Link-Based Fair Aggregation: A Simple Approach to Scalable Support of Per-Flow Service Guarantees

Yuming Jiang

Center for Quantifiable QoS in Communication Systems (Q2S)
Norwegian University of Science and Technology (NTNU)
ymjiang@ieee.org

Abstract. To support service guarantees in packet-switched networks, three approaches have been proposed. They are the Stateless Core (SCORE) approach, the Integrated Services (IntServ) approach, and the Differentiated Services (DiffServ) approach. The granularities of service guarantees provided by these approaches at each router are respectively *packet level*, *flow level*, and *class level*. In this paper, we propose a novel approach, called *Link-Based Fair Aggregation* (LBFA) approach to scalable support of service guarantees. While the granularity of service guarantees supported by LBFA is *link level* at each router, we show through analysis that the proposed LBFA approach can achieve as good as or even better *per-flow* service guarantees than the current three approaches.

1 Introduction

The Internet was initially designed to provide one simple service: best-effort datagram delivery. Such a design allows routers to be *stateless* and to forward packets in a First-In-First-Out (FIFO) manner. As a consequence, today's Internet is highly *scalable* in the sense that router complexity does not increase with the number of flows in the network. However, with the development and deployment of multimedia and network technologies, multimedia has become an indispensable feature of the Internet. Unlike traditional applications such as file transfer, many multimedia applications such as Internet telephony are delay-sensitive. Thus, there is a demand for introducing a service in the Internet with which both bandwidth and delay guarantees are provided.

Chronologically, three approaches have been proposed in the literature to provide such services in addition to best-effort service, which are the Integrated Services (IntServ) approach [4], Differentiated Services (DiffServ) approach [3], and Stateless Core (SCORE) approach [20]. Specifically, the Guaranteed service [19] in IntServ, the Expedited Forwarding service [7] in DiffServ, and the feature of providing guaranteed services in SCORE [20] are for this purpose. These approaches have important differences in achieving these services. In particular, while the IntServ approach can provide *end-to-end flow level* service guarantees, it is *stateful* in the sense that every router needs to maintain per-flow states.

The DiffServ approach is *core-stateless* since per-flow states are only maintained at edge routers. However, service guarantees under the DiffServ approach are provided only to aggregate *class level* and such guarantees are mainly defined for the *per-hop case*. Additional effort is needed to make DiffServ support end-to-end per-flow service guarantees. Like the DiffServ approach, the SCORE approach is also *core-stateless* in providing scalable support of guaranteed services, but it achieves this in a different way by letting each packet carry *packet state* and each router forward the packet by a deadline calculated based on the carried state. Hence, *packet level* service guarantees at each router and consequently end-to-end per-flow service guarantees are provided by the SCORE approach.

In this paper, we propose a novel approach, the Link-Based Fair Aggregation (LBFA) approach, to provide scalable support of per-flow service guarantees. This approach is inspired by a natural phenomenon in the network: a flow is aggregated with other flows on each link along its transit path. The idea behind LBFA comes from the conjecture that if a flow is properly aggregated with other flows of the same traffic class on each link along the path, it is possible to preserve service guarantees end-to-end to the flow by aggregating the corresponding aggregates on each link properly and providing service guarantees to the formed *link level* aggregate at each router along the path.

In the proposed LBFA approach, except for a couple of fixed or pre-configured parameters associated with each router, no per-flow information is maintained in the core as is the case for the core-stateless DiffServ and SCORE approaches. In addition, as opposed to the SCORE and DiffServ approaches, the LBFA approach does not mandate maintaining per-flow states at edge routers. Hence, the LBFA approach can be *stateless*. In addition, as opposed to the SCORE approach, no additional information needs to be added to each packet in the LBFA approach. In the LBFA approach, service guarantees at each router are provided to the aggregate of flows of the same traffic class from the same incoming link. In this sense, the LBFA approach is said to provide *link level* service guarantees at the router. In the paper, we show through analysis that end-to-end per-flow service guarantees can be provided if LBFA is implemented in the network. These guarantees are as good as or even better than those provided by the three existing approaches.

