Decoupling QoS Control from Core Routers: A Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services*

Zhi-Li Zhang†, Zhenhai Duan†, Lixin Gao‡, and Yiwei Thomas Hou§

† University of Minnesota
Minneapolis, MN 55455
{zhzhang,duan}@cs.umn.edu

‡ Smith College
Northampton, MA 01060
gao@cs.smith.edu

§ Fujitsu Labs of America
Sunnyvale, CA 94086
thou@fla.fujitsu.com

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, QoS reservation states are stored at and managed by bandwidth broker(s). There are several advantages of such a bandwidth broker architecture. Among others, it relieves core routers of QoS control functions such as admission control and QoS state management, and thus enables a network service provider to introduce new (guaranteed) services without necessarily requiring software/hardware upgrades at core routers. 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 [20].

In this paper we focus on the design of efficient admission control algorithms under the proposed bandwidth broker architecture. We consider both *per-flow* guaranteed delay services and *class-based* guaranteed delay services with flow aggregation. We demonstrate 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. We also study the impact of dynamic flow aggregation on the design of class-based admission control algorithms. Based on the proposed bandwidth broker architecture, we devise effective mechanisms to circumvent the problem caused by dynamic flow aggregation.

*This work was supported in part by NSF under the CAREER Award NCR-9734428. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

SIGCOMM '00, Stockholm, Sweden.

1. INTRODUCTION

The ability to provide end-to-end guaranteed services (e.g., guaranteed rate or 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) [5, 18] were developed to support the *Guaranteed Service* [11]. Furthermore, a signaling protocol, RSVP, for setting up end-to-end QoS reservation along a flow's path was also proposed and standardized [4, 19]. 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 using *dynamic packet state* [12, 13].

In addressing the issue of scalability in QoS provisioning, the majority of the recent works have focused on eliminating *per-flow router state* management in the data plane. For example, under the DiffServ architecture no per-flow QoS state is maintained at core routers, and user flows are aggregated and processed based on a number of service bits carried in the packet headers. Consequently, only coarse-grain QoS support is provided to users. In contrast, the *core stateless* approach aims at providing end-to-end per-flow guarantees without the complexity of per-flow QoS management. To accomplish this goal, a novel scheduling mechanism—the Core Jitter Virtual Clock (CJVC)—is developed in [12] to support end-to-end per-flow delay guarantees *without requiring per-flow states in core routers.*

Attempts at reducing the complexity of QoS control plane have mostly followed the conventional *hop-by-hop* reservation set-up approach adopted by RSVP through *QoS control state aggregation.* In the hop-by-hop reservation set-up approach, each router maintains its own QoS state database and administers a *local* admission control test to determine whether a flow reservation set-up request can be honored or not. To ensure consistency among the QoS state databases maintained by individual routers, RSVP uses *soft QoS states*, which requires periodic state exchange among routers, thus

incurring additional communication and processing overheads. These overheads can be reduced through a number of control state reduction techniques [6, 16, 17]. Other hop-by-hop reservation set-up protocols have also been proposed [8, 15]. In general, 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. Therefore it requires admission control and QoS state management modules to be installed at every router to support QoS provisioning.

In the context of the DiffServ architecture, an alternative, and perhaps more attractive, approach—the *bandwidth broker* approach—is proposed in [7] to support the so-called *Premium Service*. In this approach, admission control, resource provisioning and other policy decisions are performed by a centralized bandwidth broker (BB) in each network domain. Although several implementation efforts in building bandwidth brokers are under way (see, e.g., [14]), so far many issues regarding the design of bandwidth brokers, such as admission control and QoS provisioning, have not been addressed adequately in the literature. For example, under the proposed BB architecture in [7], it appears that only *coarse-grain* QoS provisioning can be supported and that *explicit configuration* of core routers is still needed to provide QoS guarantees [14]. Clearly, the deliverable QoS performance will hinge on how frequently such configuration is performed. In addition, it is not clear how admission control should be performed under such a BB architecture, in particular, whether core routers are still required to perform *local* admission control and QoS state management.

In this paper we present a novel bandwidth broker architecture for support of QoS provisioning that *decouples the QoS control plane from the data forwarding plane*. This BB architecture is designed under the *core stateless* framework, using the technique of dynamic packet state [13]. Under our proposed BB architecture, the QoS reservation states are stored at and managed *solely* by the BB(s) in a network domain, and *no or minimal configuration* of core routers is required. In other words, core routers are completely relieved of QoS control functions such as admission control and state management (whether per-flow or aggregate), making them potentially more efficient. This decoupling of data plane and QoS control plane is appealing in several aspects. Among others, it allows a network service provider to introduce new QoS services without necessarily requiring software/hardware upgrades at core routers. Furthermore, as we will demonstrate, it also enables the deployment of 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.

The objective of this paper is to demonstrate how *guaranteed* services (both per-flow and class-based) can be supported using the proposed BB architecture, despite the fact that no QoS states are maintained at core routers. We focus on guaranteed services partly because these services are well-defined and understood, and partly because a unifying core stateless framework—the virtual time reference system [20]—has been developed that provides a QoS abstraction of the data plane for supporting guaranteed delay and rate services. Using guaranteed services as examples, we illustrate how admission control can be performed using the proposed BB architecture without the assistance of core routers. In particular, we develop a *path-oriented* approach to the design of efficient admission control algorithms under the proposed BB architecture. We establish that 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, despite the fact that all QoS reservation states are removed from core routers and maintained solely at the bandwidth broker. This is achieved without the potential complexity and scalability problems of the IntServ model.

However, the bandwidth broker approach to QoS provisioning introduces a set of new issues. In particular, the scalability of the BB architecture—its ability to manage a large number of QoS control states and process a large volume of user flow QoS requests—is an important issue that must be investigated. To partially address this issue, in this paper we also consider the support of coarse-grain class-based guaranteed services using the proposed BB architecture. Via flow aggregation, the number of QoS states maintained by the BB can be reduced, and the complexity of admission control operations can be lowered, thereby enhancing the processing capacity of the BB. However, in the context of guaranteed services, (*dynamic*) flow aggregation can have an undesirable transient effect[1] that may result in delay bound violation, if proper care is not taken. We illustrate how this problem can be solved using relatively simple mechanisms under the proposed BB architecture.

