A Core Stateless Bandwidth Broker Architecture for Scalable Support of Guaranteed Services

Zhenhai Duan, Zhi-Li Zhang, *Member*, *IEEE*,
Yiwei Thomas Hou, *Member*, *IEEE*, and Lixin Gao, *Member*, *IEEE*

Abstract—We present a novel bandwidth broker architecture for scalable support of guaranteed services that decouples the QoS control plane from the packet forwarding plane. More specifically, under this architecture, *core routers do not maintain any QoS reservation states, whether per-flow or aggregate*. Instead, the QoS reservation states are stored at and managed by a bandwidth broker. There are several advantages of such a bandwidth broker architecture. Among others, it avoids the problem of inconsistent QoS states faced by the conventional hop-by-hop, distributed admission control approach. Furthermore, it allows us to design efficient admission control algorithms without incurring any overhead at core routers. The proposed bandwidth broker architecture is designed based on a *core stateless* virtual time reference system developed recently. This virtual time reference system provides a unifying framework to characterize, in terms of their abilities to support delay guarantees, both the *per-hop behaviors* of core routers and the *end-to-end properties* of their concatenation. In this paper, we focus on the design of efficient admission control algorithms under the proposed bandwidth broker architecture. We consider both *per-flow* end-to-end guaranteed delay services and *class-based* guaranteed delay services with flow aggregation. Using our bandwidth broker architecture, we demonstrate how admission control can be done on a *per domain* basis instead of on a "hop-by-hop" basis. Such an approach may significantly reduce the complexity of the admission control algorithms. In designing class-based admission control algorithms, we investigate the problem of *dynamic* flow aggregation in providing guaranteed delay services and devise a new apparatus to effectively circumvent this problem. We conduct detailed analyses to provide theoretical underpinning for our schemes as well as to establish their correctness. Simulations are also performed to demonstrate the efficacy of our schemes.

Index Terms—Network resource management, quality of services, bandwidth broker, admission control, flow aggregation.

1 Introduction

The ability to provide end-to-end guaranteed services (e.g., guaranteed delay) for networked applications is a desirable feature of the future Internet. To enable such services, Quality-of-Service (QoS) support from *both the network data plane* (e.g. packet scheduling) *and the control plane* (e.g., admission control and resource reservation) is needed. For example, under the Internet IETF Integrated Services (IntServ) architecture, scheduling algorithms such as Weighted Fair Queueing (WFQ), Virtual Clock (VC), and Rate-Controlled Earliest Deadline First (RC-EDF) [6], [8], [19], [20] were developed to support the *Guaranteed Service* [13]. Furthermore, a signaling protocol, RSVP, for setting up end-to-end QoS reservation along a flow's path was also proposed and standardized [4], [21]. However, due to its need for performing per-flow management at core routers, the *scalability* of the IntServ architecture has been questioned. To address the issue of scalability, several alternative architectures have been proposed in recent years, among others, the IETF DiffServ model [2] and the more recent *core stateless* approach based on the notion of *dynamic packet state* [14], [15].

In addressing the issue of scalability in QoS provisioning, the majority of the recent work has focused on eliminating *per-flow router state* management in the data plane. Attempts at reducing the complexity of QoS control plane have mostly followed the conventional *hop-by-hop* reservation set-up approach adopted by RSVP and ATM through *QoS control state aggregation*. In the conventional hop-by-hop reservation set-up approach, the QoS reservation set-up request of a flow is passed from the ingress router toward the egress router along the path of the flow, where each router along the path processes the reservation set-up request and determines whether the request can be honored or not *by administering a local admission control test using its own QoS state information.* However, due to the distributed nature of this approach and unreliability of the network, potential inconsistency (e.g., due to loss of signaling messages) may result in the QoS states maintained by each router, which may cause serious problems in network QoS management. RSVP addresses this problem by using *soft states*, which requires routers to periodically retransmit PATH and RESV messages, thus incurring additional communication and processing overheads. These overheads can be reduced through a number of state reduction techniques [9], [17], [18]. Under the *core stateless* framework proposed in [14], the scalability issue of the QoS control plane is addressed by maintaining only *aggregate reservation state* at each router. The problem of inconsistent QoS states is tackled via a novel *bandwidth estimation* algorithm, which

- *Z. Duan is with the Computer Science Department, Florida State University, Tallahassee, FL 32306. E-mail: duan@cs.fsu.edu.*
- *Z.-L. Zhang is with the Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455. E-mail: zhzhang@cs.umn.edu.*
- *Y.T. Hou is with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061. E-mail: thou@vt.edu.*
- *L. Gao is with the Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA 01002. E-mail: lgao@ecs.umass.edu.*

Manuscript received 2 Apr. 2002; revised 4 Mar. 2003; accepted 23 July 2003. For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number 116215.

relies on the dynamic reservation information periodically carried in packets and incurs additional processing overhead at core routers.

The conventional hop-by-hop reservation set-up approach ties such QoS control functions as admission control, resource reservation, and QoS state management to core routers, whether per-flow or aggregate QoS states are maintained at core routers. Besides the issues discussed above, this approach requires admission control and QoS state management modules to be installed at every single router to support guaranteed services. As a result, if a new level of service (say, a new guaranteed delay service class) is introduced into a network, it may require an upgrade or reconfiguration of the admission control modules at some or all core routers. An alternative, and perhaps more attractive, approach is the *bandwidth broker* (BB) architecture, which is first proposed in [10] for the *Premium Service* using the DiffServ model. Under this BB architecture, admission control, resource provisioning, and other policy decisions are performed by a centralized bandwidth broker in each network domain. Although several implementation efforts in building bandwidth brokers are under way (see, e.g., [16]), so far it is not clear what level of guaranteed services can be supported and whether core routers are still required to perform *local* admission control under the proposed BB architecture in [10].

In this paper, we present a novel *conceptually centralized* bandwidth broker architecture for scalable support of guaranteed services that *decouples the QoS control plane from the packet forwarding plane*. More specifically, under this BB architecture, the QoS reservation states are stored at and managed solely by the bandwidth broker(s) in a network domain. Despite this fact, our bandwidth broker architecture is still *capable of providing end-to-end guaranteed services, whether fine-grain per-flow delay guarantees or coarse-grain class-based delay guarantees*. This bandwidth broker architecture is built upon the *virtual time reference system* (VTRS) developed in [22]. VTRS is designed as a *unifying* scheduling framework based on which both the *per-hop behaviors* of core routers and the *end-to-end properties* of their concatenation can be characterized. Furthermore, it also provides a QoS abstraction for scheduling mechanisms of core routers that allows the bandwidth broker(s) in a network domain to perform QoS control functions such as admission control and reservation set-up with no or minimal assistance from core routers.

Because of this decoupling of data plane and QoS control plane, our bandwidth broker architecture is appealing in several aspects. First of all, by maintaining QoS reservation states only in a bandwidth broker (or bandwidth brokers), core routers are relieved of QoS control functions such as admission control, making them potentially more efficient. Second, and perhaps more importantly, a QoS control plane that is decoupled from the data plane allows a network service provider to introduce new (guaranteed) services without necessarily requiring software/hardware upgrades at core routers. Third, with QoS reservation states maintained by a bandwidth broker, it can perform sophisticated QoS provisioning and admission control algorithms to optimize network utilization in a *network-wide* fashion. Such network-wide optimization is difficult, if not impossible, under the conventional hop-by-hop reservation set-up approach. Furthermore, the problem of inconsistent QoS states facing the hop-by-hop reservation set-up approach is also significantly alleviated under our approach. Last but not least, under our approach, the reliability, robustness, and scalability issues of the QoS control plane (i.e., the bandwidth broker architecture) can be addressed *separately from, and without incurring additional complexity to, the data plane*, for example, by using distributed or hierarchical bandwidth brokers [23].

To illustrate some of the advantages presented above, in this paper, we will primarily focus on the design of efficient admission control under the proposed bandwidth broker architecture. We consider both *per-flow* end-to-end guaranteed delay services and *class-based* guaranteed delay services with flow aggregation. Using our bandwidth broker architecture, we demonstrate how admission control can be performed at a *per domain* level, instead of on a "hop-by-hop" basis. Such an approach *can* significantly reduce the complexity of the admission control algorithms. In designing class-based admission control algorithms, we investigate the problem of flow aggregation in providing guaranteed delay services, and devise a new apparatus to effectively circumvent this problem.

The remainder of this paper is structured as follows: In Section 2, we first briefly review the virtual time reference system and, then, present an overview of our proposed bandwidth broker architecture. In Section 3, we present per-flow path-oriented admission control algorithms. These admission control algorithms are extended in Section 4 to address class-based guaranteed delay services with flow aggregation. Simulation investigation is conducted in Section 5 and the paper is concluded in Section 6.

2 Virtual Time Reference System and Bandwidth Broker Architecture

In this section, we first give an overview of the virtual time reference system and, based on this reference system, we then outline the basic architecture of the proposed bandwidth broker, which decouples the QoS control plane from the packet forwarding plane for scalable support of guaranteed services.

2.1 Virtual Time Reference System and QoS Abstraction of the Data Plane

The virtual time reference system (VTRS) was developed in [22] as a unifying scheduling framework to provide scalable support for guaranteed services. The key construct in VTRS is the notion of *packet virtual time stamps*, which, as part of the packet state, are referenced and updated as packets traverse each core router. A key property of packet virtual time stamps is that they can be computed using solely the packet state carried by packets (plus a couple of *fixed* parameters associated with core routers). In this sense, the virtual time reference system is *core stateless* as no per-flow state is needed at core routers for computing packet virtual time stamps. Similar to the DiffServ framework, VTRS distinguishes edge routers from core routers. Conceptually, it consists of three logical components (Fig. 1): *edge traffic conditioning* at the network edge, *packet state* carried by packets, and *per-hop virtual time reference/update mechanism* at core routers. Below, we will briefly discuss these components (see [22] for more details).

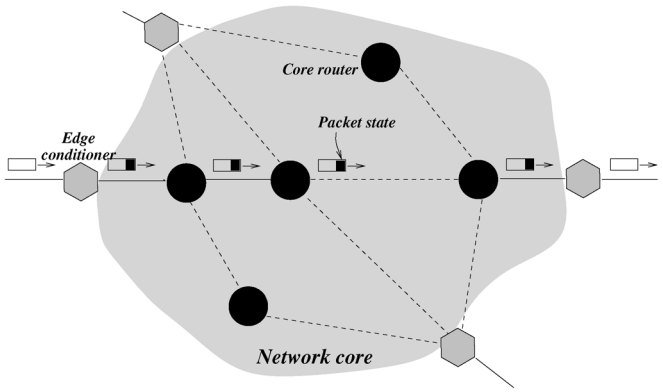

Fig. 1. Illustration of the Virtual Time Reference System.

Edge routers maintain the per-flow state. When packets are released into the network core, edge routers also insert certain packet state into the packet headers. Another important role that edge routers play is to ensure that the packets of a flow will never be injected into the network core at a rate exceeding the flow's reserved rate.

As mentioned above, when a packet is released into the network core, the edge router needs to initialize a certain packet state in the packet header. The packet state carried by the kth packet $p^{j,k}$ of a flow j contains three types of information: 1) QoS reservation (a rate-delay parameter pair $\langle r^j, d^j \rangle$) of the flow, 2) the virtual time stamp $\tilde{\omega}_i^{j,k}$ of the packet that is associated with the router i currently being traversed, and 3) the virtual time adjustment term $\delta^{j,k}$ of the packet. At the network edge, the rate-delay parameter pair $\langle r^j, d^j \rangle$, which is determined by the bandwidth broker based on flow j's QoS requirement, is inserted into every packet of the flow. For the kth packet of flow j, its virtual time stamp $\tilde{\omega}_1^{j,k}$ is initialized to $\hat{a}_1^{j,k}$, the actual time it leaves the edge conditioner. The virtual time adjustment term is primarily used for eliminating possible out-of-order transmissions of packets of a flow at routers along the path. For packet $p^{j,k}$, its virtual time adjustment term $\delta^{j,k}$ is set to $\Delta^{j,k}/q$, where q is the number of rate-based schedulers employed by the routers along the flow's path and $\Delta^{j,k}$ is computed at the network edge using the following recursive formula [22]:

$$\Delta^{j,1} = 0 \text{ and } \Delta^{j,k} = \max\left\{0, \Delta^{j,k-1} + q\frac{L^{j,k-1} - L^{j,k}}{r^j} + \hat{a}_1^{j,k-1} - \hat{a}_1^{j,k} + \frac{L^{j,k}}{r^j}\right\}, \quad \text{for } k = 2, 3, \ldots. \tag{1}$$

In the *conceptual* framework of the virtual time reference system, each core router is equipped with a per-hop virtual time reference/update mechanism to maintain the continual progression of the *virtual time* embodied by the packet virtual time stamps. This virtual time stamp, $\tilde{\omega}_i^{j,k}$, represents the arrival time of the kth packet $p^{j,k}$ of flow j at the ith core router *in the virtual time* and, thus, it is also referred to as the *virtual arrival time* of the packet at the core router. When the packet arrives at the core router, a *virtual delay*, $\tilde{d}_i^{j,k}$, is assigned to the packet. Depending on the characteristics of the router (or rather the scheduler), the virtual delay is computed differently. We distinguish two types of schedulers: rate-based and delay-based (see below for examples of them). Packets at a core router are scheduled based on their *virtual finish time*, $\tilde{\nu}^{j,k}{}_i$, which is calculated as $\tilde{\nu}^{j,k}{}_i = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k}$. To ensure bounded edge-to-edge packet delays, the virtual time stamps need to be updated properly when packets depart from a core router. Due to the page limit, we refer interested readers to [22].