2 Network Model and Scheduling Model

2.1 Network Model

We consider a single multi-service network domain with feedforward routing. Routers are building blocks of the network. In the network, flows are partitioned based on the service classes that they belong to. The service discipline at each output port of a router allocates resources of the corresponding output link among different service classes in a link-sharing manner [9]. For each scheduler at the scheduling hierarchy of such link-sharing [9], its buffer, if there is any, is assumed to operate in FIFO manner and its size is large enough to ensure no packet drop.

When entering the network, every flow is shaped at the edge before releasing into the network. Packets of the flow are transmitted in the network along a single path, which is modeled as a list of link servers. These packets traverses the path in the FIFO order so that the ordering of packets in the flow is preserved at every router along the path. In this paper, we are interested in the end-to-end service guarantees provided to the flow.

All routers in the network are assumed to be output-buffered and implement aggregate class-based scheduling. Fig. 1 depicts the architecture of a typical router. The router possibly has multiple input links and multiple output links. For each packet that arrives on an input link, the router determines the next hop on its path and transmits the packet on the corresponding output link. At each output link, a certain percentage of bandwidth is reserved for each traffic class according to some link-sharing policy. Throughout the rest of the paper, we use *link server* to represent the whole link-sharing scheduling hierarchy. Specifically, the link server includes both the class-based flow aggregation part and the class-based scheduling part shown in Fig. 1.

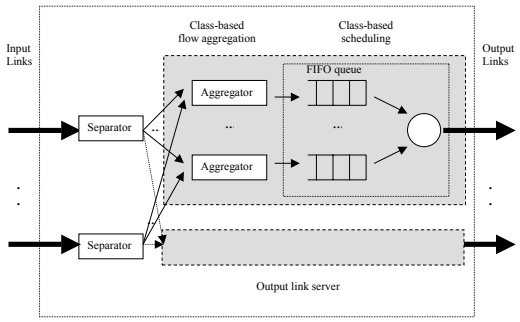

Fig. 1. Architecture of a typical router

Consider a flow f of a certain traffic class c. Let $\mathcal{F}_s$ denote the set of all class c flows constituting the class c aggregate on link server s along the path of flow f; I_s the number of incoming links to the router with output link s; R_s the reserved bandwidth for class c traffic on link s. Assume f is shaped before entering the network to conform to a token bucket with parameters (r^f, σ^f) where r^f is the token arrival rate and σ^f is the bucket size.

We further make the following assumptions for later analysis in the paper. 1) For ease of exposition, the delay experienced by a packet at the edge traffic shaper and the propagation delay are excluded from the end-to-end delay. 2) The amount of class c traffic on any link s does not exceed a certain ratio $\alpha_s (\leq 1)$ of the reserved rate R_s. Specifically, we require that for any link s in the network $\sum_{f \in \mathcal{F}_s} r^f \leq \alpha_s R_s$. 3) For any link s, let $\beta_s = \frac{1}{R_s} \sum_{f \in \mathcal{F}_s} \sigma^f$ and β be a bound on β_s. 4) Any flow in the network is assumed to traverse at most H hops. In other

words, the network diameter is H. 5) There exists a bound on the size of any packet in the network, which is denoted by L. 6) No packet delay is introduced by traffic separator. 7) Finally, we adopt the convention that a packet has been received/transmitted if and only if its last bit has been received/transmitted.

2.2 Scheduling Model

Each scheduler is assumed to be Guaranteed Rate (GR) server [10], which includes the class-based scheduler in Fig. 1 and the corresponding link server.

GR server is defined based on the *guaranteed rate clock* (GRC) value of a packet [10]. Let $p^{f,j}$ be the jth packet of flow f, $l^{f,j}$ be its length, and r_s^f be the bandwidth allocated to the flow. Then, the GRC value for packet $p^{f,j}$ at server s, denoted by $F_s^{f,j}$, is iteratively defined as:

$$F_s^{f,j} = max\{a_s^{f,j}, F_s^{f,j-1}\} + \frac{l^{f,j}}{r_s^f}, \tag{1}$$

where $F_s^{f,0} = 0$, and $a_s^{f,j}$ denotes the arrival time of packet $p^{f,j}$ to the server.