Our work is only a first step towards addressing many problems that still remain in the design and implementation of the proposed BB architecture. For example, to further improve scalability, a distributed (or hierarchical) architecture consisting of multiple BBs may be necessary to support QoS provisioning in a large network domain. Such an architecture introduces many new design and implementation issues. The problem of inter-domain QoS reservation and service-level agreement [2, 7] is another important issue that must be addressed. In addition, supporting statistical or other forms of QoS guarantees using the proposed BB architecture is also a challenging problem. Clearly, all these issues must be satisfactorily resolved before the proposed BB architecture can be deployed in practice.

The remainder of this paper is structured as follows. Section 2 outlines the proposed bandwidth broker architecture, where we also briefly review the virtual time reference system developed in [20], and present a high-level overview of the admission control operations under the proposed BB architecture. In Section 3, we design efficient path-oriented admission control algorithms for per-flow guaranteed services. These admission control algorithms are extended in Section 4 to support class-based guaranteed services with dynamic flow aggregation. Simulation investigation is conducted in Section 5, and the paper is concluded in Section 6.

[1] This phenomenon is *not* unique to the core stateless framework, and may occur in any guaranteed services with dynamic flow aggregation.

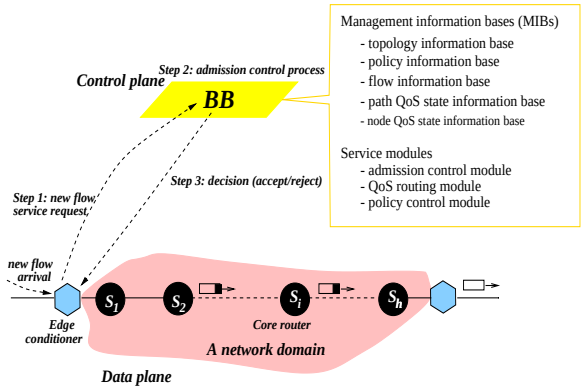

Figure 1: Illustration of a bandwidth broker (BB) and its operation in a VTRS network domain.

2. OVERVIEW OF THE PROPOSED BANDWIDTH BROKER ARCHITECTURE

In this section we outline the proposed bandwidth broker architecture for support of guaranteed services within a single network domain. This bandwidth broker architecture is built upon the *virtual time reference system* (VTRS) developed in [20], a core stateless framework that provides a QoS abstraction of the data plane. Under the proposed BB architecture, core routers perform only *data plane functions* such as packet scheduling and forwarding, using the dynamic packet state carried in packet headers. No QoS states, whether per-flow or aggregate, are maintained at any core routers. Instead, all QoS states are solely maintained at and managed by the BB(s), no or minimal configuration of core routers is needed. Furthermore, all *QoS control plane functions* such as admission control and resource reservation are performed by the BB(s), without involvement of core routers. Figure 1 illustrates the basic components and operations of the proposed BB architecture as well as its relationship to the data plane.

As shown in Figure 1, a bandwidth broker[2] (BB) consists of several service modules (i.e., servers) such as admission control, routing and policy control. The routing module peers with routers to obtain the topology information of the network domain [1] and is responsible for path selection and set-up (using, e.g., MPLS [10]). The policy control module maintains a policy information base and is responsible for network policy administration. The admission control module maintains the QoS states of the network domain, which are stored in several management information bases (see Section 2.2), and is responsible for admission control and resource reservation.

In this paper we will focus primarily on the operations of the admission control module for support of guaranteed services. In particular, we will demonstrate how efficient admission control can be performed under the proposed BB architecture. Before we embark on the problem of admission control using the proposed BB architecture, we need to understand the basic operations of the virtual time reference system and its implication on QoS provisioning. In the following, we first briefly review the virtual time reference system, and then present a high-level description of the admission control module and the QoS state information bases it maintains.

[2]In this paper we assume that there is a single centralized BB for a network domain. In practice, a distributed or hierarchical architecture consisting of multiple BBs can be employed to improve reliability and scalability. We will explore these issues in future research. Note that an important advantage of the BB approach is that the reliability 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.

2.1 The Virtual Time Reference System

The virtual time reference system is designed as a *unifying* scheduling framework based on which both the *per-hop behaviors* of core routers (in terms of their abilities to provide delay and bandwidth guarantees) and the *end-to-end properties* of their concatenation can be characterized. This unifying scheduling framework enables core routers to focus on packet scheduling and forwarding functions based on the packet state carried in packet headers without *direct* involvement in QoS control. The key construct in the virtual time reference system 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. Packet virtual time stamps are computed using only the packet state carried by packets (plus a couple of fixed parameters associated with core routers), and thus the virtual time reference system is *core stateless*.

Conceptually, the virtual time reference system consists of three logical components: *packet state* carried by packets, *edge traffic conditioning* at the network edge, and *per-hop virtual time reference/update mechanism* at core routers (see Figures 2 and 3). These three components are briefly described below. A more detailed description can be found in [20].

Edge Traffic Conditioning. Edge traffic conditioning plays a key role in the VTRS, as it ensures that the packets of a flow[3] will never be injected into the network core at a rate exceeding its reserved rate. Formally, for a flow j with a reserved rate r^j, the inter-arrival time of two consecutive packets of the flow at the first hop core router is such that $\hat{a}_1^{j,k+1} - \hat{a}_1^{j,k} \geq \frac{L^{j,k+1}}{r^j}$, where $\hat{a}_1^{j,k}$ denotes the arrival time of the kth packet $p^{j,k}$ of flow j at the network core, $L^{j,k}$ the size of packet $p^{j,k}$, and r^j the reserved rate of flow j.

Packet State. After going through the edge conditioner at the network edge, packets entering the network core carry in their packet headers certain packet state information that is initialized and inserted at the network edge. The packet state carried by packet $p^{j,k}$ of a flow j contains three types of information: 1) a rate-delay parameter pair $\langle r^j, d^j \rangle$ of the flow, determined by the bandwidth broker based on flow j's QoS requirement; 2) the virtual time stamp $\tilde{\omega}_i^{j,k}$ of the packet that is associated with the router i currently being traversed (it is initialized to $\hat{a}_1^{j,k}$, the actual time it leaves the edge conditioner and enters the first core router along the flow's path); and 3) the virtual time adjustment term

[3]Here a flow can be either an individual user flow, or an aggregate traffic flow of multiple user flows, defined in any appropriate fashion.

