis determined that any combination of WFQ, CBQ and PQ in the edge and core requires capacity of the same order of magnitude and on the basis of capacity there is no significant difference between the three traffic handling schemes. As expected, the BE network requires significantly more capacity, this is quantified in this study; a FIFO network needs an order of magnitude more capacity to achieve the same delay performance as class and flow-based schemes. Also with FIFO in the edge, the edge capacity dominates the total network capacity and again there is no significant difference between WFQ, CBQ and PQ in the core. For the network designer, these results indicate that for the same delay budget and the same number of core nodes, increasing the network connectivity increases the edge capacity but decreases the core capacity and a larger network (more core nodes) requires more capacity due to the increased network diameter.

References

[1] ATM Forum, Draft TM 4.1 Traffic Management Specification, Dec. 1998.
[2] Y. Bernet, Integrating RSVP and Diffserv, Internet Bandwidth Summit, iBAND2, May 1999.
[3] S. Blake, et al., A framework for differentiated services, IETF Internet Draft, draft-ietf-diffserv-framework-02, Feb. 1999.
[4] S. Blake, et al., An architecture for differentiated services, IETF Internet Draft, draft-ietf-diffserv-arch-02, Oct. 1998.
[5] R. Braden, D. Clark, S. Shenker, Integrated services in the internet architecture: an overview, IETF RFC 1633, June 1994.
[6] A. Bragg, S. Wright, Delay bounds for a mix of latency-rate (LR) and non-LR schedulers, ATM Forum Contribution, 99-0595, Dec. 1999.
[7] L. Breslau, S. Shenker, Best effort versus reservations: a simple comparative analyis, Proceedings of the ACM SIGOCMM, 1998 pp. 3–16.
[8] A. Charny, J. Leboudec, Delay bounds in a network with aggregate scheduling, First International Workshop on Quality of future Internet Services, QofIS 2000, Berlin, Germany, Sept. 2000.

[9] Cisco Government Affairs Website, Facts and Statistics, http://www.cisco.com/warp/public/779/govtaffs/factsNStats/internetusage.html.
[10] D. Clark, S. Shenker, L. Zhang, Supporting real-time applications in an integrated services packet network: architecture and mechanisms, Proceedings of the ACM SIGCOMM, 1992 pp. 14–26.
[11] R.L. Cruz, A calculus for network delay, Part I: Network elements in isolation, IEEE Transactions on Information Theory 37 (1) (1991).
[12] R.L. Cruz, A calculus for network delay, Part II: Network analysis, IEEE Transactions on Information Theory 37 (1) (1991).
[13] G. de Veciana, G. Kesidis, Bandwidth allocation for multiple qualities of service using generalized processor sharing, IEEE Transactions on Information Theory 42 (1) (1996).
[14] K. Dolzer, W. Payer, M. Eberspacher, A simulation study of traffic aggregation in multi-service networks, Proceedings of the IEEE Conference on High Performance Switching and Routing, Heidelberg, June 2000.
[15] C. Dovrolis, A case for relative differentiated services and the proportional differentiation model, IEEE NW 1999.
[16] G. Eichler, et al., Implementing integrated and differentiated services for the internet with ATM networks: a practical approach, IEEE COM 2000, 132.
[17] A.D. Elwalid, Mitra, R. Wentworth, A new approach for allocating buffers and bandwidth to heterogeneous regulated traffic in an ATM node, IEEE/ACM Transactions on Networking 13 (6) (1995) 1127–1165.
[18] P. Ferguson, G. Huston, Quality of Service, Delivering QoS on the Internet and in Corporate Networks, Wiley, New York, 1998.
[19] IETF RFC 2211, Specification of the Controlled-Load Network Element Service, ftp://ftp.isi.edu/in-notes/rfc2211.txt.
[20] IETF RFC 2212, Specification of Guaranteed Quality of Service, ftp://ftp.isi.edu/in-notes/rfc2212.txt.
[21] K. Iida, Performance evaluation of the architecture for end-to-end QoS provisioning, IEEE COM 38 (4) (2000).
[22] K. Kilkki, Differentiated Services for the Internet, Macmillan, New York, 1999.
[23] J. Le Boudec, Network Calculus Made Easy, Tech. Report EPFL-DI 96/218, http://lrcwww.epfl.ch/PSfiles/d4paper.ps, Dec. 1996.
[24] J. Le Boudec, Application of network calculus to guaranteed service networks, IEEE Transactions on Information Theory 44 (3) (1998).
[25] J. Liebeherr, S. Patek, E. Yilmaz, Tradeoffs in designing networks with end-to-end statistical QoS guarantees, Proceedings of IEEE/IFIP Eighth International Workshop on QoS, IWQoS 2000, Philadelphia, June 2000.
[26] Microsoft, Quality of Service Technical White Paper, http://www.microsoft.com/windows2000/techinfo/howitworks/communications/trafficmgmt/qosover.asp, Sept. 1999.
[27] P.R. Nyirenda-Jere Towela, Impact of Traffic Aggregation on Network Capacity and Quality of Service, PhD Dissertation, University of Kansas, Oct. 2001.
[28] A. Parekh, R. Gallager, A generalized processor sharing approach to flow control in integrated services networks: the single node case, IEEE/ACM Transactions on Networking 1 (3) (1993).
[29] A. Parekh, R. Gallager, A generalized processor sharing approach to flow control in integrated services networks: the multiple node case, IEEE/ACM Transactions on Networking 2 (2) (1994).
[30] H. Sariowan, R. Cruz, G. Polyzos, Scheduling for quality of service guarantees via service curves, Proceedings of ICC 1995.
[31] J. Schmitt, Aggregation of Guaranteed Service Flows TR-KOM-1998-06, http://www.kom.e-technik.tu-darmstadt.de.
[32] S. Shenker, Fundamental design issues for the future internet, IEEE JSAC 13 (7) (1995).
[33] Stardust Forums, Internet bandwidth management whitepaper, Proceedings of Internet Bandwidth Summit, iBAND2, May 1999.
[34] Stardust Forums, The Need for Quality of Service, http://www.stardust.com/qos/whitepapers/need.html, July 1999.
[35] D. Stilliadis, A. Varama, Latency-rate servers: a general model for analysis of traffic scheduling algorithms, IEEE/ACM Transactions on Networking 6 (5) (1998) 611–624.
[36] A.S. Tanenbaum, Computer Networks, third ed., Prentice-Hall, Englewood Cliffs, NJ, 1996. pp. 348–352.
[37] V. Trecordi, G. Verticale, QoS support for per-flow services: packet over SONET vs IP over ATM, IEEE Internet Computing 2000; 58.
[38] S. Wright, Delay accumulation procedures, ATM Forum Contribution, 99-0149, April 1999.
[39] S. Wright, Delay accumulation proposal, ATM Forum Contribution, 99-0295, July 1999.