An important consequence of the virtual time reference system outlined above is that the end-to-end delay bound on the delay experienced by packets of a flow across the network core can be expressed in terms of the rate-delay parameter pair of a flow and some fixed terms associated with the routers along the flow's path. Suppose there are a total of h hops along the path of flow j, of which q routers employ rate-based schedulers and $h - q$ delay-based schedulers. Then, for each packet $p^{j,k}$ of flow j, letting $\hat{f}_i^{j,k}$ denote the real departure time of the packet from the ith hop along the path, we have

$$\hat{f}_h^{j,k} - \hat{a}_1^{j,k} \leq d_{core}^j = q\frac{L^{j,max}}{r^j} + (h-q)d^j + \sum_{i=1}^{h}\Psi_i + \sum_{i=1}^{h-1}\pi_i, \tag{2}$$

where $L^{j,max}$ is the maximum packet size of flow j, Ψ_i the error term of the ith scheduler along the path, and π_i the propagation delay from the ith scheduler to the $i+1$th (see [22]).

Suppose the traffic profile of flow j is specified using the standard dual-token bucket regulator $(\sigma^j, \rho^j, P^j, L^{j,max})$, where $\sigma^j \geq L^{j,max}$ is the maximum burst size of flow j, ρ^j is the sustained rate of flow j, and P^j is the peak rate of flow j. Then, the maximum delay packets of flow j experienced at the edge shaper is bounded by

$$d_{edge}^j = T_{on}^j\frac{P^j - r^j}{r^j} + \frac{L^{j,max}}{r^j}, \tag{3}$$

where $T_{on}^j = (\sigma^j - L^{j,max})/(P^j - \rho^j)$ is the maximum duration that flow j can inject traffic at its peak rate into the network (here, the edge traffic conditioner). Hence, the end-to-end delay bound for flow j is given by

$$d_{end-to-end}^j = d_{edge}^j + d_{core}^j = T_{on}^j\frac{P^j - r^j}{r^j} + (q+1)\frac{L^{j,max}}{r^j} + (h-q)d^j + \sum_{i=1}^{h}\Psi_i + \sum_{i=1}^{h-1}\pi_i. \tag{4}$$

Observe that the end-to-end delay formula is quite similar to that specified in the IETF Guaranteed Service using the WFQ as the reference system. In this sense, the virtual time reference system provides a conceptual *core stateless* framework based on which guaranteed services can be implemented in a scalable manner using the DiffServ paradigm.

For the sake of later discussions in this paper, below we present two representative examples of core stateless scheduling algorithms designed in the VTRS framework: the rate-based *core stateless virtual clock* ($C_{\$}$VC) and the delay-based *virtual time earliest deadline first* (VT-EDF) scheduling algorithms.

$C_{\$}$VC is a work-conserving counterpart of the CJVC scheduling algorithm developed in [14]. It services packets

in the order of their virtual finish times where, as defined before, the virtual finish time of packet $p^{j,k}$ is given by $\tilde{\nu}^{j,k} = \tilde{\omega}^{j,k} + L^{j,k}/r^j + \delta^{j,k}$. It is shown in [22] that, as long as the total reserved rate of flows traversing a $C_{\$}$VC scheduler does not exceed its capacity (i.e., $\sum_j r^j \leq C$), then the $C_{\$}$VC scheduler can guarantee each flow its reserved rate r^j with the minimum error term $\Psi = L^{*,max}/C$, where $L^{*,max}$ is the largest packet size among all flows traversing the $C_{\$}$VC scheduler.

Unlike the conventional rate-controlled EDF, VT-EDF supports delay guarantees without per-flow rate control and, thus, is core stateless. It services packets in the order of their virtual finish times where, as defined before, the virtual finish time of packet $p^{j,k}$ is given by $\tilde{\nu}^{j,k} = \tilde{\omega}^{j,k} + \tilde{d}^j$. It is shown in [22] that the VT-EDF scheduler can guarantee each flow its delay parameter d^j with the minimum error term $\Psi = L^{*,max}/C$, if the following schedulability condition is satisfied:

$$\sum_{j=1}^{N} [r^j(t - d^j) + L^{j,max}]\mathbf{1}_{\{t \geq d^j\}} \leq Ct, \text{ for any } t \geq 0, \qquad (5)$$

where we assume that there are N flows traversing the VT-EDF scheduler with $0 \leq d^1 \leq d^2 \leq \cdots \leq d^N$. The indicator function $\mathbf{1}_{\{t \geq d^j\}} = 1$ if $t \geq d^j$, 0 otherwise.

2.2 A Bandwidth Broker Architecture for Support of Guaranteed Services

In this section, we present a brief overview of a novel bandwidth broker architecture for scalable support of guaranteed services in *one* network domain.[1] This bandwidth broker architecture relies on the virtual time reference system to provide a QoS abstraction of the data plane. Each router[2] in the network domain is characterized by an error term. The novelty of the proposed bandwidth broker architecture lies in that *all QoS reservation and other QoS control state information (e.g., the amount of bandwidth reserved at a core router) is removed from core routers and is solely maintained at and managed by bandwidth broker(s).* In supporting guaranteed services in the network domain, core routers perform *no QoS control and management functions* such as admission control, but only *data plane functions* such as packet scheduling and forwarding. Despite the fact that all the QoS reservation states are removed from core routers, the proposed bandwidth broker architecture is capable of supporting guaranteed services with the same granularity and expressive power (if not more) as the IntServ/Guaranteed Service model. It also allows the control and data planes to evolve independently, thus promoting faster innovation. This is in line with the current Internet IETF effort in advocating the separation of control plane and data plane [1]. In this paper, we will illustrate some of these advantages by addressing the admission control problem under the proposed bandwidth broker architecture. The major components of the bandwidth broker architecture (in particular, those pertinent to the admission control) are described below.

1. To emphasize the fact that we are considering bandwidth broker designs within a single network domain, below we will use the term "edge-to-edge delay" instead of "end-to-end delay." Moreover, the term "path" refers to a path segment (from an ingress router to an egress router) within the network domain.

2. Throughout the paper, by a router we mean the scheduler of the router used for supporting guaranteed services.

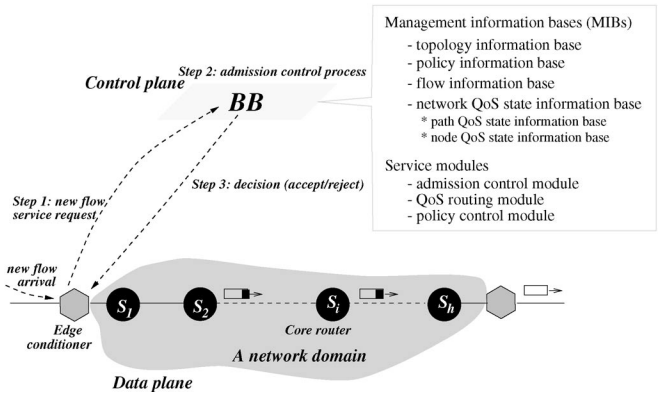

Fig. 2. Illustration of a bandwidth broker (BB) and its operation in a VTRS network domain.

As shown in Fig. 2, the bandwidth broker (BB)[3] consists of several modules such as admission control, QoS routing, and policy control. In this paper, we will focus primarily on the admission control module. The BB also maintains a number of management information bases (MIB) for the purpose of QoS control and management of the network domain. For example, the *topology information base* contains topology information that the BB uses for route selection and other management and operation purposes and the *policy information base* contains policies and other administrative regulations of the network domain. In the following, we describe the MIBs that are used by the admission control module.

Flow Information Base. This MIB contains information regarding individual flows such as flow id., traffic profile (e.g., $(\sigma^j, \rho^j, P^j, L^{j,max})$), service profile (e.g., edge-to-edge delay requirement D^j), route id. (which identifies path that a flow traverses in the network domain), and QoS reservation ($\langle r^j, d^j \rangle$ in the case of per-flow guaranteed delay services or a delay service class id. in the case of class-based guaranteed services) associated with each flow. Other administrative (e.g., policy, billing) information pertinent to a flow may also be maintained here.

Network QoS State Information Bases. These MIBs maintain the QoS states of the network domain and, thus, are the key to the QoS control and management of the network domain. Under our BB architecture, the network QoS state information is represented in two-levels using two separate MIBs: *path QoS state information base* and *node QoS state information base*.

Path QoS state information base maintains a set of paths (each with a route id.) between various ingress and egress routers of the network domain. These paths can be preconfigured or dynamically set up.[4] Associated with each path are certain static parameters characterizing the

3. For simplicity, we assume that there is a single centralized BB for a network domain. In practice, there can be multiple BBs for a network domain to improve reliability and scalability [23].

4. Note that, during the process of a path set-up, no admission control test is administered. The major function of the path set-up process is to configure forwarding tables of the routers along the path and, if necessary, provision certain scheduling/queue management parameters at the routers, depending on the scheduling and queue management mechanisms deployed. Hence, we refer to such a path as a *traffic engineered* (TE) path. Set-up of such a TE path can be done by using a path set-up signaling protocol, say, MPLS [5], [12], or a simplified version (minus resource reservation) of RSVP.

path and dynamic QoS state information regarding the path. Examples of static parameters associated a path $\mathcal{P}$ are the number of hops h on $\mathcal{P}$, the number of rate-based schedulers (q) and delay-based schedulers $(h-q)$ along $\mathcal{P}$, sum of the router error terms and propagation delay along $\mathcal{P}$, $D_{tot}^{\mathcal{P}} = \sum_{i \in \mathcal{P}}(\Psi_i + \pi_i)$, and the maximum permissible packet size (i.e., MTU) $L^{\mathcal{P},max}$. The dynamic QoS state information associated with $\mathcal{P}$ include, among others, the set of flows traversing $\mathcal{P}$ (in the case of per-flow guaranteed delay services) or the set of delay service classes and their aggregate traffic profiles (in the case of class-based guaranteed delay services) and a number of QoS state parameters regarding the (current) QoS reservation status of $\mathcal{P}$ such as the minimal remaining bandwidth $C_{res}^{\mathcal{P}}$ along $\mathcal{P}$, a sorted list of delay parameters currently supported along $\mathcal{P}$ and associated minimal residual service points and the set of "bottleneck" nodes along $\mathcal{P}$.

Node QoS state information base maintains information regarding the routers in the network domain. Associated with each router is a set of static parameters characterizing the router and a set of dynamic parameters representing the router's current QoS state. Examples of static parameters associated a router $\mathcal{S}$ are the scheduler type(s) (i.e., rate or delay-based), its error term(s) Ψ, propagation delays to its next-hop routers πs, configured total bandwidth C, and buffer size B for supporting guaranteed delay services and, if applicable, a set of delay classes and their associated delay parameters and/or a set of preprovisioned bandwidth and buffer size pairs $\langle C_k, B_k \rangle$ for each delay class supported by $\mathcal{S}$. The dynamic router QoS state parameters include the current residual bandwidth $C_{res}^{\mathcal{S}}$ at $\mathcal{S}$, a sorted list of delay parameters associated with flows traversing $\mathcal{S}$ and their associated minimal residual service points at $\mathcal{S}$, and so forth.

In the following sections, we will illustrate how some of the path and router parameters will be utilized and maintained by the BB to perform efficient admission control. Before we move to the problem of admission control using the proposed BB architecture, we briefly discuss the basic operations of the BB, in particular, those pertinent to the *admission control module*.