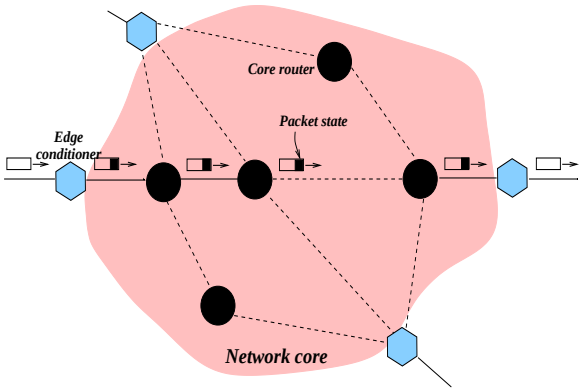

Figure 2: A conceptual network model in the virtual time reference system.

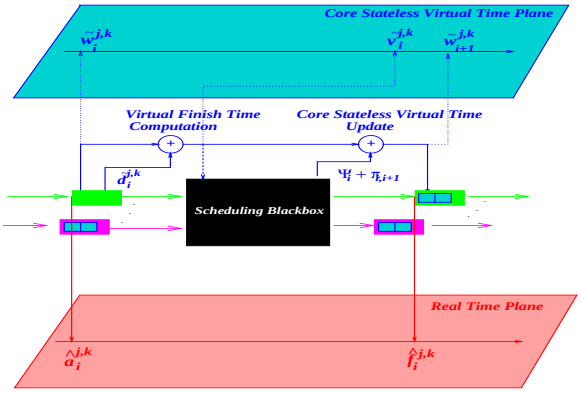

Figure 3: Virtual time reference system: per-hop behavior and operations.

$\delta^{j,k}$ of the packet, a parameter that is computed at the edge and used to ensure that the virtual spacing property (see below) of virtual time stamps is satisfied [20].

Virtual Time Reference/Update Mechanism and Per-Hop Router Behavior Characterization. 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 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. The virtual time stamps $\tilde{\omega}_i^{j,k}$'s associated with packets of flow j satisfy the following two important properties: 1) *virtual spacing property:* $\tilde{\omega}_i^{j,k+1} - \tilde{\omega}_i^{j,k} \geq \frac{L^{j,k+1}}{r^j}$, and 2) the *reality check property:* $\hat{a}_i^{j,k} \leq \tilde{\omega}_i^{j,k}$, where $\hat{a}_i^{j,k}$ denotes the actual arrival time of packet $p^{j,k}$ at router i.

In order to ensure that these two properties are satisfied, the virtual time stamps must be appropriately referenced or updated as packets enter or leave a core router. The referencing/updating rule depends on the scheduling algorithm (or *scheduler*) employed by a core router. We distinguish two types of schedulers: *rate-based* and *delay-based*, depending on how the *virtual deadline* and *virtual finish time* are computed for packets traversing it. For example, if the scheduler $\mathcal{S}_i$ at the ith router is rate-based, packet $p^{j,k}$ is associated with the virtual deadline $\tilde{d}_i^{j,k} = L^{j,k}/r^j + \delta^{j,k}$ and its virtual finish time is defined as $\tilde{\nu}_i^{j,k} = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k}$. Whereas, if $\mathcal{S}_i$ is delay-based, $\tilde{d}_i^{j,k} = d^j$ and $\tilde{\nu}_i^{j,k} = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k}$.

The *per-hop behavior* of a core router (or rather, its scheduler) is characterized by an *error term*, which is defined with respect to the virtual finish time and *actual* finish time of packets at the router. Let $\hat{f}_i^{j,k}$ denote the actual time packet $p^{j,k}$ departs the scheduler $\mathcal{S}_i$. We say that $\mathcal{S}_i$ can guarantee flow j its reserved rate r^j (if $\mathcal{S}_i$ is rate-based) or its delay parameter d^j (if $\mathcal{S}_i$ is delay-based) with an error term Ψ_i, if for any k, $\hat{f}_i^{j,k} \leq \tilde{\nu}_i^{j,k} + \Psi_i$. In other words, each packet of flow j is guaranteed to depart $\mathcal{S}_i$ by the time $\tilde{\nu}_i^{j,k} + \Psi_i = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k} + \Psi_i$.

Given the error term Ψ_i of the scheduler $\mathcal{S}_i$, the virtual time stamp of packet $p^{j,k}$ after it has traversed $\mathcal{S}_i$ is updated using the following *concatenation rule*:

$$\tilde{\omega}_{i+1}^{j,k} = \tilde{\nu}_i^{j,k} + \Psi_i + \pi_i = \tilde{\omega}_i^{j,k} + \tilde{d}_i^{j,k} + \Psi_i + \pi_i \quad (1)$$

where π_i denotes the propagation delay from the ith router to the next-hop router along the flow's path. In [20] it is shown that using the concatenation rule (1) the virtual spacing and reality check properties of virtual time stamps are satisfied at every router.

End-to-End Delay Bound and QoS Abstraction of Data Plane. Using the virtual time reference system outlined above, the delay experienced by packets of a flow across the network core can be upper bounded in terms of the rate-delay parameter pair of a flow and the error terms of the routers along the flow's path. Suppose there are total 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, we have

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

where $L^{j,max}$ is the maximum packet size of flow j.

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, 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

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

where $T_{on}^j = (\sigma^j - L^{j,max})/(P^j - \rho^j)$. Hence the end-to-end delay bound for flow j is given by

$$d_{e2e}^j = d_{edge}^j + d_{core}^j$$
$$= T_{on}^j \frac{P^j - r^j}{r^j} + (q+1)\frac{L^{j,max}}{r^j} + (h-q)d^j + \sum_i(\Psi_i + \pi_i) \quad (4)$$

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 per-hop behavior (i.e., its ability to support delay guarantees) of a core router can be characterized using the notion of error term. This simple abstraction enables us to derive *end-to-end delay bounds* for flows traversing an arbitrary concatenation of core routers *within a network domain*. Furthermore, the virtual time reference system does *not* mandate any specific scheduling mechanisms to be implemented in a network domain as long as their abilities to provide delay guarantees can be characterized using the notion of error term. In fact, in [20] it is shown that almost all known scheduling algorithms can thus be characterized, be they *core stateless* or *stateful*. In addition, the virtual time reference system leads to the design of a set of new core stateless scheduling algorithms (both rate-based and delay-based). Two representative examples of such core stateless scheduling algorithms are: the rate-based *core stateless virtual clock* ($C_\diamondsuit$VC) and delay-based *virtual time earliest deadline first* (VT-EDF) scheduling algorithms.