Similar to $F_s^{f,j}$, we define a virtual time function for flow f and denote it by $F^{f,j}$ which is iteratively obtained by replacing r_s^f with r^f as:

$$F^{f,j} = max\{a_1^{f,j}, F^{f,j-1}\} + \frac{l^{f,j}}{r^f}. \tag{2}$$

The difference between $F_s^{f,j}$ and $F^{f,j}$ is that while the former is server-dependent, the latter is server-independent. In fact, if we view the whole network as a blackbox, then $F^{f,j}$ is the GRC for the blackbox. In this sense, $F^{f,j}$ can be considered as the end-to-end GRC function for the flow f across the network.

A server s is said to be GR server to flow f with rate r_s^f and error term E_s^f, iff it guarantees that any packet $p^{f,j}$ of the flow is transmitted by [10]

$$d_s^{f,j} \leq F_s^{f,j} + E_s^f \tag{3}$$

where E_s^f is a constant that depends on the scheduling algorithm.

Similarly, we say the network provides *per-domain rate guarantee* to flow f with rate r^f and error term E^f, iff for any packet $p^{f,j}$, it guarantees that

$$d_H^{f,j} \leq F^{f,j} + E^f \tag{4}$$

where E^f is a constant that depends on the link servers along its path and $d_H^{f,j}$ denotes the departure time of packet $p^{f,j}$ from the network.

For every GR scheduling algorithm, a corresponding *Core-Stateless GR* (CSGR) algorithm can be defined [15] [16]. In such a CSGR algorithm, it assigns the following $GRCore$ values to packets of any flow f and schedules packets in the increasing order of their $GRCore$ values at each link server s:

$$GRCore_1^{f,j} = F_1^{f,j}, \quad \text{and,} \tag{5}$$

$$GRCore_{s+1}^{f,j} = GRCore_s^{f,j} + \frac{L}{r^f} + E_{s+1}^f, \tag{6}$$

for $s \geq 1$. Conversely, the GR scheduler is said to be the corresponding GR scheduler of the CSGR scheduler [15] [16].

3 Current Approaches

3.1 The IntServ Approach

In the IntServ approach [4], service guarantees are provided end-to-end on a per-flow basis. In order to do so, this approach uses end-to-end signaling to set up flow classification and reservation states on each router along the path. Usually, the router implements per-flow scheduling for resource allocation among flows. In particular, per-flow fair queueing (PFFQ) is commonly adopted by the router for the class-based flow aggregation part shown in Fig. 1. We call the resulted IntServ *PFFQ-based IntServ*. The granularity of service guarantees provided by the PFFQ-based IntServ approach at each router is hence *flow level*.

The following Theorems 1 to 3 summarize the per-flow service guarantees provided by the PFFQ-based IntServ approach. Due to space limitation, their proofs are omitted and can be found from [14].

Theorem 1. [Rate Guarantee] *If in a network, the link server at each router s along the path of a flow f belongs to GR with rate $r_s^f (\geq r^f)$ and error term E_s^f, then the network guarantees that*

$$d_H^{f,j} \leq F^{f,j} + \sum_{s=1}^{H-1} \frac{L}{r_s^f} + \sum_{s=1}^{H} E_s^f \tag{7}$$

where H is the number of routers on the path.

Theorem 2. [Bounded Delay] *Under the same condition as Theorem 1, the end-to-end delay of any packet in the flow is bounded by:*

$$D^f = \frac{\sigma^f}{r^f} + \sum_{s=1}^{H-1} \frac{L}{r_s^f} + \sum_{s=1}^{H} E_s^f. \tag{8}$$

Theorem 3. [Throughput Guarantee] *Under the same condition as Theorem 1, if the source of flow f transmits packets at least at rate r^f, the network guarantees to the flow*

$$W^f(t_1, t_2) \geq r^f \left(t_2 - t_1 - \sum_{s=1}^{H} \frac{L}{r^f} - \sum_{s=1}^{H} E_s^f \right)^+, \tag{9}$$

where $W^f(t_1, t_2)$ denotes the work done by the network to the flow in any interval $[t_1, t_2]$ with $0 \leq t_1 \leq t_2$, and $(x)^+ \equiv max\{0, x\}$.