When a new flow with traffic profile $(\sigma^j, \rho^j, P^j, L^{j,max})$ and edge-to-edge delay requirement $D^{j,req}$ arrives at an ingress router, the ingress router sends a new flow service request message to the BB. Upon receiving the service request, the BB first checks for policy and other administrative information bases to determine whether the new flow is admissible. If not, the request is immediately rejected. Otherwise, the BB selects a path (from the ingress to an appropriate egress router in the network domain) for the new flow, based on the network topology information and the current network QoS state information, in addition to other relevant information (such as policy constraints applicable to this flow).

Once the path is selected, the BB will invoke the admission control module to determine if the new flow can be admitted. The details of admission control procedure for supporting per-flow guaranteed delay services and class-based guaranteed delay services will be presented in Section 3 and Section 4, respectively. Generally speaking, the admission control procedure consists of two phases: 1) the *admission control test* phase during which it is determined whether the new flow service request can be accommodated and how much network resources must be reserved if it can be accommodated, and 2) the *bookkeeping* phase during which the relevant information bases such as the flow information base, path QoS state information base, and node QoS state information base will be updated, if the flow is admitted. If the admission control test fails, the new flow service request will be rejected and no information bases will be updated. In either case, the BB will inform the ingress of the decision. In the case that the new flow service request is granted, the BB will also pass the QoS reservation information (e.g., $\langle r^j, d^j \rangle$) to the ingress router so that it can set up a new or reconfigure an existing edge conditioner (which is assumed to be colocated at the ingress router) for the new flow. The edge conditioner will appropriately initialize and insert the packet states into packets of the new flow once it starts to send packets into the network.

3 Admission Control for Per-Flow Guaranteed Services

In this section, we study the problem of admission control for support of per-flow guaranteed services under the proposed bandwidth broker architecture. We present a *path-oriented* approach to perform efficient admission control test and resource allocation. Unlike the conventional *hop-by-hop* approach, which performs admission control *individually* based on the *local QoS state* at each router along a path, this path-oriented approach examines the resource constraints *along the entire path within the network domain simultaneously* and makes admission control decision accordingly. As a result, we can significantly reduce the time of conducting admission control test. Furthermore, we can also perform path-wide optimization when determining resource allocation for a new flow. Clearly, such a path-oriented approach is possible because of the availability of QoS state information of the entire path at the bandwidth broker.

3.1 Path with Only Rate-Based Schedulers

To illustrate how the path-oriented approach works, we first consider a simple case where we assume that the path $\mathcal{P}$ for a new flow ν consists of only rate-based schedulers. Hence, in this case, we only need to determine whether a reserved rate r^ν can be found for the new flow for it to be admitted. The delay parameter d^ν will not be used. For simplicity of exposition, we assume that a scheduler such as *core-stateless virtual clock* (*C*$_S$VC) or *core-jitter virtual clock* (CJVC) is employed at the routers $\mathcal{S}_i$ along $\mathcal{P}$. Let $j \in \mathcal{F}_i$ denote that flow j currently traverses $\mathcal{S}_i$, and C_i be the total bandwidth at $\mathcal{S}_i$. Then, as long as $\sum_{j \in \mathcal{F}_i} r^j \leq C_i$, $\mathcal{S}_i$ can guarantee each flow j its reserved bandwidth r^j. We use $C_{res}^{\mathcal{S}_i}$ to denote the residual bandwidth at $\mathcal{S}_i$, i.e., $C_{res}^{\mathcal{S}_i} = C_i - \sum_{j \in \mathcal{F}_i} r^j$. We consider the two phases of the admission control procedure.

3.1.1 Admission Test

Let $(\sigma^\nu, \rho^\nu, P^\nu, L^{\nu,max})$ be the traffic profile of a new flow ν and $D^{\nu,req}$ be its edge-to-edge delay requirement. Let h be the number of hops in $\mathcal{P}$, the path for the new flow. From (4), in order to meet its edge-to-edge delay requirement

$D^{\nu,req}$, the reserved rate r^ν for the new flow ν must satisfy: 1) $\rho^\nu \le r^\nu \le P^\nu$ and 2)

$$D^{\nu,req} \ge d^\nu_{edge} + d^\nu_{core} = T^\nu_{on}\frac{P^\nu - r^\nu}{r^\nu} + (h+1)\frac{L^{\nu,max}}{r^\nu} + D^{\mathcal{P}}_{tot}, \tag{6}$$

where $T^\nu_{on} = (\sigma^\nu - L^{\nu,max})/(P^\nu - \rho^\nu)$ and $D^{\mathcal{P}}_{tot} = \sum_{i\in\mathcal{P}}\Psi_i + \sum_{i\in\mathcal{P}, i\neq h}\pi_i$.

Furthermore, r^ν must not exceed the minimal residual bandwidth $C^{\mathcal{P}}_{res}$ along path $\mathcal{P}$, where $C^{\mathcal{P}}_{res} = \min_{i\in\mathcal{P}} C^{\mathcal{S}_i}_{res}$ is maintained, as a path QoS parameter associated with $\mathcal{P}$, in the path QoS state MIB.

Let r^ν_{min} be the smallest r^ν that satisfies (6), i.e., $r^\nu_{min} = [T^\nu_{on}P^\nu + (h+1)L^{\nu,max}]/[D^{\nu,req} - D^{\mathcal{P}}_{tot} + T^\nu_{on}]$. Define

$$r^{low}_{fea} = \max\{\rho^\nu, r^\nu_{min}\} \text{ and } r^{up}_{fea} = \min\{P^\nu, C^{\mathcal{P}}_{res}\}.$$

Then, $\mathcal{R}^*_{fea} = [r^{low}_{fea}, r^{up}_{fea}]$ is the *feasible rate range* from which a feasible reserved rate r^ν can be selected. Clearly, if $\mathcal{R}^*_{fea}$ is empty, then the service request of the new flow ν must be rejected. Otherwise, it is admissible and $r^\nu = r^{low}_{fea}$ is the *minimal* feasible reserved rate for the new flow ν. Given that the path QoS parameters $D^{\mathcal{P}}_{tot}$ and $C^{\mathcal{P}}_{res}$ associated with $\mathcal{P}$ are maintained in the path QoS state MIB, the above admission test can be done in $O(1)$.

3.1.2 Bookkeeping

If the new flow ν is admitted into the network, several MIBs (e.g., the flow MIB, the path, and node QoS state MIBs) must be updated. The flow id., traffic profile, and service profile of the new flow will be inserted into the flow MIB. The minimal residual bandwidth $C^{\mathcal{P}}_{res}$ will be subtracted by r^ν, the reserved rate for flow ν. Similarly, for each $\mathcal{S}_i$ along $\mathcal{P}$, its residual bandwidth $C^{\mathcal{S}_i}_{res}$ will also be subtracted by r^ν. Furthermore, for any path $\mathcal{P}'$ that traverses $\mathcal{S}_i$, its minimal residual bandwidth $C^{\mathcal{P}'}_{res}$ may also be updated, depending on whether the update of $C^{\mathcal{S}_i}_{res}$ changes $C^{\mathcal{P}'}_{res}$. Last, note that, when an existing flow departs the network, the relevant MIBs should also be updated.

3.2 Path with Mixed Rate and Delay-Based Schedulers

We now consider the general case where the path $\mathcal{P}$ for a new flow ν consists of both rate-based and delay-based schedulers. In this case, we need to determine whether a rate-delay parameter pair $\langle r^\nu, d^\nu\rangle$ can be found for the new flow ν for it to be admitted. Let q be the number of rate-based schedulers and $h-q$ the number of delay-based schedulers along path $\mathcal{P}$. For simplicity of exposition, we assume that the rate-based schedulers $\mathcal{S}_i$ along path $\mathcal{P}$ employ C_SVC (or any similar) scheduling algorithm whose schedulability condition is $\sum_{j\in\mathcal{F}_i} r^j \le C_i$, whereas the delay-based schedulers $\mathcal{S}_i$ employ the *VT-EDF* scheduling algorithm, whose schedulability condition is given in (5). Hence, if $\mathcal{S}_i$ is a rate-based scheduler along $\mathcal{P}$, it can guarantee each flow j its reserved bandwidth r^j, as long as $\sum_{j\in\mathcal{F}_i} r^j \le C_i$. Similarly, if $\mathcal{S}_i$ is a delay-based scheduler along $\mathcal{P}$, it can guarantee each flow j its delay parameter d^j, as long as the schedulability condition (5) is satisfied. We now consider the two phases of the admission control procedure.

3.2.1 Admission Test

Because of the interdependence of the reserved rate r^ν and the delay parameter d^ν in the edge-to-edge delay bound (4), as well as the more complex schedulability condition (5) for the delay-based schedulers, the admission test for this case is less straightforward. By uncovering the monotonicity properties of the edge-to-edge delay formula (4) and schedulability condition (5), we show how an efficient admission test can be designed using the path-oriented approach. In addition, if the new flow ν is admissible, this admission test finds an *optimal* feasible rate-delay parameter pair $\langle r^\nu, d^\nu\rangle$ in the sense that r^ν is the *minimal* feasible rate.

Before we present the algorithm, we need to introduce some notation and transform the edge-to-edge delay formula (4) as well as the schedulability condition (5) into a form such that their monotonicity properties can be derived. As before, let $(\sigma^\nu, \rho^\nu, P^\nu, L^{\nu,max})$ be the traffic profile of the new flow ν, and $D^{\nu,req}$ its edge-to-edge delay requirement. In order for the new flow ν to be admitted along the path $\mathcal{P}$ with a rate-delay parameter pair $\langle r^\nu, d^\nu\rangle$, its edge-to-edge delay requirement $D^{\nu,req}$ must be satisfied, namely, 1) $\rho^\nu \le r^\nu \le P^\nu$ and

$$\begin{aligned} &D^{\nu,req} \ge d^\nu_{edge} + d^\nu_{core} = \\ &T^\nu_{on}\frac{P^\nu - r^\nu}{r^\nu} + (q+1)\frac{L^{\nu,max}}{r^\nu} + (h-q)d^\nu + D^{\mathcal{P}}_{tot}. \end{aligned} \tag{7}$$

Furthermore, the schedulability condition at each scheduler $\mathcal{S}_i$ must not be violated. Let $C^{\mathcal{P}}_{res}$ be the minimal residual bandwidth along $\mathcal{P}$, i.e., $C^{\mathcal{P}}_{res} = \min_{i\in\mathcal{P}} C^{\mathcal{S}_i}_{res}$. Then, from the schedulability conditions for the rate and delay-based schedulers, we see that $r^\nu \le C^{\mathcal{P}}_{res}$. Furthermore, for every delay-based scheduler $\mathcal{S}_i$ along $\mathcal{P}$, let $\langle r^k_i, d^k_i\rangle$ be the rate-delay parameter pair of flow k, where $k\in\mathcal{F}_i$. Then, for each $k\in\mathcal{F}_i$, $\mathcal{S}_i\in\mathcal{P}$ such that $d^k_i \ge d^\nu$, we must have

$$\begin{aligned} &\sum_{\{j\in\mathcal{F}_i: d^j_i \le d^k_i\}} [r^j(d^k_i - d^j_i) + L^{j,max}] + [r^\nu(d^k_i - d^\nu) + L^{\nu,max}] \\ &\le C_i d^k_i. \end{aligned} \tag{8}$$

In summary, in order for $\langle r^\nu, d^\nu\rangle$ to be a feasible rate-delay parameter pair for the new flow ν, we must have that $r^\nu \in [\rho^\nu, \min\{P^\nu, C^{\mathcal{P}}_{res}]$ and that r^ν and d^ν must satisfy (7) and (8). Theorem 1 presents the conditions when a feasible rate-delay pair $\langle r^\nu, d^\nu\rangle$ exists to satisfy the delay requirement of the new flow. In addition, it also specifies how a feasible solution with an optimal r^ν should be located. Related definitions and derivation of the theorem are relegated to the Appendix.