The core stateless virtual clock ($C_\diamondsuit$VC) is a work-conserving counterpart of the CJVC scheduling algorithm developed in [12]. It services packets in the order of their virtual finish times. It is shown in [20] that as long as the total reserved rate of flows traversing a $C_\diamondsuit$VC scheduler does not exceed its capacity (i.e., $\sum_j r^j \leq C$), then the $C_\diamondsuit$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 $C_\diamondsuit$VC scheduler.

Unlike the conventional rate-controlled EDF, VT-EDF supports delay guarantees *without per-flow rate control*, and thus is core stateless. Like $C_\diamondsuit$VC, VT-EDF also services packets in the order of their virtual finish times. 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 [20]:

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

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 The Admission Control Module: QoS State Information Bases and Basic Operations

In order to support guaranteed services under the proposed bandwidth broker architecture, the admission control module maintains several QoS state information bases. They include: 1) a *flow information base*, where 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., end-to-end delay requirement D^j), and QoS reservation (e.g., the rate-delay parameter pair $\langle r^j, d^j \rangle$), is maintained; 2) a *node QoS state information base*, where information regarding each router in the network domain, such as the bandwidth, buffer capacity, type of scheduler used (i.e., rate- or delay-based) and its error term, and current QoS reservation on each outgoing link of the router, is maintained ; and 3) a *path QoS state information base*, where information regarding the characteristics of paths between ingress and egress routers is maintained. Examples of path characteristics are the hop number of a path, the sum of the router error terms and propagation delay along the path, and the minimal residual bandwidth along the path. As we shall see in the next section, maintaining separate path-level QoS state information allows us to perform efficient admissibility test *on an entire path basis*.

We now briefly discribe the basic operations of the BB, focusing, in particular, on those of the admission control module (see Figure 1). When a new flow with a traffic profile $(\sigma^j, \rho^j, P^j, L^{j,max})$ and an end-to-end 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 information base to determine whether the new flow is admissible. If not, the request is immediately rejected. Otherwise, the BB selects a path (from the ingress router to an appropriate egress router in the network domain) for the new flow, and decide whether the flow can be admitted. Generally speaking, the admission control procedure consists of two phases: 1) *admissibility 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) *bookkeeping* phase during which the relevant management information bases will be updated, if the flow is admitted. If the flow is admitted, the BB will also pass (using, e.g., COPS [3]) the QoS reservation information such as $\langle r^j, d^j \rangle$ to the ingress router, so that it can set up a new or re-configure an existing edge conditioner (which is assumed to be co-located at the ingress router) for the new flow. As the packets of the flow arrive at the ingress router, the edge conditioner will appropriately initialize and insert these packet states, before injecting them into the core of the network domain.

In the remainder of this paper we demonstrate how efficient admission control can be performed under the proposed BB architecture. In Section 3 we present a *path-oriented approach* to perform efficient admission control operations for support of per-flow guaranteed services. 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 simultaneously*, and makes admission control decision accordingly. Clearly, such a path-oriented approach is only possible because the availability of QoS state information of the entire path at the BB. Note that although any bandwidth broker architecture can adopt a similar path-oriented approach to resource allocation, under our BB architecture *no* local admission control test or explicit resource (re-)configuration is needed at any core router, thanks to the use of dynamic packet state in the data path. As a result, we can significantly reduce the time of conducting admission control and resource reservation. Furthermore, we can also perform path-wide optimization when determining resource allocation for a new flow. In Section 4, we study the problem of admission control for class-based guaranteed delay services, where a fixed number of guaranteed delay service classes are offered in a network domain. As in the DiffServ, there are two important benefits in providing class-based services under our framework. First, aggregating flows in a small number of service classes greatly simplifies the implementation of the data plane (in particular, the edge conditioners). Second, it also enhances

the scalability of the QoS control plane (i.e., the BB architecture). For example, the number of QoS states that need to be maintained by a BB can be significantly reduced. This leads to faster admission control operations, thereby increasing the number of flow service requests a BB can handle. However, as pointed out in the introduction, in class-based guaranteed services certain transient behavior of dynamic flow aggregation must be taken care of to avoid potential delay bound violation. In Section 4, we investigate the problem of dynamic flow aggregation and device simple mechanisms to effectively circumvent this problem.

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. In particular, we present a *path-oriented* approach to perform efficient admission control operations. We illustrate how this approach works by first considering a simple case, where only rate-based schedulers are employed along the path of a new flow. We then look at the general case where the path of a new flow consists of both rate-based and delay-based schedulers. We present an efficient admission control algorithm that 1) determines whether a new flow is admissible by examining the resource constraints along the entire path simultaneously, and 2) minimizes the bandwidth allocation along the path for the new flow, if it is admissible.

3.1 Path with Only Rate-based Schedulers

We first consider the 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* ($C_\not{S}$VC) or *core-jitter virtual clock* (CJVC) is employed at the routers $\mathcal{S}_i$ along $\mathcal{P}$. Let $\mathcal{F}_i$ be the set of flows currently traversing $\mathcal{S}_i$, and C_i be the total bandwidth at $\mathcal{S}_i$. Then as long as $\sum_{j\in\mathcal{F}_i} r^j \le C_i$, $\mathcal{S}_i$ can guarantee each flow j its reserved rate r^j. We use $C_{res}^{\mathcal{S}_i}$ to denote the residual bandwidth at $\mathcal{S}_i$, i.e., $C_{res}^{\mathcal{S}_i} = C_i - \sum_{j\in\mathcal{F}_i} r^j$.

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 end-to-end 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 end-to-end delay requirement $D^{\nu,req}$, the reserved rate r^ν for the new flow ν must satisfy:

$$\rho^\nu \quad \le \quad r^\nu \le P^\nu$$

and

$$D^{\nu,req} \ge T_{on}^\nu \frac{P^\nu - r^\nu}{r^\nu} + (h+1)\frac{L^{\nu,max}}{r^\nu} + D_{tot}^{\mathcal{P}}, \quad (6)$$

where $T_{on}^\nu = (\sigma^\nu - L^{\nu,max})/(P^\nu - \rho^\nu)$ and $D_{tot}^{\mathcal{P}} = \sum_{i\in\mathcal{P}}(\Psi_i + \pi_i)$.