3.2 The DiffServ Approach

In the DiffServ approach [3], packets of flows are classified into a small fixed number of classes, such as the Expedited Forwarding (EF) class [7] and Assured Forwarding (AF) groups [11]. Complex per-flow classification is implemented

only at edge routers. In the core, routers provide service guarantees only on a class basis rather than a per-flow basis. The granularity of such a service guarantee at each router is hence *class level*. Since only edge routers need to maintain per-flow states and core routers do not, the DiffServ approach is *core-stateless*.

As oppose to end-to-end service guarantees for individual flows in the IntServ approach, the current DiffServ mainly supports per-hop service guarantees, i.e. Per-Hop Behaviors (PHBs), to each traffic class aggregate. The extension of per-hop guarantees to per-domain or end-to-end guarantees is still an undergoing work [13] [18].

To date, a lot of DiffServ PHBs implementations have emerged. A typical implementation is to aggregate packets of the same class in a single FIFO queue and service them in the order of their arrival times. In such implementations, the aggregator shown in Fig. 1 is actually virtual, since the buffer of the class-based scheduler has been assumed to operate in the FIFO manner. Clearly, FIFO aggregation results in a very simple implementation of DiffServ. Throughout the rest of the paper, we shall focus on the DiffServ approach with FIFO aggregation and call it *FIFO-based DiffServ*. In the following, we present per-flow service guarantees provided by the FIFO-based DiffServ approach.

As discussed above, the link-server of a router under FIFO-based DiffServ is indeed the class-based scheduling part shown in Fig. 1. Assume that each link-server s guarantees rate R_s to the corresponding class aggregate of flow f with error term E_s. Then, we have the following theorem which presents the bounded delay guarantee supported by the FIFO-based DiffServ approach. It has been proved in a previous work [12].

Theorem 4. **[Bounded Delay]** *If the following condition on link utilization is satisfied*

$$\alpha < min_s \frac{P_s}{(H-1)(P_s - R_s)^+ + R_s}, \tag{10}$$

then a bound on end-to-end delay for any flow f exists and is

$$D^f = \frac{H}{1-(H-1)u\alpha}(u\beta + E'), \tag{11}$$

where P_s denotes the capacity sum of all input links that have the considered traffic class input, $u_s = \frac{(P_s - R_s)^+}{P_s - \alpha R_s}$, $u = max_s \{u_s\}$, and

$$E' = max_s \left\{ \frac{(1-u_s)L_s}{R_s} + E_s \right\}. \tag{12}$$

Conversely with Theorem 4, we can get the following results for the FIFO-based DiffServ approach. Their proofs can be found from [14].

Theorem 5. **[Rate Guarantee]** *If condition (10) is satisfied, then the network guarantees that*

$$d_H^{f,j} \leq F^{f,j} + D^f \tag{13}$$

where D^f is determined by (11).

Theorem 6. **[Throughput Guarantee]** *Under the same condition (10), if the source of flow f transmits packets at least at rate r^f, then the network guarantees to the flow that*

$$W^f(t_1, t_2) \geq r^f(t_2 - t_1 - D^f - \frac{L}{r^f})^+. \tag{14}$$

where D^f is determined by (11).

3.3 The SCORE Approach

Like the DiffServ approach, the SCORE approach is also *core-stateless* in providing service guarantees [20]. However, it achieves this in a different way from the DiffServ approach. The key construct in the SCORE approach is the notion of *packet state* (PS) and the main ideal behind PS is to have packets carry per-flow states instead of having routers maintain the per-flow states [20].