Theorem 1. *If $\mathcal{R}^m_{fea} \cap \mathcal{R}^m_{del}$ is empty, then no feasible rate-delay pairs $\langle r^\nu, d^\nu\rangle$ exist such that $d^\nu \in [d^{m-1}, d^m]$. Furthermore, if $\mathcal{R}^m_{fea}$ is empty or $\mathcal{R}^m_{del}$ is empty, or $r^{m,r}_{fea} < r^{m,l}_{del}$, then no intervals to the left contain a feasible solution either. More precisely, no feasible rate-delay pairs $\langle r^\nu, d^\nu\rangle$ exist such that $d^\nu \in [0, d^m)$. If $\mathcal{R}^m_{fea} \cap \mathcal{R}^m_{del}$ is not empty, then a feasible rate-delay pair $\langle r^\nu, d^\nu\rangle$ exists such that $d^\nu \in [d^{m-1}, d^m]$. Furthermore, if $r^{m,l}_{fea} < r^{m,l}_{del}$, then $r^\nu = r^{m,l}_{del}$ is the smallest rate such that there exists some $d^\nu \ge 0$ for which $\langle r^\nu, d^\nu\rangle$ is a feasible rate-delay pair. In other words, any rate-delay pair $\langle r^\nu, d^\nu\rangle$ where $r^\nu < r^{m,l}_{del}$ is not feasible.*

0. $t^\nu = \frac{1}{h-q}[D^{\nu,req} - D^{\mathcal{P}}_{tot} + T^\nu_{on}]$
1. Let m^* such that $d^{m^*-1} < t^\nu \leq d^{m^*}$
2. **for** $m = m^*, m^*-1, \ldots, 2, 1$
3. $\quad \mathcal{R}^m_{fea} \leftarrow [r^{m,l}_{fea}, r^{m,r}_{fea}]$
4. $\quad \mathcal{R}^m_{del} \leftarrow [r^{m,l}_{del}, r^{m,r}_{del}]$
5. $\quad$ **if** $(\mathcal{R}^m_{fea} \cap \mathcal{R}^m_{del} == \emptyset)$
6. $\quad\quad$ **if** $(\mathcal{R}^m_{fea} == \emptyset || \mathcal{R}^m_{del} == \emptyset || r^{m,r}_{fea} < r^{m,l}_{del})$
7. $\quad\quad\quad$ **break** with $d^\nu = d^m$
8. $\quad$ **else** /*$\mathcal{R}^m_{fea} \cap \mathcal{R}^m_{del} \neq \emptyset$*/
9. $\quad\quad$ **if** $(r^{m,l}_{fea} < r^{m,l}_{del})$
10. $\quad\quad\quad r^\nu \leftarrow r^{m,l}_{del}$
11. $\quad\quad\quad d^\nu \leftarrow t^\nu - \frac{\Xi^\nu}{r^\nu}$
12. $\quad\quad\quad$ **break** with d^ν
13. **if** $(d^\nu > t^\nu)$ no feasible value found
14. **else return** d^ν

Fig. 3. Admission test for a new flow ν on a path with mixed rate and delay-based schedulers.

Based on Theorem 1, the admission test is presented (in pseudocode) in Fig. 3. Starting with the rightmost interval $[d^{m^*-1}, d^{m^*})$, this algorithm determines iteratively whether a feasible rate-delay pair $\langle r^\nu, d^\nu \rangle$ exists such that $d^\nu \in [d^{m^*-1}, d^{m^*})$. If the new flow ν is admissible, the algorithm also finds the *feasible* rate-delay parameter pair $\langle r^\nu, d^\nu \rangle$ such that r^ν is *minimal*. The time complexity of the algorithm is $O(M)$. Note that, in general, we have $M \leq |\mathcal{F}^{del}| \leq \sum_{\mathcal{S}_i \text{ is delay-based}} |\mathcal{F}_i|$. Hence, the complexity of the algorithm hinges only on the number of distinctive delay parameters supported by the schedulers along the path of the new flow. This reduction in complexity can be significant if many flows have the same delay requirements. This is particularly the case when we consider class-based admission control with flow aggregation where a number of fixed delay classes are predefined.

3.2.2 Bookkeeping

When a new flow is admitted in the network, the BB needs to update the flow MIB, path QoS state MIB, and node QoS state MIB, among others. For a path $\mathcal{P}$ with mixed rate-based and delay-based schedulers, in addition to path parameters such as $D^{\mathcal{P}}_{tot}$ and $C^{\mathcal{P}}_{res}$, we assume that the minimum residual service S^m at each d^m is also maintained, where d^m is a distinctive delay parameter supported by one of the delay-based schedulers along $\mathcal{P}$. These parameters facilitate the admission test described in Fig. 3. Hence, when a new flow is admitted along path $\mathcal{P}$, these parameters also need to be updated. Furthermore, we assume that for each delay-based scheduler $\mathcal{S}_i$, the minimum residual service S^k_i of $\mathcal{S}_i$ at each d^k_i is also maintained at the node QoS state MIB. Furthermore, for any path traversing $\mathcal{S}_i$, the corresponding path minimum residual service parameters may also need to be updated.

4 Class-Based Guaranteed Services and Admission Control with Dynamic Flow Aggregation

Traffic aggregation is a powerful technique that can be used to significantly reduce the complexity of both the data plane and the control plane of a network domain. This reduction in complexity *may* come at a price—that guaranteed services may only be provided to individual flows at a *coarser granularity*. In this section, we address the problem of admission control for class-based guarantee services, where a fixed number of guaranteed delay service classes are offered in a network domain. The class-based guaranteed delay service model is schematically shown in Fig. 4. A new flow is placed in one of the delay service classes if it is admitted into the network. All flows in the same delay service class that traverse the same path will be aggregated into a single *macroflow*. This macroflow is shaped using an aggregate reserved rate at the edge conditioner and is guaranteed with an edge-to-edge delay bound determined by the service class. We refer to the individual user flows constituting a macroflow as the *microflows*.

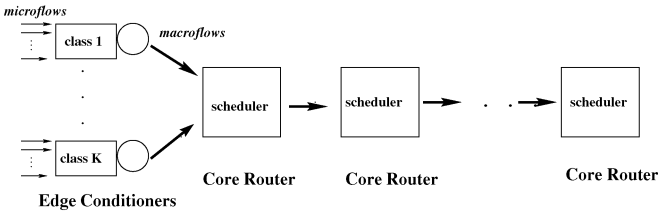

Fig. 4. Class-based guaranteed services: dynamic flow aggregation along a path.

A key issue in the design of admission control for this class-based service model is the problem of *dynamic* flow aggregation. The dynamics come from the fact that *microflows may join or leave a macroflow at any time*. Hence, the aggregate traffic profile for the macroflow may change dynamically and, as a result, the reserved rate for the macroflow may need to be adjusted accordingly. *This dynamic change in the aggregate traffic profile can cause some undesirable effect on the edge-to-edge delay experienced by the macroflow* (see Section 4.1). As far as we are aware, this problem of dynamic flow aggregation has *not* been identified nor addressed before in the literature. The existing work on traffic aggregation (in particular, in the context of guaranteed services, see, e.g., [9], [11]) has implicitly assumed *static* flow aggregation: A macroflow is an aggregation of n *fixed* microflows, with no new microflows joining or existing constituent microflows leaving in the duration of the macroflow. The remainder of this section is organized as follows: In Section 4.1, we first illustrate the impact of dynamic flow aggregation on providing edge-to-edge guaranteed services and, then, in Section 4.2, we propose solutions to circumvent the problems using our BB architecture. In Section 4.3, we will briefly describe how to perform admission control for class-based guaranteed services.

4.1 Impact of Dynamic Flow Aggregation on Edge-to-Edge Delay

Before we illustrate the impact of dynamic flow aggregation, we first introduce some notation and assumptions. Consider a macroflow α which *currently* consists of n microflows. Let $(\sigma^j, \rho^j, P^j, L^{j,max})$ be the traffic profile of the microflow j, $1 \leq j \leq n$. For simplicity, we will use a dual-token bucket

regulator, $(\sigma^\alpha, \rho^\alpha, P^\alpha, L^{\alpha,max})$, as the aggregate traffic profile for the macroflow α. Hence, we have $\sigma^\alpha = \sum_{j=1}^n \sigma^j$, $\rho^\alpha = \sum_{j=1}^n \rho^j$, $P^\alpha = \sum_{j=1}^n P^j$, and $L^{\alpha,max} = \sum_{j=1}^n L^{j,max}$. Note that $L^{\alpha,max} = \sum_{j=1}^n L^{j,max}$ is because a packet of the maximum size may arrive from each of the n microflows at the same time. Hence, the edge conditioner may see a burst of $L^{\alpha,max}$ at any time. In contrast, since only one packet from the macroflow α may leave the edge conditioner at any given time, the "maximum burst" the macroflow may carry into the network core is $\max_{j=1}^n L^{j,max}$. Let $\mathcal{P}$ denote the path of the macroflow α and $L^{\mathcal{P},max}$ denote the maximum packet size permissible in a macroflow (i.e., a delay service class) along $\mathcal{P}$. Then, $L^{\mathcal{P},max} \geq \max_{j=1}^n L^{j,max}$. Without loss of generality, we assume that $L^{\mathcal{P},max}$ is fixed.

Suppose we treat the macroflow α as *static*, i.e., with no microflows joining or leaving at any time. Let $\langle r^\alpha, d^\alpha \rangle$ be the rate-delay parameter pair reserved for the macroflow. For simplicity, assume that path $\mathcal{P}$ consists of only rate-based schedulers (h of them in total). Then, from the edge-to-edge delay formula (4), the edge-to-edge delay experienced by the macroflow α (and, therefore, by any packets from any constituent microflows) is bounded by

$$\begin{aligned} d^\alpha_{edge-to-edge} &= d^\alpha_{edge} + d^\alpha_{core} \\ &= T^\alpha_{on}\frac{P^\alpha - r^\alpha}{r^\alpha} + \frac{L^{\alpha,max}}{r^\alpha} + h\frac{L^{\mathcal{P},max}}{r^\alpha} + D^{\mathcal{P}}_{tot}, \end{aligned} \tag{9}$$

where $D^{\mathcal{P}}_{tot} = \sum_{i\in\mathcal{P}} \Psi_i + \sum_{i\in\mathcal{P}, i\neq h} \pi_i$.

We claim that, if we allow microflows to dynamically join or leave a macroflow, the edge-to-edge delay bound (9) may *no longer* hold. We illustrate this through an example. Consider a new microflow ν joins the existing macroflow α at time t^*. Let $(\sigma^\nu, \rho^\nu, P^\nu, L^{\nu,max})$ be the traffic profile of the new microflow. Denote the "new" macroflow after the microflow ν has been aggregated (i.e., the macroflow that enters the network core after t^*) by α' and let $(\sigma^{\alpha'}, \rho^{\alpha'}, P^{\alpha'}, L^{\alpha',max})$ be its traffic profile. Suppose that the reserved rate for the "new" macroflow increases from r^α to $r^{\alpha'}$ at time t^*.

We first show that the packets from the "new" macroflow may experience a worst-case delay at the edge conditioner that is larger than $d^{\alpha'}_{edge} = T^{\alpha'}_{on}\frac{P^{\alpha'} - r^{\alpha'}}{r^{\alpha'}} + \frac{L^{\alpha',max}}{r^{\alpha'}}$. This can happen, for example, in the scenario shown in Fig. 5. In this scenario, $T^\alpha_{on} \geq T^\nu_{on}$ and, thus, $T^\nu_{on} \leq T^{\alpha'}_{on} \leq T^\alpha_{on}$. We assume that all the constituent microflows of the existing macroflow α start at the same time (i.e., time 0) and are *greedy*: They dump the maximum allowed burst into the network at any time t, i.e., $A^\alpha(0,t) = \mathcal{E}^\alpha(t) = \min\{P^\alpha t + L^{\alpha,max}, \rho^\alpha t + \sigma^\alpha\}$. The new microflow ν joins the existing macroflow α at time $t^* = T^\alpha_{on} - T^\nu_{on}$ and it is also *greedy*: At any time $t \geq t^*$,

$$\begin{aligned} A^\nu(t^*, t) &= \mathcal{E}^\nu(t - t^*) \\ &= \min\{P^\nu(t - t^*) + L^{\nu,max}, \rho^\nu(t - t^*) + \sigma^\nu. \end{aligned}$$

Then, it is not difficult to see that, at time $t = T^\alpha_{on}$, the total amount of traffic that is queued at the edge conditioner is given by

$$Q(t) = (P^\alpha - r^\alpha)T^\alpha_{on} + (P^\nu + r^\alpha - r^{\alpha'})T^\nu_{on} + L^{\alpha',max}.$$

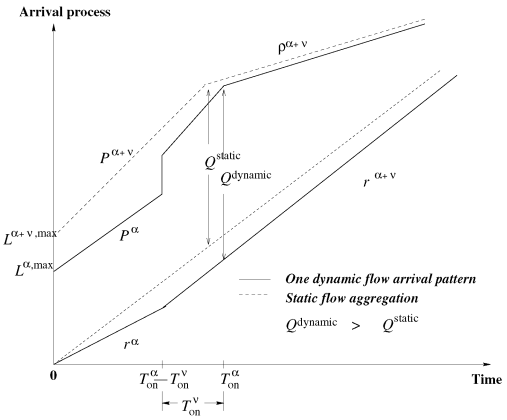

Fig. 5. An example illustrating the edge delay bound violation when a new microflow joins.

