

End-to-end Fairness Analysis of Fair Queuing Networks*

Jasleen Kaur and Harrick M. Vin

Laboratory for Advanced Systems Research
Department of Computer Sciences
University of Texas at Austin
E-mail: {jks, vin}@cs.utexas.edu; Phone: (512) 232-7887
URL: <http://www.cs.utexas.edu/users/lasr>

Abstract

In this paper, we present the *first* end-to-end fairness analysis of a network of fair servers. We argue that it is difficult to extend existing single-node fairness analysis to an end-to-end analysis of a network where each node may employ a different fair scheduling algorithm. We then present a two-step approach for end-to-end fairness analysis of heterogeneous networks. First, we define a class of scheduling algorithms, referred to as the *Fair Throughput (FT)* class, and prove that most known fair scheduling algorithms belong to this class. Second, we develop an analysis methodology for deriving the end-to-end fairness bounds for a network of FT servers. Our analysis is general and can be applied to heterogeneous networks where different nodes employ different scheduling algorithms from the FT class.

1 Introduction

With the commercialization of the Internet and the advent of real-time and mission-critical Internet applications, it has become important for network service providers to export rich service semantics to users. Over the past decade, several packet scheduling algorithms that allow networks to offer such rich service semantics have been proposed [4, 5, 9, 10, 14, 19, 23, 24]. These scheduling algorithms arbitrate access to the shared link bandwidth available at routers among flows.

Why Fair Scheduling? Fairness of bandwidth allocation among competing flows is an important property of packet scheduling algorithms. A fair scheduling algorithm allows routers (1) to provide throughput guarantees to backlogged flows at short time-scales, independent of past usage of the link bandwidth by the flows; and (2) to allocate idle link capacity to competing flows in proportion to their weights (or reserved rates).

The property of providing throughput guarantees at short time-scales independent of the past bandwidth usage by the

flow is important for two reasons.

1. In many applications, sources may not be able to predict precisely their bandwidth requirements at short time-scales (consider, for instance, the problem of transmitting variable bit-rate encoded live video stream). To support these applications effectively, a network should allow flows to utilize occasionally more than their reserved bandwidth if such over-usage does not come at the expense of violating the bandwidth guarantees provided to other flows. Further, when a network allows a flow to utilize such idle bandwidth, it should not penalize the flow in the future. In networks that penalize sources for using idle bandwidth, applications may prefer to use constant bit-rate flows (with possibly fluctuating quality), instead of allowing the network to enforce arbitrary penalties. Hence, to support applications with unpredictable fluctuations in bandwidth requirements, it is essential that networks utilize fair scheduling algorithms.
2. It is in the best interest of a network to allow sources to transmit data in bursts; bursty transmissions allow a network to benefit from statistical multiplexing of the available network bandwidth among competing traffic. Once again, if a network were to penalize a flow for using idle bandwidth, then the source would have no incentive to transmit bursts into the network; this, in turn, would reduce the statistical multiplexing gains and thereby reduce the overall utilization of network resources. Thus, fair scheduling algorithms allow networks to achieve higher gains due to statistical multiplexing.

The property of fair scheduling algorithms of allocating available bandwidth to flows in proportion to their reserved rates is desirable from an economic perspective. Consider, for instance, the case when a network provider charges its customers based on their reserved bandwidth. In such a network, if a user A pays twice as much as user B , then A expects the network to allocate bandwidth in the ratio 2:1 to users A and B ; any other allocation would be considered unfair. Fair scheduling algorithms allow a network to ensure this

*This research was supported in part by grants from NSF (award ANI-0082294), Intel, IBM, and Cisco.

proportionate allocation property independent of the amount of available bandwidth.

Why End-to-end Fairness Guarantees? Over the last decade, many fair scheduling algorithms have been proposed [4, 5, 10, 14]; the literature contains analyses that derive single-server fairness guarantees for these fair scheduling algorithms. From an end-user perspective, though, the notion of *end-to-end fairness* is more meaningful. Additionally, from a network provider’s perspective, quantification of end-to-end fairness guarantees is necessary for offering service level agreements (SLAs) to customers.

Deriving end-to-end fairness guarantees of fair queuing networks is also important for evaluating the effectiveness of *core-stateless* networks in providing fairness guarantees. Over the past few years, core-stateless networks have been designed to provide end-to-end service guarantees without maintaining or using any per-flow state at the core routers of a network [21]; this property improves the scalability of the core routers to large number of flows and high-speed links. Existing proposals for providing fairness in core-stateless networks only provide *approximate* fairness in the end-to-end throughput achieved by flows over large time-scales [6, 8, 18, 21]. To design and evaluate core-stateless networks that provide deterministic end-to-end fairness guarantees at short time-scales, it is important to first understand and determine the end-to-end fairness guarantees that can be provided by core-stateful networks that employ fair scheduling algorithms.