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

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

$$r_{fea}^{low} = \max\{\rho^\nu, r_{min}^\nu\} \text{ and } r_{fea}^{up} = \min\{P^\nu, C_{res}^{\mathcal{P}}\}.$$

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

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. Because of the inter-dependence of the reserved rate r^ν and the delay parameter d^ν in the end-to-end delay bound (4) as well as the more complex schedulability condition (5) for the delay-based schedulers, the admissibility test for this case is less straightforward. In this section, we present an efficient admission control algorithm using the path-oriented approach that minimizes the bandwidth allocation along the path of the new flow: if the new flow ν is admissible, this admission control algorithm finds a feasible rate-delay parameter pair $\langle r^\nu, d^\nu\rangle$ such that r^ν is the *minimal* feasible rate. In other words, no other rate-delay parameter pair $\langle r^{\nu'}, d^{\nu'}\rangle$ such that $r^{\nu'} < r^\nu$ is feasible.

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 $C_\not{S}$VC (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).

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 end-to-end 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 end-to-end delay requirement $D^{\nu,req}$ must be satisfied, namely,

$$\rho^\nu \quad \le \quad r^\nu \le P^\nu$$

and

$$D^{\nu,req} \ge T_{on}^\nu \frac{P^\nu - r^\nu}{r^\nu} + (q+1)\frac{L^{\nu,max}}{r^\nu} + (h-q)d^\nu + D_{tot}^{\mathcal{P}}. \quad (7)$$

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

```
0.   t^ν = 1/(h−q)[D^{ν,req} − D^P_tot + T^ν_on]
1.   Let m* such that d^{m*−1} < t^ν ≤ d^{m*}
2.   for m = m*, m* − 1, ..., 2, 1
3.     R^m_fea ← [r^{m,l}_fea, r^{m,r}_fea]
4.     R^m_del ← [r^{m,l}_del, r^{m,r}_del]
5.     if (R^m_fea ∩ R^m_del == ∅)
6.       if (R^m_fea == ∅ || R^m_del == ∅ || r^{m,r}_fea < r^{m,l}_del)
7.         break with d^ν = d^m
8.       else /* R^m_fea ∩ R^m_del ≠ ∅ */
9.         if (r^{m,l}_fea < r^{m,l}_del)
10.          r^ν ← r^{m,l}_del, d^ν ← t^ν − Ξ^ν / r^ν
11.        break with d^ν
12.  if (d^ν > t^ν) no feasible value found
13.  else return d^ν
```

Figure 4: Path-oriented admissibility test for mixed rate/delay-based schedulers.

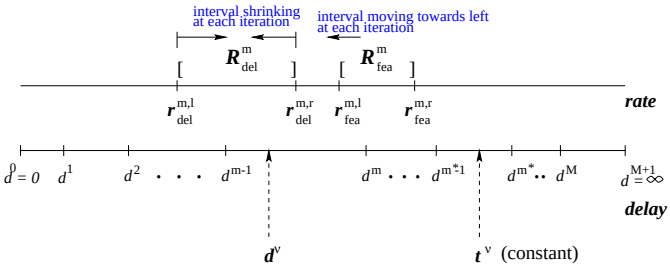

Figure 5: The relationship between feasible range $\mathcal{R}^m_{fea}$ and delay constraint range $\mathcal{R}^m_{del}$.

$k \in \mathcal{F}_i$, $\mathcal{S}_i \in \mathcal{P}$ such that $d^k_i \geq d^\nu$, we must have

$$\sum_{\{j \in \mathcal{F}_i : d^j_i \leq d^k_i\}} [r^j(d^k_i - d^j_i) + L^{j,max}] + [r^\nu(d^k_i - d^\nu) + L^{\nu,max}] \leq C_i d^k_i. \quad (8)$$

Hence, 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).

In the following, we present an efficient algorithm to identify the feasible rate-delay parameter pair $\langle r^\nu, d^\nu \rangle$ for a new flow ν. First, we introduce some notation. Define $t^\nu = \frac{1}{h-q}(D^{\nu,req} - D^{\mathcal{P}}_{tot} + T^\nu_{on})$, $\Xi^\nu = \frac{1}{h-q}[T^\nu_{on} P^\nu + (q+1)L^{\nu,max}]$. From (7) we see that the following condition holds,

$$d^\nu \leq t^\nu - \frac{\Xi^\nu}{r^\nu}. \quad (9)$$

For any flow k traversing a delay-based scheduler $\mathcal{S}_i$ such that $d^k_i \geq d^\nu$, define

$$S^k_i = C_i d^k_i - \sum_{\{j \in \mathcal{F}_i : d^j_i \leq d^k_i\}} [r^j(d^k_i - d^j_i) + L^{j,max}].$$

Let $\mathcal{F}^{del}$ be the union of the sets of the flows at all the delay-based schedulers along the path $\mathcal{P}$ of flow ν, 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

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

Note that S^k_i denotes the *minimum residual service* over any time interval of length d^k_i at scheduler $\mathcal{S}_i$. Therefore S^m represents 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*.

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 (9), 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.

For $m = m^*, m^* - 1, \ldots, 2, 1$, define $\mathcal{R}^m_{fea} = [r^{m,l}_{fea}, r^{m,r}_{fea}]$, where

$$r^{m,l}_{fea} = \max\{\frac{\Xi^\nu}{t^\nu - d^{m-1}}, \rho^\nu\},$$
$$r^{m,r}_{fea} = \min\{\frac{\Xi^\nu}{t^\nu - d^m}, P^\nu, C^{\mathcal{P}}_{res}\}. \quad (10)$$

Similarly, define $\mathcal{R}^m_{del} = [r^{m,l}_{del}, r^{m,r}_{del}]$, where

$$r^{m,r}_{del} = \min\left\{\min_{m \leq k < m^*}\{\frac{\Xi^\nu + L^{\nu,max}}{t^\nu - d^k}\}, \min_{k \geq m^*}\frac{S^k - \Xi^\nu - L^{\nu,max}}{d^k - t^\nu}\right\},$$
$$r^{m,l}_{del} = \max_{m \leq k < m^*}\{\frac{S^k - \Xi^\nu - L^{\nu,max}}{d^k - t^\nu}\} \quad (11)$$

$\mathcal{R}^m_{fea}$ and $\mathcal{R}^m_{del}$ have the following important monotonicity properties, as illustrated in Figure 5. 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 $\mathcal{R}^m_{fea}$ determined by (10) shifts to the left. In contrast, the corresponding feasible rate range $\mathcal{R}^m_{del}$ determined by the delay constraints (11) shrinks: the left edge $r^{m,l}_{del}$ increases while the right edge $r^{m,r}_{del}$ decreases. (For an explanation why these properties hold, see [21].) Based on this observation, we obtain the following theorem, which states whether a feasible rate-delay pair $\langle r^\nu, d^\nu \rangle$ exists such that $d^\nu \in [d^{m-1}, d^m]$.