Hence, the delay experienced by a packet arriving at the edge conditioner at time $t = T^\alpha_{on}$ will be at least $Q(t)/r^{\alpha'}$, which can be shown to be larger than $d^{\alpha'}_{edge}$, in general. This larger delay is caused by the fact that, at the time, a new microflow is aggregated into an existing macroflow flow, the buffer at the edge conditioner *may not be empty*. The "old" packets queued there can cause the "new" packets to experience additional delay that is no longer bounded by $d^{\alpha'}_{edge}$.

We now consider the delay experienced by packets from the "new" macroflow α' inside the network core. Despite the fact that packets from the "new" macroflow α' are serviced with a higher reserved rate $r^{\alpha'}$ $(\geq r^\alpha)$, it can be formally established that some of these packets may experience a worst-case delay in the network core that is bounded by $d^\alpha_{core} = hL^{\mathcal{P},max}/r^\alpha + D^{\mathcal{P}}_{tot}$, not by $d^{\alpha'}_{core} = hL^{\mathcal{P},max}/r^{\alpha'} + D^{\mathcal{P}}_{tot}$. Intuitively, this can happen because the packets from the "new" macroflow may catch up with the last packets from the "old" macroflow. Hence, they may be queued behind the "old" packets, incurring a worst-case delay bounded by d^α_{core} instead of $d^{\alpha'}_{core}$. Considering both the delay at the edge conditioner and that in the network core, we see that packets of the "new" macroflow may experience an edge-to-edge delay that is no longer bounded by the edge-to-edge delay formula (9).

A similar situation may also occur when a constituent microflow leaves an existing macroflow, *if we immediately decrease the reserved rate r^α to a new lower $r^{\alpha'}$*. For example, consider the scenario illustrated in Fig. 6. Assume that a microflow ν dumps one packet with the maximum packet size of the microflow at time 0 and this packet is serviced first by the edge conditioner. Furthermore, we assume that the microflow leaves the system at the time $L^{\nu,max}/r^\alpha$. Suppose all other microflows in the macroflow are all greedy from time 0. Then, it is not hard to see if the reserved rate for α' is immediately reduced to $r^{\alpha'}$ that, at time $t = T^{\alpha'}_{on}$, the total amount of traffic that is queued at the edge conditioner is given by,

$$Q(t) = L^{\alpha',max} + T^{\alpha'}_{on}P^{\alpha'} - r^{\alpha'}T^{\alpha'}_{on} + L^{\nu,max}\frac{r^{\alpha'}}{r^\alpha}.$$

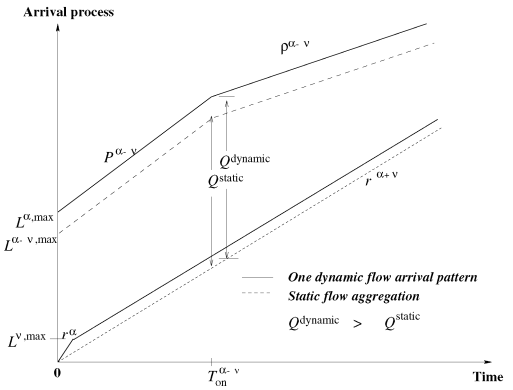

Fig. 6. An example illustrating the edge delay bound violation when a constituent microflow leaves.

Hence, the delay experienced by a packet arriving the edge conditioner at time $t = T_{on}^{\alpha'}$ will be at least $Q(t)/r^{\alpha'}$, which is larger than $d_{edge}^{\alpha'}$.

In conclusion, we see that, when a new microflow joins or a constituent microflow leaves an existing macroflow, the edge-to-edge delay experienced by the resulting macroflow after the microflow is aggregated or deaggregated may not be bounded by the edge-to-edge delay formula (9). In other words, we cannot simply treat the resulting macroflow *as if it were a completely new and independent flow.* This is because, when a new microflow joins or a constituent microflow leaves an existing macroflow, the buffer at the edge conditioner may not be empty, in general. These packets queued at the edge conditioner may have a lingering effect that invalidates the edge-to-edge delay formula (9). *New apparatuses are thus needed to tackle the problem caused by dynamic flow aggregation.* Before we move on, we would like to comment that the problem of dynamic flow aggregation is *not* unique to the virtual time reference system used in this paper. The same problem exists in a more general context. For example, dynamic flow aggregation will have the same effect on a network of WFQ schedulers, the reference system used in the IntServ model. This is because the situation happening at the edge conditioner described above will also apply to a WFQ scheduler.

4.2 Edge-to-Edge Delay Bounds under Dynamic Flow Aggregation

In this section, we present new mechanisms to effectively circumvent the problems caused by dynamic flow aggregation. The basic objective of our approach is to enable the bandwidth broker to make admission control decisions at any given time, using only the traffic profile and reserved rate of the macroflow at that time. In other words, we do not want the bandwidth broker to maintain an elaborate history record of the microflow arrival and departure events of a macroflow. To take care of the potential extra delay at the edge conditioner, in Section 4.2.1, we introduce the notion of *contingency bandwidth*, which is allocated for a short period of time (referred to as *contingency period*) after a microflow joins or leaves an existing macroflow to eliminate the lingering delay effect of the packets queued in the edge conditioner at the time the microflow joins or leaves an existing macroflow α. In Section 4.2.2, we extend the virtual time reference system to accommodate the problem of dynamic flow aggregation. With these new mechanisms implemented, we can show that the edge-to-edge delay of the macroflow after a microflow joins or leaves is bounded by a modified edge-to-edge delay formula.

4.2.1 Contingency Bandwidth and Edge Delay Bound

Contingency bandwidth works as follows: Suppose at time t^*, a microflow ν joins or leaves an existing macroflow α. Besides the reserved rate r^{α} being adjusted to a new reserved rate $r^{\alpha'}$ at t^*, *a contingency bandwidth Δr^{ν} is also temporarily allocated to the resulting "new" macroflow α' for a contingency period of τ^{ν} time units*. The contingency bandwidth Δr^{ν} and contingency period τ^{ν} is chosen in such a manner that the maximum delay in the edge conditioner experienced by any packet from the "new" macroflow α' after time t^* is bounded above by

$$d_{edge}^{new} \leq \max\{d_{edge}^{old}, d_{edge}^{\alpha'}\}, \tag{10}$$

where d_{edge}^{old} denotes the maximum edge delay bound on the "old" macroflow (i.e., before t^*) and $d_{edge}^{\alpha'} = T_{on}^{\alpha'}(P^{\alpha'} - r^{\alpha'})/r^{\alpha'} + L^{\alpha',max}/r^{\alpha'}$.

The following two theorems state the sufficient conditions on Δr^{ν} and τ^{ν} so that (10) holds, the proofs of which are fairly straightforward; we omit them here ([7]).

Theorem 2 (Microflow Join). *Suppose at time t^*, a new microflow ν with the traffic profile $(\sigma^{\nu}, \rho^{\nu}, P^{\nu}, L^{\nu,max})$ joins an existing macroflow α. Let $r^{\nu} = r^{\alpha'} - r^{\alpha}$ and $Q(t^*)$ be the size of the backlog in the edge conditioner at time t^*. Then, (10) holds if*

$$\Delta r^{\nu} \geq P^{\nu} - r^{\nu} \;\; and \;\; \tau^{\nu} \geq \frac{Q(t^*)}{\Delta r^{\nu}}. \tag{11}$$

Theorem 3 (Microflow Leave). *Suppose at time t^*, a constituent microflow ν with the traffic profile $(\sigma^{\nu}, \rho^{\nu}, P^{\nu}, L^{\nu,max})$ leaves an existing macroflow α. Let $r^{\nu} = r^{\alpha} - r^{\alpha'}$ and $Q(t^*)$ be the size of the backlog in the edge conditioner at time t^*. Then, (10) holds if*

$$\Delta r^{\nu} \geq r^{\nu} \;\; and \;\; \tau^{\nu} \geq \frac{Q(t^*)}{\Delta r^{\nu}}. \tag{12}$$

When a microflow ν joins or leaves an existing macroflow α, the BB can choose a contingency bandwidth allocation Δr^{ν} using the two theorems above. For example, when a microflow ν joins, we can set $\Delta r^{\nu} = P^{\nu} - r^{\nu} = P^{\nu} + r^{\alpha} - r^{\alpha'}$. Whereas, when a microflow ν leaves, we can set $\Delta r^{\nu} = r^{\nu} = r^{\alpha} - r^{\alpha'}$. To compute the contingency period τ^{ν} *precisely*, we need to know the backlog $Q(t^*)$ in the edge conditioner at time t^*. Since, at time t^*, the maximum delay at the edge conditioner is bounded by d_{edge}^{old}, we have

$$Q(t^*) \leq d_{edge}^{old} r(t^*) = d_{edge}^{old}(r^{\alpha} + \Delta r^{\alpha}(t^*)), \tag{13}$$

where $r(t^*)$ is total bandwidth allocated to the macroflow at time t^*, which includes the reserved rate r^{α} and the *total* contingency bandwidth $\Delta r^{\alpha}(t^*)$ allocated to the macroflow α at time t^*. Given this upper bound on $Q(t^*)$, the BB can determine an upper bound $\hat{\tau}^{\nu}$ on the contingency period τ^{ν} as follows:

$$\hat{\tau}^{\nu} = d_{edge}^{old} \frac{r^{\alpha} + \Delta r^{\alpha}(t^*)}{\Delta r^{\nu}}. \tag{14}$$

Hence, after $\hat{\tau}^\nu$, the BB can deallocate the contingency bandwidth Δr^ν at time $t^* + \hat{\tau}^\nu$. We refer to this method of determining contingency period τ^ν as the (theoretical) *contingency period bounding* approach. This scheme does not require any feedback information from the edge conditioner regarding the status of its buffer occupancy. However, in general, it can be quite *conservative*.

A more *practical* approach is to have the edge conditioner to *feedback the actual contingency period* to the BB. This scheme is referred to as the *contingency feedback* method, and works as follows: When a new microflow ν joins or leaves an existing macroflow α, the BB sends an *edge conditioner reconfiguration* message to the edge conditioner. In this message, in addition to the new reserved rate $r^{\alpha'}$, the contingency bandwidth Δr^ν is also included. Suppose the edge conditioner receives this message at time t^*. It checks the current queue length $Q(t^*)$ and computes the new contingency period τ^ν. It can immediately inform the BB of the actual value of τ^ν. Or, it can wait until $t^* + \tau^\nu$, and then send a *contingency bandwidth reset* message back to the BB. Note that, whenever the buffer in the edge conditioner becomes empty, a *contingency bandwidth reset* message can also be sent back to the BB, resetting *all* of the contingency bandwidth allocated to the macroflow α (i.e., setting $\Delta r^\alpha = 0$). This is because, after this point, the maximum delay experienced by any packets of the macroflow α is bounded by d^α_{edge}, which is solely determined by the *current* aggregate traffic profile of the macroflow α.

4.2.2 Extension to VTRS and Core Delay Bound

VTRS is developed based on the assumption that the reserved rate of a flow is fixed. In this section, we illustrate how VTRS can be extended to accommodate flow aggregation with dynamic rate changes. Based on this extension, we also derive a modified core-delay bound for flow aggregation.

Consider an existing macroflow α which traverses the path $\mathcal{P}$. For simplicity of exposition, we first assume that all the schedulers $\mathcal{S}_i$s along the path are rate-based. Suppose that, at time τ^*, the reserved rate of the macroflow α is adjusted at the edge shaper from r to r' (this happens every time the rate of the edge conditioner is adjusted). Let p^{k^*} be the last packet that leaves the edge conditioner before the rate change at τ^* and p^{k^*+1} be the first packet that leaves the edge conditioner after the rate change at τ^*. Then, for $k < k^*$, $\hat{a}_1^{k+1} - \hat{a}_1^k \geq L^{k+1}/r$ and, for $k \geq k^*$, $\hat{a}_1^{k+1} - \hat{a}_1^k \geq L^{k+1}/r'$, where recall that $\hat{a}_1^k$ denotes the time packet p^k departs the edge conditioner and arrives at the first-hop scheduler.