The packet state is inserted by ingress edge routers which, as in the DiffServ approach, maintain per-flow states. In the core, a router processes each incoming packet based on the state carried by the packet and the router's internal state. Before forwarding the packet to the next hop, the core router may update both the packet state and its internal state. In such a way, PS coordinates actions of edge and core routers along the path of a flow to provide service guarantees to the flow. In fact, a router schedules packets even unaware of each individual flow. The router guarantees that each packet is forwarded to the next hop within a certain time limit that is computed from its carried PS. In this sense, the SCORE approach provides *packet level* service guarantees at each router. Like the DiffServ approach, the SCORE approach does not maintain per-flow states at core routers and hence achieves scalability of core routers.

The following results show that the same service guarantees are provided by a SCORE network as by a PFFQ-based IntServ network if each router in the SCORE network implements the corresponding core-stateless version of the GR algorithm used in the IntServ network [15]. Here, Theorem 7 has been proved in [15]. Theorem 8 and Theorem 9 can be easily proved using the same method as for Theorem 2 and Theorem 3.

Theorem 7. **[Rate Guarantee]** *A SCORE network of CSGR servers provides the same rate guarantee (7) as a PFFQ-based IntServ network of the corresponding GR servers.*

Theorem 8. **[Bounded Delay]** *A SCORE network of CSGR servers provides the same bounded delay (8) as a PFFQ-based IntServ network of the corresponding GR servers.*

Theorem 9. **[Throughput Guarantee]** *A SCORE network of CSGR servers provides the same throughput guarantee (9) as a PFFQ-based IntServ network of the corresponding GR servers.*

4 The Link-Based Fair Aggregation Approach

4.1 The Approach

The idea of LBFA was motivated by the following observation. Let us look at Fig. 2, which shows that when a flow passes through each link of its path, it is aggregated with other flows. As shown in Fig. 2, other flows may join and leave the path. In other words, the considered flow is aggregated with other flows of the same traffic class on each link along the path. Observing this, the idea of LBFA comes from the conjecture that since flows traversing the same link have already been aggregated, it may be possible to preserve service guarantees to each individual flow by aggregating those aggregates on each link properly.

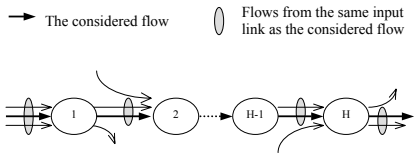

Fig. 2. Aggregation nature of an end-to-end flow

In particular, we treat the aggregation of all flows of the same traffic class, which are from the same input link and destined to the same output link, as a single link-level aggregate. We let this link-level aggregate first pass through a traffic shaper, and then use a fair queueing (FQ) scheduler to multiplex such shaped link-level aggregates from different input links to generate the class-based aggregate fed into the class-based scheduler. We call the resulted rate-controlled scheduler [21], including both the shaper and the normal FQ scheduler, *Fair Queueing with Shaping* (FQ-S) scheduler and call the normal FQ scheduler the corresponding FQ scheduler of the FQ-S scheduler.

More specifically, we require the shaper for any input link i at router s to be a greedy shaper [17] that works like a link with fixed capacity $r_s^{f_i}$. Such a shaper can be implemented by a leaky bucket shaper with leaking rate $r_s^{f_i}$. It is easy to verify that such a shaper has a shaping curve $r_s^{f_i} \cdot t + L$ (e.g. see [12]). Here, f_i denotes the corresponding link-level aggregate from input link i of flow f; $r_s^{f_i}$ denotes the sum of allocated rates of all constituent flows of the link-level aggregate f_i. In addition, we require that both the class-based scheduler and the FQ scheduler belong to the Latency-Rate Worst-case Service Guarantee (LR-WSG) server class [13]. This requirement is (to some extent) necessary since for analyzing hierarchical schedulers, to the best of our knowledge, [13] and [2] are the two main available references. Particularly, the class-based scheduler is an LR-WSG server to the considered traffic class aggregate with rate R_s and error term E_s; the corresponding FQ scheduler provides LR-WSG to each link-level aggregate with rate $r_s^{f_i}$ and error term $e_s^{f_i}$.