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 $d^\nu \geq 0$ 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. ∎

Based on Theorem 1, the admission control algorithm is presented (in pseudo-code) in Figure 4. Note that the time complexity of the algorithm is $O(M)$, and 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 pre-defined. Clearly this reduction in complexity is achieved because our admission control algorithm considers all the admissibility constraints along a path simultaneously. This is *not* possible using the conventional hop-by-hop reservation set-up approach (e.g., as employed in the IETF IntServ model with RSVP).

4. CLASS-BASED GUARANTEED SERVICES AND ADMISSION CONTROL WITH DYNAMIC FLOW AGGREGATION

In this section we address the problem of admission control for class-based guaranteed services under the proposed BB architecture. The class-based guaranteed delay service model is schematically shown in Figure 6. A new user flow will be placed in one of the delay service classes if it can be 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 end-to-end delay bound determined by the service class. We refer to the individual user flows constituting a macroflow as the *microflows*.

A key issue in the design of admission control for this class-based service model is the problem of *dynamic* flow aggregation. The dynamics comes 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. Dynamic flow aggregation can cause certain undesirable effect on the end-to-end delay experienced by the macroflow, which we will describe shortly in Section 4.1. Here we note that the existing work on traffic aggregation (in particular, in the context of guaranteed services, see, e.g. [6, 9]) 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. As far as we are aware, this problem of dynamic flow aggregation has *not* been identified nor addressed before in the literature.

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

In this section we illustrate the potential negative impact of dynamic flow aggregation on end-to-end delay guarantee provisioning: if proper care is not taken, microflow arrivals or departures in an aggregate macroflow can cause potential delay bound violation. We use an example to illustrate this problem of dynamic flow aggregation. First let us 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 $L^{\alpha,max} = \sum_{j=1}^n L^{j,max}$, as a packet of the maximum size may arrive from each of the n microflow at the same 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 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 that 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 end-to-end delay formula (4), the end-to-end delay experienced by packets from the macroflow α is bounded by

$$\begin{aligned} d^\alpha_{e2e} &= d^\alpha_{edge} + d^\alpha_{core} \\ &= T^\alpha_{on} \frac{P^\alpha - r^\alpha}{r^\alpha} + \frac{L^{\alpha,max}}{r^\alpha} + h\frac{L^{\mathcal{P},max}}{r^\alpha} + D^{\mathcal{P}}_{tot} \quad (12) \end{aligned}$$

where $D^{\mathcal{P}}_{tot} = \sum_{i \in \mathcal{P}} (\Psi_i + \pi_i)$.

Assume that 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 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 Figure 7. In this scenario, $T^\alpha_{on} \geq T^\nu_{on}$, and thus $T^\nu_{on} \leq T^{\alpha'}_{on} \leq T^\alpha$. 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 $t \geq t^*$, $A^\nu(t^*, t) = \mathcal{E}^\nu(t - t^*) = \min\{P^\nu(t - t^*) + L^{\nu,max}, \rho^\nu(t - t^*) + \sigma^\nu\}$. 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

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

Hence the delay experienced by a packet arriving 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

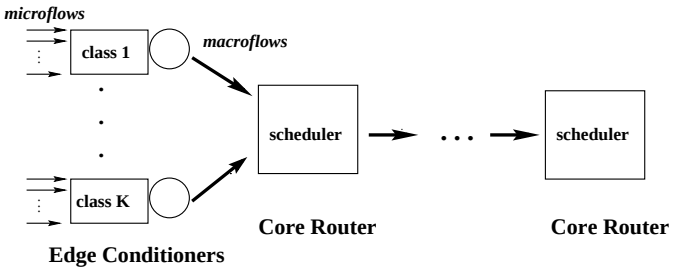

Figure 6: Class-based guaranteed services: dynamic flow aggregation along a path.

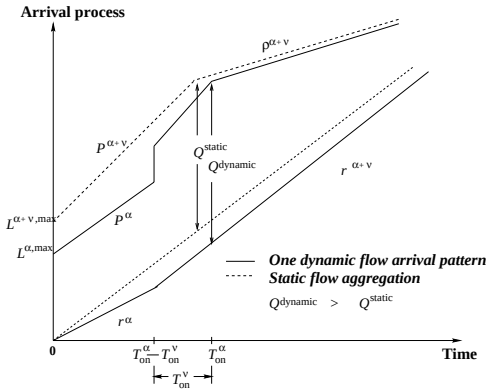

Figure 7: An example illustrating the edge delay bound violation after rate change in dynamic flow aggregation.

fact that packets from the "new" macroflow α' are serviced with a higher reserved rate $r^{\alpha'}$ ($\geq r^{\alpha}$), 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. 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 end-to-end delay that is no longer bounded by the end-to-end delay formula (12). A similar situation may occur when a constituent microflow leaves an existing macroflow, *if we immediately decrease the reserved rate r^{α} to a new lower reserved rate $r^{\alpha'}$* (see [21] for details).

Before we leave this section, 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. It appears that this problem might be more difficult to address using the conventional hop-by-hop reservation set-up approach. In the following we show how this problem can be solved using relatively simple mechanisms under the proposed BB architecture.

4.2 End-to-End 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.

4.2.1 Contingency Bandwidth and Edge Delay Bound

We introduce the notion of *contingency bandwidth* to eliminate the lingering delay effect of the backlog queued in the edge conditioner at the time a microflow is aggregated into or de-aggregated from an existing macroflow. It 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 packets from the "new" macroflow α' after time t^* is bounded above by

$$d^{new}_{edge} \quad \leq \quad \max\{d^{old}_{edge}, d^{\alpha'}_{edge}\}. \tag{13}$$

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

The following two theorems state the sufficient conditions on Δr^{ν} and τ^{ν} so that (13) holds (see [21] for their proofs).