To accommodate reserved rate change in the virtual time reference system, we need to modify the definition of the virtual time adjustment for the transitional packet p^{k^*+1} as follows: Define

$$\Delta^{k^*+1} = \max\left\{0, \Delta^{k^*} + h\left(\frac{L^{k^*}}{r} - \frac{L^{k^*+1}}{r'}\right) + \hat{a}_1^{k^*} - \hat{a}_1^{k^*+1} + \frac{L^{k^*+1}}{r'}\right\}. \quad (15)$$

For the packets after p^{k^*+1}, the definition of Δ^k is not changed, namely, it is given in (1) with r^j replaced by r'. Indeed, we can show that, for $k = k^*+1, k^*+2, \ldots$, Δ^k is the cumulative delay experienced by packet p^k along the path $\mathcal{P}$ in the *ideal dedicated per-flow system* [22], where the rate of the servers is changed from r^α to $r^{\alpha'}$ at time τ^*. With the above modification, we can show that the following theorem holds ([7]).

Theorem 4. *For $k = k^*+1, k^*+2, \ldots$, let $\delta^k = \Delta^k/h$ where, for $k = k^*+1$, Δ^k is given by (15) and, for $k = k^*+2, \ldots$, Δ^k is given in (1) with r replaced by r'. Then, the virtual spacing and reality check properties hold for the macroflow after the rate change at τ^*. Namely, for $k = k^*+1, \ldots$, $\tilde{\omega}_i^k - \tilde{\omega}_i^{k-1} \geq L^k/r'$, and $\hat{a}_i^k \leq \tilde{\omega}_i^k$, $i \in \mathcal{P}$. Furthermore, the delay experienced by these packets in the network core is bounded by the following modified core delay formula:*

$$\hat{f}_h^k - \hat{a}_1^k \leq h \max\left\{\frac{L^{\mathcal{P},max}}{r}, \frac{L^{\mathcal{P},max}}{r'}\right\} + D^{\mathcal{P}}_{tot}. \quad (16)$$

The above modified core delay bound is derived under the assumption that all the schedulers along path $\mathcal{P}$ are rate-based. We now consider the case where some schedulers along path $\mathcal{P}$ are *delay-based*. In order to ensure the validity of the virtual time reference system under this case, we need to impose an assumption:[5] *The delay parameter d^α associated with a macroflow α is fixed, no matter whether there are microflow arrivals or departures in the macroflow*. Under this assumption, delay-based schedulers can be easily incorporated into the extended virtual time reference system presented above. Suppose there are q rate-based schedulers and $h-q$ delay-based schedulers along path $\mathcal{P}$. Then, the delay experienced by packets p^k's, $k = k^*+1, k^*+2, \ldots$, from the macroflow α after the rate change at τ^* is bounded by the following core delay formula:

$$\hat{f}_h^k - \hat{a}_1^k \leq q \max\left\{\frac{L^{\mathcal{P},max}}{r}, \frac{L^{\mathcal{P},max}}{r'}\right\} + (h-q)d^\alpha + D^{\mathcal{P}}_{tot}. \quad (17)$$

4.3 Admission Control with Dynamic Flow Aggregation

We now illustrate how to perform admission control and resource reservation with dynamic flow aggregation, based on the results obtained in Section 4.2.1 and Section 4.2.2. Consider a macroflow α. Let $D^{\alpha,req}$ be its edge-to-edge delay requirement, which we assume is *fixed* throughout the *entire* duration of the macroflow. Whenever a microflow joins or leaves the macroflow α, we need to ensure that its edge-to-edge delay requirement is still satisfied. At a given time, let r^α be the *reserved* rate of macroflow α *excluding the contingency bandwidth allocated*. Let $\Delta r^\alpha(t)$ denote the *total* contingency bandwidth allocated to α at any time t. (Here, we denote $\Delta r^\alpha(t)$ as a function to *emphasize its time dependent nature* as, every time a contingent period τ^ν expires, the corresponding contingency bandwidth is reduced from Δr^α.) Hence, at any given time t, the *actual* bandwidth allocated to the macroflow α is $r(t) = r^\alpha + \Delta r^\alpha(t) \geq r^\alpha$. Using this fact and (17), we see that if no new microflow joins or leaves, the delay in the network core experienced by packets of macroflow α is always bounded by $d^\alpha_{core} = qL^{\mathcal{P},max}/r^\alpha + (h-q)d^\alpha + D^{\mathcal{P}}_{tot}$, despite that the actual rate for macroflow α is not a constant. Let $\mathcal{P}$ be the path of macroflow α and let $C^{\mathcal{P}}_{res}$ be the minimal residual bandwidth along path $\mathcal{P}$. Hence, at most, $C^{\mathcal{P}}_{res}$

5. This assumption is *not* too restrictive in a network where a number of *fixed* delay service classes are provided.

additional amount of bandwidth (reserved and contingent) can be allocated to any macroflow along $\mathcal{P}$. We now consider how to deal with microflow joins and leaves. We consider these two cases separately below.

4.3.1 Microflow Join

Consider a new microflow ν wanting to join the existing macroflow α at time t^*. If the new microflow can be admitted, we need to determine, for the resulting "new" macroflow α', a new reserved rate $r^{\alpha'} \geq r^{\alpha}$ as well as Δr^{ν} amount of new contingency bandwidth for a contingency period of τ^{ν}. From Theorem 2, without loss of generality, we choose $\Delta r^{\nu} = P^{\nu} - r^{\alpha'} + r^{\alpha}$. Hence, in order to be able to admit the new microflow ν into the existing macroflow α, first of all, we must have $P^{\nu} \leq C_{res}^{\mathcal{P}}$. If this condition is satisfied, then we need to find the minimal new reserved rate $r^{\alpha'}$ so that the edge-to-edge delay requirement $D^{\alpha,req}$ can be satisfied for the resulting macroflow α'. Note that, after the contingency period, the edge queuing delay for any packets of the class is determined by the new class traffic profile and the reserved rate, therefore,

$$d_{edge-to-edge}^{\alpha'} = d_{edge}^{\alpha'} + \max\{d_{core}^{\alpha}, d_{core}^{\alpha'}\} \leq D^{\alpha,req}. \quad (18)$$

Since $r^{\alpha'} \geq r^{\alpha}$, $L^{\mathcal{P},max}/r^{\alpha'} \leq L^{\mathcal{P},max}/r^{\alpha}$. Hence, $d_{core}^{\alpha} \leq d_{core}^{\alpha'}$. The constraint (18) is reduced to ensure that $d_{edge}^{\alpha'} \leq D^{\alpha,req} - d_{core}^{\alpha}$. From this constraint, $r^{\alpha'}$ can be easily computed. Hence, the new microflow can be admitted if the new reserved rate $r^{\alpha'}$ can be accommodated along path $\mathcal{P}$ (i.e., if $\rho^{\nu} \leq r^{\alpha'} - r^{\alpha} \leq P^{\nu} \leq C_{res}^{\mathcal{P}}$).

If the microflow can be admitted, $r^{\alpha} + P^{\nu}$ is allocated to the macroflow during the contingency period (i.e., from t^* to $t^* + \tau^{\nu}$) and, after $t^* + \tau^{\nu}$, only $r^{\alpha'}$ will be allocated for macroflow α'.

4.3.2 Microflow Leave

When a constituent flow ν leaves the macroflow α at time t^*, we may reduce the current reserved rate r^{α} to $r^{\alpha'}$. Clearly, we must ensure that the amount of bandwidth reduced, $r^{\nu} = r^{\alpha} - r^{\alpha'}$, is chosen such that the edge-to-edge delay requirement $D^{\alpha,req}$ must not be violated. Furthermore, *this reduction in reserved rate may not take place immediately, in general.* From Theorem 3 and choosing $\Delta r^{\nu} = r^{\nu}$, we must continue to service the macroflow α with the current reserved rate r^{α} for a period of τ^{ν}. Only after the contingency period ends at $t^* + \tau^{\nu}$ can we reduce the current reserved rate by $r^{\nu} = r^{\alpha} - r^{\alpha'}$ amount. To determine $r^{\nu} = r^{\alpha} - r^{\alpha'}$, we must ensure that (18) holds. Since $r^{\alpha'} \leq r^{\alpha}$, we have $d_{core}^{\alpha} \leq d_{core}^{\alpha'}$. Hence, we need to find a new reserved rate $r^{\alpha'}$ such that $d_{edge}^{\alpha'} + d_{core}^{\alpha'} \leq D^{\alpha,req}$. In either of these cases, the minimal $r^{\alpha'}$ (if it exists) that satisfies the edge-to-edge delay bound (18) can be found easily.

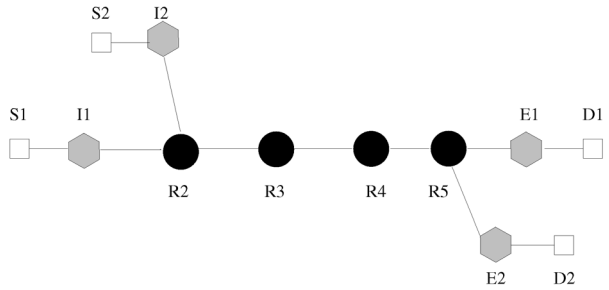

Fig. 7. The network topology used in the simulations.

5 Simulation Investigation

In this section, we conduct simulations to explore the efficacy of our admission control algorithms for both per-flow and class-based guaranteed services. In particular, we compare the performance of our per-flow admission control algorithm with that used in IntServ Guaranteed Service (GS) model. We also investigate the impact of dynamic flow aggregation on class-based guaranteed services.

Fig. 7 depicts the network topology used in the simulations, where flows generated from source 1 (S1) are destined to destination 1 (D1) via the path connecting the ingress node (I1) to the egress node (E1) and flows generated from source 2 (S2) are destined to destination 2 (D2) via the path connecting the ingress node (I2) to the egress node (E2). Each ingress node consists of two components: edge conditioners and a *core stateless* scheduler, which is the first-hop scheduler along the path. Let $x \rightarrow y$ denote the outgoing link from node x to node y. The capacity of outgoing links of all core routers is set to $1.5Mb/s$. The link capacity of $Si \rightarrow Ii$ and that of $Ei \rightarrow Di$, $i = 1, 2$, are assumed to be infinity. All the links are assumed to have zero propagation delay. We consider two simulation settings. In the first setting (*rate-based schedulers only*), all core routers employ $C_{\$}$VC schedulers. In the second setting (*mixed rate/delay-based schedulers*), schedulers employed for the outgoing links $I1 \rightarrow R2$, $I2 \rightarrow R2$, $R2 \rightarrow R3$, $R5 \rightarrow E1$ are $C_{\$}$VCs, while those for $R3 \rightarrow R4$, $R4 \rightarrow R5$, and $R5 \rightarrow E2$ are VT-EDFs. The flow traffic profiles and possible delay requirements used in the simulations are listed in Table 1.

We first conduct a set of simulations to compare the efficacy of the admission control schemes (both per-flow and class-based) in the BB/VTRS model with the standard admission control scheme [8], [13] used for the GS in the IntServ model. In the GS model, the counterpart of a $C_{\$}$VC

TABLE 1
Traffic Profiles Used in the Simulations

Type	Burst size (b)	Mean rate (b/s)	Peak rate (b/s)	Max pkt size (B)	Delay Bounds (s)	
0	60000	0.05M	0.1M	1500	2.44	2.19
1	48000	0.04M	0.1M	1500	2.74	2.46
2	36000	0.03M	0.1M	1500	3.24	2.91
3	24000	0.02M	0.1M	1500	4.24	3.81

TABLE 2
Comparison of IntServ/GS, Per-Flow BB/VTRS, and Aggregate BB/VTRS Schemes

		Number of Calls admitted			
		Rate-Based Only		Mixed Rate/Delay-Based	
Delay bounds (s)		2.44	2.19	2.44	2.19
IntServ/GS		30	27	30	27
Per-flow BB/VTRS		30	27	30	27
Aggr BB/VTRS	cd = 0.10 (s)	29	29	29	29
	cd = 0.24 (s)			29	29
	cd = 0.50 (s)			29	28

scheduler is VC while, for VT-EDF, it is RC-EDF. The RC-EDF [8], [19] scheduler employs a per-flow shaper to enforce that the traffic of each flow entering the EDF scheduler conforms to its traffic profile. In this set of simulations, traffic is sent *only* from source $S1$ to destination $D1$ (i.e., there is no cross traffic). All flows are of type 0 and have the same edge-to-edge delay requirement (either $2.44\,s$ or $2.19\,s$). Moreover, each flow has an infinite lifetime. Note that, under the per-flow guaranteed services, when the delay requirement of a type 0 flow is $2.44\,s$, a reserved rate equal to its mean sending rate will meet the delay requirement, whereas, when the delay requirement is $2.19\,s$, a higher reserved rate is needed to meet the delay requirement. In the BB/VTRS aggregate scheme, a single delay service class is used, where the edge-to-edge delay requirement of the class is set to either either $2.44\,s$ or $2.19\,s$. For each flow in the class, a fixed delay parameter (cd) is used at all of the delay-based schedulers (this parameter will only be used in the mixed rate/delay-based scheduler setting). Simulations are conducted using three different values of cd ($0.10\,s$, $0.24\,s$, and $0.50\,s$). The objective of our simulation investigation is to compare the *maximum* number of flows that can be admitted under the three different admission control schemes: IntServ/GS, Per-flow BB/VTRS, and Aggr BB/VTRS. For IntServ/GS, we use the conventional hop-by-hop admission control algorithm [3], which is also conducted at the bandwidth broker instead of indivual routers.