The following are some properties of the link server with LBFA implemented, which consists of the shaper, the FQ scheduler and the class-based scheduler. Their proofs can be found from [14].

Lemma 1. *(i) The class-based scheduler is GR server to the class aggregate with rate R_s and error term E_s. (ii) The hierarchical scheduler made of the class-based scheduler and the FQ scheduler in FQ-S is GR server to the link-level aggregate with rate $r_s^{f_i}$ and error term $E_s^{f_i}(= e_s^{f_i} + E_s)$.*

Lemma 2. *The link server implementing LBFA is GR server to flow f_i with rate $r_s^{f_i}$ and error term $E_s^{f_i} + \frac{L}{r_s^{f_i}}$.*

Remarks: With Lemma 2, it can be shown that various service guarantees are provided by router s to the link-level aggregate f_i. In this sense, we say *link level* service guarantees are provided by the LBFA approach at each router.

Lemma 3. *The link server implementing LBFA guarantees to f_i*

$$G_s^{g,k} \leq F^{f_i,j} + \mathcal{E}_s \tag{15}$$

with

$$\mathcal{E}_s = \frac{L}{r_s^{f_i}} + \frac{I_s \cdot L}{R_s} + E_s + E_s^{f_i}. \tag{16}$$

Here in Lemma 3, g denotes the corresponding class-based aggregate of f_i, $p^{g,k} = p^{f_i,j}$ and with $G_s^{g,0} = 0$, $G_s^{g,k}$ is iteratively defined as

$$G_s^{g,k} = max[d_s^{g,k}, G_s^{g,k-1}] + \frac{l^{g,k}}{R_s}. \tag{17}$$

4.2 Per-Flow Service Guarantees

Having introduced the idea of LBFA and some properties of the link server implementing LBFA, we now show through analysis that end-to-end per-flow service guarantees are provided by the network if LBFA is implemented even though each node is unaware of individual flows. We shall see that these guarantees are independent of link utilization level (as long as the link is not overloaded) as in the IntServ and SCORE approaches.

In the following analysis, we assume that in the network, for flow f, the allocated rate R_s to the corresponding class-based aggregate at any router s along its path is not less than the sum of allocated rates to the link-level aggregates forming the class-based aggregate at their pervious hops. It is worth highlighting that this assumption makes sense for real networks. An example is IntServ networks with aggregation of end-to-end reservation [1]. Another example is MPLS networks with LSP merging [8]. In these networks, when bandwidth guarantee is required, it is usually assumed that in cases of reservation aggregation or LSP merging, the reserved rate at downstream routers of the merged point is sufficient to carry the sum of merged traffic [1] [8]. The above assumption could be relaxed based on the idea of enforcing spacing between packets at the edge which has been used in [5].

Lemma 4. *For any packet $p^{f,j}$, the network guarantees that*

$$d_H^{f,j} \leq F^{f_1,k} + \sum_{s=1}^{H} \mathcal{E}_s, \tag{18}$$

where f_1 denotes the corresponding link-level aggregate of flow f at the first router, $\mathcal{E}_s$ is determined by (16), and $p^{f,j} = p^{f_1,k}$.

With Lemma 4, the following can be further derived. (See [14] for proofs.)

Theorem 10. [Rate Guarantee] *For the same network as in Lemma 4, it guarantees to flow f that, with $\sigma^{\overline{f}_1} = \sum_{v \in f_1, \neq f} \sigma^v$,*

$$d_H^{f,j} \leq F^{f,j} + \sum_{s=1}^{H} \mathcal{E}_s + \frac{\sigma^{\overline{f}_1}}{r^{f_1}}. \tag{19}$$

Theorem 11. [Bounded Delay] *For the same network as in Lemma 4, the end-to-end delay of any packet in flow f is bounded by:*

$$d_H^{f,j} \leq \frac{\sigma^{f_1}}{r^{f_1}} + \sum_{s=1}^{H} \mathcal{E}_s. \tag{20}$$