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 (13) holds if*

$$\Delta r^{\nu} \geq P^{\nu} - r^{\nu} \text{ and } \tau^{\nu} \geq \frac{Q(t^*)}{\Delta r^{\nu}}. \tag{14}$$

■

THEOREM 3 (MICROFLOW LEAVE). *Suppose at time t^* a 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 (13) holds if*

$$\Delta r^{\nu} \geq r^{\nu} \text{ and } \tau^{\nu} \geq \frac{Q(t^*)}{\Delta r^{\nu}}. \tag{15}$$

■

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

$$Q(t^*) \leq d_{edge}^{old} r(t^*) \quad = \quad d_{edge}^{old}(r^\alpha + \Delta r^\alpha(t^*)) \qquad (16)$$

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:

$$\hat{\tau}^\nu \quad = \quad d_{edge}^{old} \frac{r^\alpha + \Delta r^\alpha(t^*)}{\Delta r^\nu}. \qquad (17)$$

Hence after $\hat{\tau}^\nu$, the BB can de-allocate 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. In general this scheme can be quite *conservative*.

A more *practical* approach is to have the edge conditioner to *feedback the actual contingency period* to the BB based on the current buffer occupancy of the macroflow at the edge conditioner. We refer to this scheme as the *contingency feedback* method. Note also that whenever the buffer in the edge conditioner becomes empty, the lingering delay effect of the old macroflow has gone away. Hence, the maximum delay experienced by any packets of the macroflow α is bounded by d_{edge}^{α}, which is solely determined by the *current* aggregate traffic profile of the macroflow α. Therefore, the edge conditioner can send a message to the BB to reset *all* of the contingency bandwidth allocated to the macroflow α (i.e., setting $\Delta r^\alpha = 0$) before a contingency period expires.

4.2.2 Extension to VTRS and Core Delay Bound

As discussed in Section 4.1, packets of the new macroflow may catch up with the packets of the old macroflow inside the network core and hence experience additional queueing delay. In this section we illustrate how the virtual time reference system can be extended to accommodate flow aggregation with dynamic rate changes. Based on this extension, we present a modified core-delay bound for flow aggregation.

Consider an existing macroflow α which traverses the path $\mathcal{P}$ where there are q rate-based schedulers and $h-q$ delay-based schedulers. In order to simplify the derivation of the delay experienced by packets inside a network core, we impose an assumption: *the delay parameter d^α associated with a macroflow α is fixed, no matter whether there are microflow arrivals or departures in the macroflow.*

Suppose that at time τ^*, the reserved rate of the macroflow α is adjusted at the edge shaper from r to r'. 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 the packets are shaped as follows. For $k < k^*$, $\hat{a}_1^{k+1} - \hat{a}_1^k \geq L^{k+1}/r$, and for $k \geq k^*$, $\hat{a}_1^{k+1} - \hat{a}_1^k \geq L^{k+1}/r'$. Furthermore, we need to ensure that the virtual time adjustment term that is carried inside packet headers is properly calculated at the network edge. Due to space limitation, we omit the details here, and refer the interested reader to [21]. The following theorem states a modified delay bound inside the network core for packets of a macroflow.

THEOREM 4. *Let packets of one macroflow shaped at the network edge as follows. For $k < k^*$, $\hat{a}_1^{k+1} - \hat{a}_1^k \geq L^{k+1}/r$, and for $k \geq k^*$, $\hat{a}_1^{k+1} - \hat{a}_1^k \geq L^{k+1}/r'$. Moreover, let the virtual time adjustment term be properly defined as in [21]. Then the virtual spacing and reality check properties hold for the macroflow after the rate change at τ^*. Furthermore, the delay experienced by these packets in the network core is bounded by the following modified core delay formula:*

$$\hat{f}_h^k - \hat{a}_1^k \leq q \max\left\{\frac{L^{\mathcal{P},max}}{r}, \frac{L^{\mathcal{P},max}}{r'}\right\} + (h-q)d^\alpha + D_{tot}^{\mathcal{P}}. \qquad (18)$$

■

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 end-to-end 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 end-to-end 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 $\mathcal{P}$ be the path of macroflow α, and let $C_{res}^{\mathcal{P}}$ be the minimal residual bandwidth along path $\mathcal{P}$. We consider the cases of microflow join and leave separately below.

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, 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 end-to-end delay requirement $D^{\alpha,req}$ can be satisfied for the resulting macroflow α'. Note that after contingency period, the edge queueing delay for any packets of the class is determined by the new class traffic profile and the reserved rate, therefore,

$$d_{e2e}^{\alpha'} = d_{edge}^{\alpha'} + \max\{d_{core}^{\alpha}, d_{core}^{\alpha'}\} \leq D^{\alpha,req}. \qquad (19)$$

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 (19) is reduced to $d_{edge}^{\alpha'} \leq D^{\alpha,req} - d_{core}^{\alpha}$. 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'.

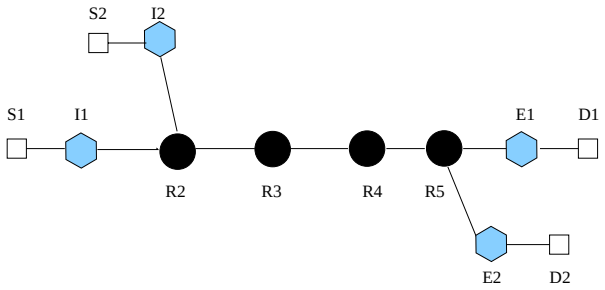

Figure 8: The network topology used in the simulations

Microflow Leave. From Theorem 3, when a constituent flow ν leaves the macroflow α at time t^*, we must continue to service the macroflow α with the current reserved rate r^α for a period of τ^ν with a contingency bandwidth $\Delta r^\nu = r^\nu$. 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 (19) holds. Since $r^{\alpha'} \leq r^\alpha$, we have $d^\alpha_{core} \leq d^{\alpha'}_{core}$. Therefore, (19) is reduced to $d^{\alpha'}_{edge} + d^{\alpha'}_{core} \leq D^{\alpha,req}$, from which we can find a new reserved rate $r^{\alpha'}$ for the new macroflow.

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.

Figure 8 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_SVC 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_SVCs, 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 [5, 11] used for the GS in the IntServ model. In the GS model, the counterpart of a C_SVC scheduler is VC, while for VT-EDF, it is RC-EDF. The RC-EDF [5, 18] 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 end-to-end delay requirement (either 2.44 or 2.19). 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 $2.44s$, a reserved rate equal to its mean sending rate will meet the delay requirement. Whereas, when the delay requirement is $2.19s$, 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 end-to-end delay requirement of the class is set to either either 2.44 or 2.19. 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, 0.24 and 0.50). The objective of our simulation investigation is to compare what is 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.