The simulation results are shown in Table 2. From the table, we see that the IntServ/GS and Per-flow BB/VTRS schemes accept exactly the same number of flows under all the simulation settings, whereas the Aggr BB/VTRS scheme has either slightly worse or better performance, depending on the edge-to-edge delay requirements of the flows. When the delay requirement is $2.44\,s$, the Aggr BB/VTRS scheme accepts one fewer flow than can be accepted by either the IntServ/GS or Per-flow BB/VTRS scheme. This performance loss is due to contingency bandwidth allocation in the Aggr BB/VTRS scheme: When a new flow is accepted into the delay service class, an amount of bandwidth equal to its peak rate is reserved during the contigency period to avoid potential delay bound violation. In contrast, in both the IntServ/GS and Per-flow BB/VTRS schemes, the bandwidth reserved for the new flow is equal to its mean rate. However, when the delay requirement is $2.19\,s$, the Aggr BB/VTRS scheme can accept one or two more flows than that can be accepted by either the IntServ/GS or Per-flow BB/VTRS scheme. This performance gain is due to a number of factors:

1. Each flow has precisely the same delay requirement as is provided by the delay service class.
2. The aggregate flow has a smaller core-delay bound than that of each individual flow in the per-flow guaranteed services.
3. All flows have infinite life time which, in this case, masks the *transient* effect of contingency bandwidth allocation used in the Aggr BB/VTRS scheme.

To better understand why the Aggr BB/VTRS scheme yields better performance in the case when the edge-to-edge delay requirement of the flows is $2.19\,s$, we examine more closely the bandwidth allocation allocated under the three schemes. Fig. 8 plots the average bandwidth allocated to each flow using the three schemes (under the mixed rate/delay-based scheduler setting) as a function of the number of flows accepted into the network. From the figure, we see that, under the Aggr BB scheme, the average reserved bandwidth per flow decreases as more flows are aggregated into the delay service class. (Note, in particular, that with the fixed delay parameter $cd = 0.10\,s$, a per-flow bandwidth allocation that is equal to the mean rate of the flows is sufficient to support the edge-to-edge delay bound $2.19\,s$ of the delay service class.) The average reserved bandwidth eventually drops considerably below those of the Per-flow BB/VTRS and IntServ/GS schemes. As a result, under the Aggr BB/VTRS scheme, there is sufficient residual bandwidth left to admit one or two more flows into the network. Under the Per-flow BB/VTRS scheme, a VT-EDF scheduler starts with allocating the minimum possible delay parameter to a flow, thereby producing the minimum bandwidth allocation (i.e., the mean rate of the flow). However,

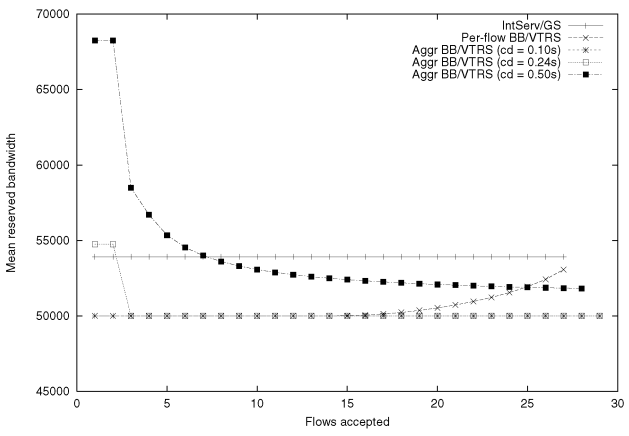

Fig. 8. Mean reserved bandwidth.

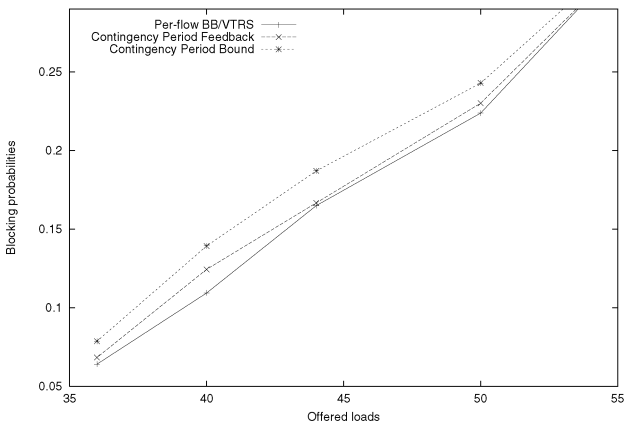

Fig. 9. Flow blocking rates.

as more flows are admitted, the feasible delay parameter that can be allocated to a new flow becomes larger, resulting in higher reserved rate. As a result, the average reserved bandwidth per flow increases. It is interesting to note that, although the Per-flow BB/VTRS and IntServ/GS admit the same number of flows (i.e., 27), the Per-flow BB/VTRS scheme has a slight smaller average reserved rate per-flow. Hence, there is more residual bandwidth left under the Per-flow BB/VTRS scheme than that under the IntServ/GS scheme, albeit this residual bandwidth is not enough to admit another flow. This slight gain in the residual bandwidth is due to the ability of the Per-flow BB/VTRS scheme to perform path-wide optimization when determining the minimum feasible rate-delay parameter pair for a flow. In contrast, in the IntServ/GS scheme, the reserved rate of a flow is determined using the WFQ reference model, which then limits the range that the delay parameter can be assigned to the flow in an RC-EDF scheduler.

In the above simulations, we have assumed that all flows have infinite lifetime. We now conduct another set of simulations in which flows have finite holding times and investigate the impact of dynamic flow aggregation on the flow blocking performance of class-based guaranteed services. In this set of simulatations, flow holding time is generated using an exponential distribution with a mean of 200 seconds. Flows may originate from either of the two sources $S1$ or $S2$. We vary the flow interarrival times to produce various offered loads. We implement two versions of the aggregate BB/VTRS scheme: one using the contingency period bounding method and another using the contingency period feedback method, as described in Section 4.2.1. Fig. 9 shows the flow blocking rates of these two schemes as well as that of the per-flow BB/VTRS scheme as we increase the flow arrival rates (and, thus, the offered load to the network). Each point in the plots of this figure is the average of five simulation runs. From the figure, we can see that, with dynamic flow arrivals and departures, the per-flow BB/VTRS scheme has the lowest flow blocking rate, as is expected. The theoretical contingency period bounding method has the worst flow blocking rate because it uses the worst-case bound on the backlog of the edge conditioners. This leads to a portion of the link bandwidth used as the contingency bandwidth, which is not immediately released. Using the contingency period feedback method, the contingency period τ^ν is, in general, very small, thus the contingency bandwidth allocated is deallocated in a very short period of time. In general, because it requires peak rate allocation at the time a new microflow arrives, the Aggr BB/VTRS schemes have a higher flow blocking rate than that of the per-flow BB/VTRS scheme. However, as the offered load increases, the flow blocking rates of these schemes converge. Hence, as the network is close to its saturation point, the (transient) effect of contigency bandwidth allocation under the Aggr BB/VTRS scheme on the flow blocking performance becomes much less prominent.

6 Conclusions

In this paper, we have presented a novel bandwidth broker architecture for scalable support of guaranteed services that decouples the QoS control plane from the packet forwarding plane. More specifically, under this architecture, *core routers do not maintain any QoS reservation states, whether per-flow or aggregate*. Instead, the QoS reservation states are stored at and managed by a bandwidth broker. There are several advantages of such a bandwidth broker architecture. Among others, it avoids the problem of inconsistent QoS states faced by the conventional hop-by-hop, distributed admission control approach. Furthermore, it allows us to design efficient admission control algorithms without incurring any overhead at core routers. The proposed bandwidth broker architecture is designed based on a *core stateless* virtual time reference system developed in [22]. In this paper, we focused on the design of efficient admission control algorithms under the proposed bandwidth broker architecture. We consider both *per-flow* end-to-end guaranteed delay services and *class-based* guaranteed delay services with flow aggregation. Using our bandwidth broker architecture, we demonstrated how admission control can be done on an entire *path* basis, instead of on a "hop-by-hop" basis. Such an approach may significantly reduce the complexity of the admission control algorithms. In designing class-based admission control algorithms, we investigated the problem of dynamic flow aggregation in providing guaranteed delay services and devised new mechanisms to effectively circumvent this problem. We conducted detailed analyses to provide theoretical underpinning for our schemes as well as to establish their correctness. Simulations were also performed to demonstrate the efficacy of our schemes.

Appendix

Proof of Theorem 1. Define $t^\nu = \frac{1}{h-q}(D^{\nu,req} - D^{\mathcal{P}}_{tot} + T^\nu_{on})$ and $\Xi^\nu = \frac{1}{h-q}[T^\nu_{on}P^\nu + (q+1)L^{\nu,max}]$. After some simple algebraic manipulations, we can rewrite (7) in the following form:

$$d^\nu \leq t^\nu - \frac{\Xi^\nu}{r^\nu} \tag{19}$$

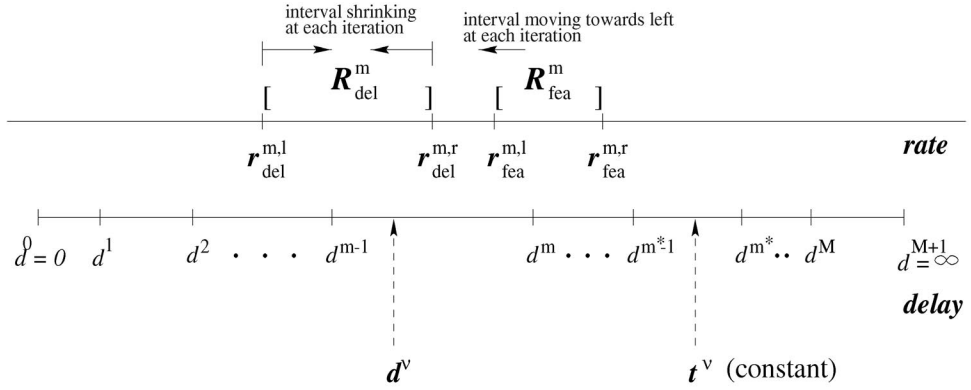

Fig. 10. The behavior of feasible range $\mathcal{R}_{fea}^m$ and delay constraint range $\mathcal{R}_{del}^m$ at the mth iteration in the search of feasible rate-delay parameter pair $\langle r^\nu, d^\nu \rangle$ for a new flow ν.

or, equivalently,

$$r^\nu \geq \frac{\Xi^\nu}{t^\nu - d^\nu}. \tag{20}$$

Note that, from (19), it is clear that $d^\nu \leq t^\nu$. Furthermore, if d^ν decreases, the upper bound on r^ν in (20) also decreases. Hence, the feasible range for r^ν shrinks from the right as d^ν decreases.

We now consider the delay constraints (8). Given any flow k traversing a delay-based scheduler $\mathcal{S}_i$ such that $d_i^k \geq d^\nu$, define

$$S_i^k = C_i d_i^k - \sum_{\{j \in \mathcal{F}_i : d_i^j \leq d_i^k\}} [r^j (d_i^k - d_i^j) + L^{j,max}]. \tag{21}$$

Then, (8) becomes

$$r^\nu (d_i^k - d^\nu) + L^{\nu,max} \leq S_i^k. \tag{22}$$

Note that S_i^k denotes the *minimum residual service* over any time interval of length d_i^k at scheduler $\mathcal{S}_i$. Hence, (22) states that the new flow ν can be accommodated at $\mathcal{S}_i$ with a rate-delay parameter pair $\langle r^\nu, d^\nu \rangle$ while *without* affecting the delay guarantee for flow k, if the service required by the new flow ν over any time interval of length d_i^k does not exceed S_i^k. For simplicity, we shall refer to S_i^k as the *minimum residual service* of $\mathcal{S}_i$ at time d_i^k.