Towela Nyirenda-Jere graduated from the University of Kansas with an MSc in Electrical Engineering in 1996 and thereafter joined Sprint Technology Planning and Integration as a member of technical staff working on traffic management. She returned to the University of Kansas in 1997 as a Graduate Research Assistant at the Information and Telecommunication Technology Center (ITTC). She obtained her PhD in Electrical Engineering in 2001 and joined the Malawi Polytechnic (University of Malawi) in March 2002 as a Senior Lecturer. She continues to lecture and is currently serving as a United Nations Volunteer attached to the Cisco Networking Academy Program in Malawi. Her research interests include traffic management for multi-service networks and the use of information technology in developing countries.

Victor S. Frost is currently the Dan F. Servey Distinguished Professor of Electrical Engineering and Computer Science and Director of the University of Kansas Telecommunication and Information Technology Center (ITTC). He is a Fellow of the IEEE and received a Presidential Young Investigator Award from the National Science Foundation in 1984. His current research interest is in the areas of traffic management, integrated broadband communication networks, communications system analysis, and network simulation. He has been involved in research on several national scale high-speed wide area testbeds. Government agencies, including NSF, DARPA, Rome Labs, and NASA, have sponsored his research. Professor Frost has been involved in research for numerous corporations, including Harris, Sprint, NCR, BNR, Telesat Canada, AT&T, McDonnel Douglas, DEC, and COMDISCO Systems. From 1987 to 1996 Dr. Frost was the Director of the Telecommunications and Information Sciences Laboratory. He has published over 100 journal articles and conference papers and made several presentations to committees of the Kansas Legislature on telecommunications and the future. Dr. Frost received an Air Force Summer Faculty Fellowship, a Ralph R. Teetor Educational Award from the Society of Automotive Engineers, and the Miller Professional Development Awards for Engineering Research and Service in 1986 and 1991, respectively. He is a member of Eta Kappa Nu and Tau Beta Pi. Dr. Frost received the BS, MS, and PhD degrees from the University of Kansas, Lawrence in 1977, 1978, and 1982, respectively. He joined the faculty of the University of Kansas in 1982.

Nail Akar received the BS degree from Middle East Technical University, Turkey, in 1987 and MS and PhD degrees from Bilkent University, Turkey, in 1989 and 1994, respectively, all in electrical and electronics engineering. From 1994 to 1996, he was a visiting scholar and a visiting assistant professor in the Computer Science Telecommunications program at the University of Missouri, Kansas City. He joined the Technology Planning and Integration group at Long Distance Division, Sprint, where he held a senior member of technical staff position from 1999 to 2000. Since 2000, he has been with Bilkent University as an assistant professor. His current research interests include performance analysis of computer and communication networks, queueing systems, traffic engineering, traffic control and resource allocation, and multimedia networking.