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 end-to-end delay requirements of the flows. When the delay requirement is $2.44s$, the Aggr BB/VTRS scheme accepts one fewer flow than that 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 contingency 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 $2.19s$, 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; and 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 end-to-end delay requirement of the flows is 2.19, we examine more closely the bandwidth allocation allocated under the three schemes. Figure 9 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

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		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	29	29	29	29
	cd = 0.24			29	29
	cd = 0.50			29	28

service class. (Note in particular that with the fixed delay parameter $cd = 0.10$, a per-flow bandwidth allocation that is equal to the mean rate of the flows is sufficient to support the end-to-end delay bound 2.19 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, 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 slightly 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 life time. 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 simulations, 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 inter-arrival 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. Figure 10 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 5 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 de-allocated 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. We also observe that 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 contingency 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. The proposed bandwidth broker architecture is designed based on a *core stateless* virtual time reference system developed in [20]. Using the proposed bandwidth broker architecture, we designed efficient admission control algorithms for both *per-flow* end-to-end guaranteed delay services and *class-based* guaranteed delay services with flow aggregation. In particular, we demonstrated how admission control can be done on an entire *path* basis, instead of on a "hop-by-hop" basis. 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.

In this paper we have demonstrated the potential advantages of the proposed bandwidth broker architecture from the per-

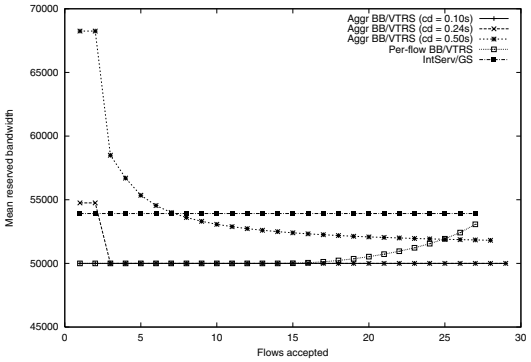

Figure 9: Mean reserved bandwidth.

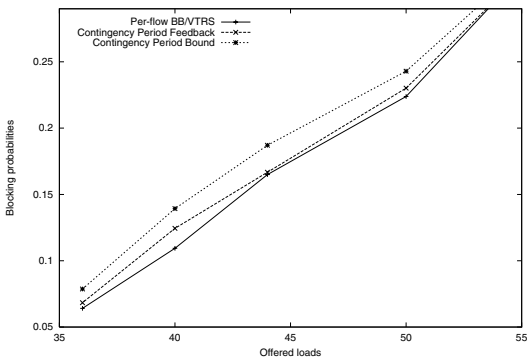

Figure 10: Flow blocking rates.

spective of admission control. There are still many challenging problems, both theoretical and practical, in the design and implementation of such a BB architecture. Currently we are investigating a number of approaches to the design of a distributed/hierarchical bandwidth broker architecture. We are also exploring ways to extend our virtual time reference system framework and the proposed BB architecture to support statistical and other forms of QoS guarantees.

Acknowledgement: We are extremely grateful to Roch Guerin for his insightful comments and valuable suggestions that have greatly improved the presentation of this paper. We would also like to thank the anonymous reviewers for their helpful feedback on the paper.

7. REFERENCES

[1] P. Aukia, M. Kodialam, P. Koppol, T. Lakshman, H. Sarin, and B. Suter. RATES: A server for MPLS traffic engineering. *IEEE Network*, pages 34–41, March/April 2000.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services. RFC 2475, December 1998.

[3] J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, and A. Sastry. The COPS (common open policy service) protocol. RFC 2748, January 2000.

[4] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource reservation protocol (RSVP) – version 1 functional specification. RFC 2205, September 1997.

[5] L. Georgiadis, R. Guérin, V. Peris, and K. N. Sivarajan. Efficient network QoS provisioning based on per node traffic shaping. *IEEE/ACM Transactions on Networking*, 4(4):482–501, 1996.

[6] R. Guérin, S. Blake, and S. Herzog. Aggregating RSVP-based QoS requests. Internet Draft, 1997. Work in Progress.

[7] K. Nichols, V. Jacobson, and L. Zhang. A two-bit differentiated services architecture for the internet. RFC 2638, July 1999.

[8] P. Pan and H. Schulzrinne. YESSIR: a simple reservation mechanism for the Internet. *ACM Computer Communication Review*, 29(2):89–101, April 1999.

[9] S. Rampal and R. Guérin. Flow grouping for reducing reservation requirements for guaranteed delay service. Internet Draft, July 1997. Work in Progress.

[10] E. C. Rosen, A. Viswanathan, and R. Callon. Multiprotocol label switching architecture. Internet Draft, August 1999. Work in Progress.

[11] S. Shenker, C. Partridge, and R. Guérin. Specification of guaranteed quality of service. RFC 2212, September 1997.

[12] I. Stoica and H. Zhang. Providing guaranteed services without per flow management. In *Proc. ACM SIGCOMM*, Boston, MA, September 1999.

[13] 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, February 1999. Work in Progress.

[14] A. Terzis, J. Ogawa, S. Tsui, L. Wang, and L. Zhang. A prototype implementation of the two-tier architecture for differentiated services. In *Proceedings of IEEE RTAS'99*, Vancouver, Canada, 1999.

[15] J. Y. Le Boudec, T. Ferrari, W. Almesberger. SRP: a scalable resource reservation protocol for the internet. In *Proceedings of IWQoS'98*, pages 107–116, Napa, CA, May 1998.

[16] L. Wang, A. Terzis, and L. Zhang. A new proposal of RSVP refreshes. In *Proceedings of IEEE ICNP*, Toronto, Canada, November 1999.

[17] L. Wang, A. Terzis, and L. Zhang. RSVP refresh overhead reduction by state compression. Internet Draft, June 1999. Work in Progress.

[18] H. Zhang and D. Ferrari. Rate-controlled static-priority queueing. In *IEEE INFOCOM'93*, pages 227–236, April 1993.

[19] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. RSVP: A new resource reservation protocol. *IEEE Network*, pages 8–18, September 1993.

[20] Z.-L. Zhang, Z. Duan, and Y. T. Hou. Virtual time reference system: A unifying scheduling framework for scalable support of guaranteed services. *IEEE Journal on Selected Areas in Communication,* Special Issue on Internet QoS, To appear 2000.

[21] Z.-L. Zhang, Z. Duan, 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 00-028*, Computer Science Department, University of Minnesota, March 2000. http://www.cs.umn.edu/~zhzhang/papers.html.