We can consolidate the delay constraints at all the delay-schedulers along $\mathcal{P}$ as follows: Let $\mathcal{F}^{del}$ be the union of the sets of the flows at all the delay-based schedulers, i.e., $\mathcal{F}^{del} = \cup\{j \in \mathcal{F}_i : \mathcal{S}_i \text{ is delay-based}\}$. Suppose there are a total of M *distinctive* delay parameters associated with the flows in $\mathcal{F}^{del}$. Let these distinctive M delay parameters be denoted by $d^1, d^2, \ldots, d^M$, where $0 \leq d^1 < d^2 < \cdots < d^M$. For $m = 1, 2, \ldots, M$, define

$$S^m = \min\{S_i^k : k \in \mathcal{F}_i \text{ and } d_i^k = d^m, \mathcal{S}_i \text{ is delay-based}\}. \tag{23}$$

Clearly, S^m denotes the minimal residual service *among all the delay-based schedulers* at time d^m. Hence, we refer to S^m as the *minimal residual service of path $\mathcal{P}$ at time d^m*. With this notation, we see that the new flow ν can be accommodated along path $\mathcal{P}$ with a rate-delay parameter pair $\langle r^\nu, d^\nu \rangle$ while *without* affecting the delay guarantee for any flow at any delay-based scheduler along the path if, for any $d^m \geq d^\nu$, we have

$$r^\nu (d^m - d^\nu) + L^{\nu,max} \leq S^m.$$

Using (20), we can rewrite the above inequality as the following constraint on r^ν:

$$r^\nu (d^m - t^\nu) \leq S^m - \Xi^\nu - L^{\nu,max}. \tag{24}$$

We now show how to use (20) and (24) to determine whether a feasible rate-delay parameter pair $\langle r^\nu, d^\nu \rangle$ exists and, if it exists, how to find the minimum feasible r^ν. Define $d^0 = 0$ and $d^{M+1} = \infty$. Then, if the new flow ν is admissible, there must exist a rate-delay parameter pair $\langle r^\nu, d^\nu \rangle$, where $d^\nu \in [d^{m-1}, d^m)$ for some $m = 1, 2, \ldots, M+1$. From (19), it is clear that $0 \leq d^\nu \leq t^\nu$. Let m^* be such that $d^{m^*-1} < t^\nu \leq d^{m^*}$. Clearly, $[d^{m^*-1}, d^{m^*})$ is the rightmost delay interval that may contain a feasible d^ν. We now examine *iteratively* the validity of each of the delay intervals $[d^{m-1}, d^m)$, starting from $m = m^*$ down to $m = 1$.

For $m = m^*, m^* - 1, \ldots, 2, 1$, suppose $d^\nu \in [d^{m-1}, d^m)$. Then, from (20), as well as the constraint that $r^\nu \in [\rho^\nu, \min\{P^\nu, C_{res}^{\mathcal{P}}\}]$, we must have $r^\nu \in \mathcal{R}_{fea}^m = [r_{fea}^{m,l}, r_{fea}^{m,r}]$, where

$$r_{fea}^{m,l} = \max\left\{\frac{\Xi^\nu}{t^\nu - d^{m-1}}, \rho^\nu\right\}, \quad r_{fea}^{m,r} = \min\left\{\frac{\Xi^\nu}{t^\nu - d^m}, P^\nu, C_{res}^{\mathcal{P}}\right\}. \tag{25}$$

Similarly, from (24), it is not too hard to see that $d^\nu \in [d^{m-1}, d^m)$ implies that $r^\nu \in \mathcal{R}_{del}^m = [r_{del}^{m,l}, r_{del}^{m,r}]$, where

$$r_{del}^{m,l} = \max_{m \leq k < m^*} \left\{\frac{S^k - \Xi^\nu - L^{\nu,max}}{d^k - t^\nu}\right\},$$
$$r_{del}^{m,r} = \min\left\{\min_{m \leq k < m^*} \left\{\frac{\Xi^\nu + L^{\nu,max}}{t^\nu - d^k}\right\}, \min_{k \geq m^*} \frac{S^k - \Xi^\nu - L^{\nu,max}}{d^k - t^\nu}\right\}. \tag{26}$$

Observe that, when we move from the current delay interval $[d^{m-1}, d^m)$ to the next delay interval to the left $[d^{m-2}, d^{m-1})$, the corresponding feasible rate range $\mathcal{R}_{fea}^m$ determined by (25) also shifts to the left (see Fig. 10). In contrast, the corresponding feasible rate range $\mathcal{R}_{del}^m$ determined by the delay constraints (26) shrinks: The

left edge $r_{del}^{m,l}$ increases, while the right edge $r_{del}^{m,r}$ decreases (see Fig. 10). Using these monotonicity properties of $\mathcal{R}_{fea}^{m}$ and $\mathcal{R}_{del}^{m}$, we have Theorem 1. □

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers of ACM SIGCOMM 2000 and *IEEE Transactions on Parallel and Distributed Systems* for many valuable comments. Z. Duan and Z.-L. Zhang were supported in part by US National Science Foundation (NSF) Grants CAREER Awards NCR-9734428, EIA-9818338, and ITR ANI-0085824. Lixin Gao was supported in part by NSF Grants ITR ANI-0085848 and ANI-0208116. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the US National Science Foundation. An abridged version of this paper appeared in *Proceedings of the SIGCOMM 2000* with the title "Decoupling QoS Control from Core Routers: A Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services."

REFERENCES

[1] T. Anderson et al., "Requirements for Separation of IP Control and Forwarding," Internet Draft, Feb. 2002.
[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," RFC 2475, Dec. 1998.
[3] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An Overview," RFC 1633, June 1994.
[4] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource Reservation Protocol (RSVP)—Version 1 Functional Specification," RFC 2205, Sept. 1997.
[5] R. Callon, P. Doolan, N. Feldman, A. Fredette, G. Swallow, and A. Viswanathan, "A Framework for Multiprotocol Label Switching," Internet Draft, Sept. 1999.
[6] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *Proc. ACM SIGCOMM,* pp. 1-12, Sept. 1989.
[7] Z. Duan, Z.-L. Zhang, Y.T. Hou, and L. Gao, "Decoupling QoS Control from Core Routers: A Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services," technical report, Dept. of Computer Science and Eng., Univ. of Minnesota, Jan. 2000.
[8] L. Georgiadis, R. Guérin, V. Peris, and K.N. Sivarajan, "Efficient Network QoS Provisioning Based on Per Node Traffic Shaping," *IEEE/ACM Trans. Networking,* vol. 4, no. 4, pp. 482-501, 1996.
[9] R. Guérin, S. Blake, and S. Herzog, "Aggregating RSVP-Based QoS Requests," Internet Draft, 1997.
[10] K. Nichols, V. Jacobson, and L. Zhang, "A Two-Bit Differentiated Services Architecture for the Internet," RFC 2638, July 1999.
[11] S. Rampal and R. Guérin, "Flow Grouping for Reducing Reservation Requirements for Guaranteed Delay Service," Internet Draft, July 1997.
[12] E.C. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," Internet Draft, Aug. 1999.
[13] S. Shenker, C. Partridge, and R. Guérin, "Specification of Guaranteed Quality of Service," RFC 2212, Sept. 1997.
[14] I. Stoica and H. Zhang, "Providing Guaranteed Services Without Per Flow Management," *Proc. ACM SIGCOMM,* Sept. 1999.
[15] I. Stoica, H. Zhang, S. Shenker, R. Yavatkar, D. Stephens, A. Malis, Y. Bernet, Z. Wang, F. Baker, J. Wroclawski, C. Song, and R. Wilder, "Per Hop Behaviors Based on Dynamic Packet States," Internet Draft, Feb. 1999.
[16] A. Terzis, J. Ogawa, S. Tsui, L. Wang, and L. Zhang, "A Prototype Implementation of the Two-Tier Architecture for Differentiated Services," *Proc. IEEE Real-Time Technology and Applications Symp.,* 1999.
[17] L. Wang, A. Terzis, and L. Zhang, "A New Proposal of RSVP Refreshes," *Proc. IEEE Int'l Conf. Network Protocols,* Nov. 1999.
[18] L. Wang, A. Terzis, and L. Zhang, "RSVP Refresh Overhead Reduction by State Compression," Internet Draft, June 1999.
[19] H. Zhang and D. Ferrari, "Rate-Controlled Static-Priority Queueing," *Proc. IEEE INFOCOM,* pp. 227-236, Apr. 1993.
[20] L. Zhang, "Virtual Clock: A New Traffic Control Algorithm for Packet Switching Networks," *Proc. ACM SIGCOMM,* pp. 19-29, Sept. 1990.
[21] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: A New Resource Reservation Protocol," *IEEE Network,* pp. 8-18, Sept. 1993.
[22] Z.-L. Zhang, Z. Duan, and Y.T. Hou, "Virtual Time Reference System: A Unifying Scheduling Framework for Scalable Support of Guaranteed Services," *IEEE J. Selected Areas in Comm.,* special issue on Internet QoS, Dec. 2000.
[23] Z.-L. Zhang, Z. Duan, and Y.T. Hou, "On Scalable Design of Bandwidth Brokers," *IEICE Trans. Comm.,* vol. 84, no. B(8), Aug. 2001.

Zhenhai Duan received the BS degree from Shandong University, China, in 1994, the MS degree from Beijing University, China, in 1997, and the PhD degree from the University of Minnesota in 2003, all in computer science. He is currently an assistant professor in the Computer Science Department at Florida State University. His research interests include computer networks and multimedia communications, especially the scalable QoS control and management in the Internet, Internet routing protocols and service architectures, and networking security. Dr. Duan is a corecipient of the 2002 IEEE International Conference on Network Protocols Best Paper Award.

Zhi-Li Zhang received the BS degree in computer science from Nanjing University, China, in 1986, and the MS and PhD degrees in computer science from the University of Massachusetts in 1992 and 1997. In 1997, he joined the Computer Science and Engineering Faculty at the University of Minnesota, where he is currently an associate professor. From 1987 to 1990, he conducted research in the Computer Science Department at Århus University, Denmark, under a fellowship from the Chinese National Committee for Education. He has held visiting positions at Sprint Advanced Technology Labs, IBM T.J. Watson Research Center, Fujitsu Labs of America, Microsoft Research China, and INRIA, Sophia-Antipolis, France. His research interests include computer communication and networks, especially the QoS guarantee issues in high-speed networks, multimedia and real-time systems, and modeling and performance evaluation of computer and communication systems. Dr. Zhang received the US National Science Foundation CAREER Award in 1997. He was also awarded the prestigious McKnight Land-Grant Professorship at the University of Minnesota. Dr. Zhang is corecipient of an ACM SIGMETRICS best paper award and an IEEE ICNP best paper award. He currently serves on the editorial board of *IEEE/ACM Transactions on Networking* and *Computer Network, an International Journal*. He is a member of the IEEE, ACM, and INFORMS Telecommunication Section.

Yiwei Thomas Hou received the BE degree (summa cum laude) from the City College of New York in 1991, the MS degree from Columbia University in 1993, and the PhD degree from Polytechnic University, Brooklyn, New York, in 1998, all in electrical engineering. From 1997 to 2002, Dr. Hou was a research scientist and project leader at Fujitsu Laboratories of America, IP Networking Research Department, Sunnyvale, California. He is currently an assistant professor at Virginia Tech, the Bradley Department of Electrical and Computer Engineering, Blacksburg, Virginia. Dr. Hou's research interests include wireless video sensor networks, multimedia delivery over wireless networks, scalable architectures, protocols, and implementations for differentiated services Internet, and service overlay networking. He has published extensively in the above areas and is a corecipient of the 2002 IEEE International Conference on Network Protocols Best Paper Award and the 2001 *IEEE Transactions on Circuits and Systems for Video Technology* Best Paper Award. He is a member of the IEEE and ACM.

Lixin Gao received the PhD degree in computer science from the University of Massachusetts at Amherst in 1996. She is an associate professor of electrical and computer engineering at the University of Massachusetts, Amherst. Her research interests include multimedia networking and Internet routing. She was a visiting researcher at AT&T Research Labs and DIMACS between May 1999 and January 2000. She received a US National Science Foundation CAREER award in 1999 and an Alfred P. Sloan fellowship in 2003. Dr. Gao is on the editorial board of *IEEE/ACM Transactions on Networking*. She is a member of the IEEE.