Theorem 12. [Throughput Guarantee] *For the same network as in Lemma 4, if flow f transmits packets at least at rate r^f, then the network guarantees to the flow that, with $\sigma^{\overline{f}} = \sum_{v \in f_1, \neq f} \sigma^v$,*

$$W^f(t_1, t_2) \geq r^f \left(t_2 - t_1 - \sum_{s=1}^{H} \mathcal{E}_s - \frac{\sigma^{\overline{f}}}{r^{f_1}} - \frac{L}{r^f} \right)^+. \tag{21}$$

Remarks: By expending $\mathcal{E}_s$ in (19), we get $d_H^{f,j} \leq F^{f,j} + \sum_{s=1}^{H} \frac{L}{r_s^{f_i}} + \sum_{s=1}^{H} (E_s + E_s^{f_i}) + \frac{\sigma^{\overline{f}_1}}{r^{f_1}} + \sum_{s=1}^{H} \frac{I_s \cdot L}{R_s}$, comparing which with (7), it is not difficult to verify that in general (19) is comparable with (7). In addition, if the edge router maintains per-flow states as in SCORE and DiffServ, a per-flow FQ scheduler can be used to aggregate constituent flows of f_1 to form the link-level aggregate. As a result, the forth term can be removed. Consequently, the resulted network is only *core-stateless* while the guarantee can be better than (7).

A similar approach as LBFA can be found from [6], in which flows are aggregated at path level using fair aggregator. (The author would like to thank the anonymous reviewers for pointing out this.) In [6], a fair aggregator is implemented by limiting the total output rate of a fair queueing scheduler. Hence, there are two major differences between LBFA and the approach in [6]. One is that the levels of service guarantees provided by them at each router are different; the other is their suggested implementations of flow aggregators are different. Nevertheless, one might view LBFA as an extension of the approach in [6]. In fact, LBFA may be applied to aggregating flows at path level and the fair aggregator suggested in [6] may be used to perform link-level flow aggregation.

5 Comparison

Table 1. Comparison of approaches to service guarantees

Approach	PFFQ-based IntServ	FIFO-based DiffServ	SCORE	LBFA
Granualarity	flow level	class level	packet level	link level
Guarantees	Th. 1 - 3	Th. 4 - 6	Th. 7 - 9	Th. 10 - 12
Limitation	no	yes	yes[3]	no
Flow states	stateful	core-stateless	core-stateless	(core-)stateless
# of states	$O(\mathcal{N})$	no	no	no
Complexity	$O(log(\mathcal{N}))^1/O(1)^2$	straightforward	$O(log(\mathcal{M}))$	$O(\mathcal{I})^1$ / $O(1)^2$
Packet state	no	no	yes	no

[1]: use deadline-based FQ. [2]: use RR-based FQ. [3]: "No" only if PS is stored finestly.
$\mathcal{N}$: # of active flows $\mathcal{I}$: # of input links $\mathcal{M}$: # of packets in queue

Table 1 presents a brief comparison of the existing and proposed approaches. More detailed discussion can be found from [14]. From the table, it can be seen that PFFQ-based IntServ provides powerful service guarantees but has serious scalability problem. FIFO-based DiffServ is scalable but cannot provide comparable per-flow service guarantees as IntServ and has a limitation on link utilization level. While SCORE is both scalable and has the same ability as IntServ in providing per-flow service guarantees, it introduces additional implementation requirements that could limit its use in the current Internet infrastructure. The proposed LBFA approach, which is simple to implement, provides another option for providing per-flow service guarantees that are comparable to those provided by PFFQ-based IntServ. LBFA can be considered as a compromise between the PFFQ-based IntServ approach and the FIFO-based DiffServ approach. In fact, if we simply treat LBFA as a flow aggregation method, it could be used under both the IntServ architecture and the DiffServ architecture. In addition, LBFA may be used with IntServ to solve its scalability problem or with DiffServ to avoid its link utilization problem.

6 Conclusion

In this paper, we discussed from the service guarantee granularity perspective the three widely studied approaches for providing service guarantees, which are IntServ, DiffServ and SCORE. We also summarized per-flow service guarantees provided by them. In addition, we proposed a new approach, called LBFA approach to scalable support of per-flow service guarantees. While the granularity of service guarantees provided by LBFA at each route is *link level*, we showed that the per-flow service guarantees provided by LBFA are comparable

to those by PFFQ-based IntServ and SCORE, and are better than those provided by FIFO-based DiffServ. Moreover, we compared briefly the three current approaches and the LBFA approach. The comparison showed that the LBFA approach is as scalable as the FIFO-based DiffServ and SCORE approaches and provides comparable per-flow service guarantees as the PFFQ-based IntServ and SCORE approaches. We believe that LBFA is a simple yet effective approach and it could be used with IntServ or DiffServ to solve their specific problems.

References

1. F. Baker et al. Aggregation of RSVP for IPv4 and IPv6 reservation. *IETF RFC 3175*, Sept. 2001.
2. J. C. R. Bennett and H. Zhang. Hierarchical packet fair queueing algorithms. *IEEE/ACM Trans. Networking*, 5(5):675 – 689, Oct 1997.
3. S. Blake et al. An architecture for Differentiated Services. *IETF RFC 2475*, 1998.
4. R. Braden, D. Clark, and S. Shenker. Integrated services in the Internet architecture: An overview. *IETF RFC1633*, 1994.
5. D. Chlamtac et al. A deterministic approach to the end-to-end analysis of packet flows in connection-oriented networks. *IEEE/ACM ToN*, 6(4):422–431, Aug. 1998.
6. J. A. Cobb. Preserving quality of service guarantees in spite of flow aggregation. *IEEE/ACM Trans. Networking*, 10(1):43–53, Feb. 2002.
7. B. Davie et al. An Expedited Forwarding PHB. *IETF RFC 3246*, March 2002.
8. F. L. Faucheur and et al. Multiprotocol label switching (MPLS) support of Differentiated Services. *IETF RFC 3270*, May 2002.
9. S. Floyd and V. Jacobson. Link-sharing and resource management models for packet networks. *IEEE/ACM Trans. Networking*, 3(4), 1995.
10. P. Goyal, S. S. Lam, and H. M. Vin. Determining end-to-end delay bounds in heterogeneous networks. *Springer Multimedia Systems*, 5:157–163, 1997.
11. J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured forwarding PHB group. *IETF RFC 2597*, June 1999.
12. Y. Jiang. Delay bounds for a network of Guaranteed Rate servers with FIFO aggregation. *Computer Networks*, 40(6):683–694, Dec. 2002.
13. Y. Jiang. Per-domain packet scale rate guarantee for Expedited Forwarding. Proc. IWQoS 2003, LNCS 2707, pp. 422–439, 2003
14. Y. Jiang. Link-Based Fair Aggregation: A Simple Approach to Scalable Support of Per-Flow Service Guarantees. Technical Report, Q2S, NTNU, 2004.
15. J. Kaur and H. M. Vin. Core-stateless guaranteed rate scheduling algorithms. In *Proc. INFOCOM'01*, 2001.
16. J. Kaur and H. M. Vin. Core-stateless guaranteed throughput networks. In *Proc. INFOCOM'03*, 2003.
17. J.-Y. Le Boudec and P. Thiran. *Network Calculus: A Theory of Deterministic Queueing Systems for the Internet*. Springer-Verlag, 2001.
18. K. Nichols and B. Carpenter. Definition of differentiated services per domain behaviors and rules for their specification. *IETF RFC 3086*, April 2001.
19. S. Shenker, C. Partridge, and R. Guerin. Specification of guaranteed quality of service. *IETF RFC 2212*, Sept 1997.
20. I. Stoica and H. Zhang. Providing guaranteed services without per flow management. In *Proc. SIGCOMM'99*, 1999.
21. H. Zhang and D. Ferrari. Rate-controlled service disciplines. *J. High Speed Networks*, 3(4), 1994.