Research Contributions In this paper, we present an end-to-end fairness analysis of a network of routers, each employing a fair scheduling algorithm. To the best of our knowledge, this is the *first* such analysis in the literature. We argue that it is difficult to extend existing single-node fairness analysis to an end-to-end analysis of a network where each node may employ a different fair scheduling algorithm. We then present a two-step approach for end-to-end fairness analysis of heterogeneous networks. First, we define a class of scheduling algorithms, referred to as the *Fair Throughput (FT)* class, and show that most known fair scheduling algorithms belong to this class. Second, we develop an analysis methodology for deriving the end-to-end fairness bounds for a network of FT servers. Our analysis is general and can be applied to heterogeneous networks where different nodes employ different scheduling algorithms from the FT class.

The rest of this paper is organized as follows. In Section 2, we formulate the problem of end-to-end fairness. In Section 3, we discuss the challenges involved in extending existing single-node fairness analyses to an end-to-end analysis. In Section 4, we define the class of FT algorithms. We present an end-to-end fairness analysis for a network of FT servers in Section 5. We discuss related work in Section 6 and summarize our conclusions in Section 7.

2 Problem Formulation

Consider a flow f that traverses a network path of length H . Let r_f be the rate reserved for the flow on all nodes. The *throughput* received by flow f at server j during a time interval $[t_1, t_2]$, denoted by $W_{f,j}(t_1, t_2)$, is defined as the number of bits of flow f that depart server j during the time interval $[t_1, t_2]$. Also, a flow f is said to be *continuously backlogged* at server j in a time interval $[t_1, t_2]$ if, at all instances within this interval, there is at least one packet belonging to flow f in the server queue. Throughout this paper, we use the terms node, server, and hop interchangeably.

In any time interval during which flows are backlogged¹, an ideal fair server provides throughput to flows *exactly* in proportion to their reserved rates. This idealized notion of fairness, however, is infeasible to realize in a packetized system. Fair algorithms for packet scheduling, instead, guarantee an upper bound on the difference in the normalized throughput (weighted by the reserved rate) received by flows at a server in intervals during which they are continuously backlogged [10]. This notion of a *fairness guarantee*² is formally defined as follows:

Definition 1 *The scheduling algorithm at node j is said to provide a fairness guarantee if in any time interval $[t_1, t_2]$ during which two flows f and m are continuously backlogged, the number of bits of flows f and m transmitted by the server, $W_{f,j}(t_1, t_2)$ and $W_{m,j}(t_1, t_2)$ respectively, satisfy:*

$$\left| \frac{W_{f,j}(t_1, t_2)}{r_f} - \frac{W_{m,j}(t_1, t_2)}{r_m} \right| \leq U_{j,\{f,m\}} \quad (1)$$

where r_f and r_m are the rates reserved for flows f and m respectively, and $U_{j,\{f,m\}}$ is the *unfairness measure*—a constant that depends on the scheduling algorithm and traffic characteristics at server j .

Different fair scheduling algorithms [4, 5, 10, 14, 19] differ in the value of $U_{j,\{f,m\}}$, the unfairness measure. Table 1 lists the $U_{j,\{f,m\}}$ values for several known fair scheduling algorithms.

In Definition 2, we generalize the above definition to that of end-to-end fairness guarantee. Observe that the fairness property is meaningful only across flows that share a resource; thus, the notion of end-to-end fairness is defined only across flows that share the same end-to-end network path.

¹Fairness in throughput allocation is usually defined only with respect to flows that are backlogged. This is because the throughput of a non-backlogged flow may be constrained by the source traffic rather than by the allocation of link capacity.

²The literature also contains a different notion of fairness—namely, worst-case fairness—in which a flow is guaranteed a minimum throughput at its reserved rate, irrespective of traffic arrival in other flows [4]. This kind of guarantee has also been referred to as a throughput guarantee [11]. It can be shown that a server that provides a fairness guarantee as given by Definition 1, when used in conjunction with admission control, also provides a throughput guarantee. In this paper, we focus on the (stronger) notion of fairness provided by Definition 1.

Definition 2 A network is said to provide an end-to-end fairness guarantee if in any time interval $[t_1, t_2]$ during which two flows, f and m , that traverse the same network path of length H hops, are continuously backlogged at the **first** server, the number of bits of flows f and m that depart the network, $W_{f,H}(t_1, t_2)$ and $W_{m,H}(t_1, t_2)$ respectively, satisfy:

$$\left| \frac{W_{f,H}(t_1, t_2)}{r_f} - \frac{W_{m,H}(t_1, t_2)}{r_m} \right| \leq U_{H,\{f,m\}}^{net} \quad (2)$$

where $U_{H,\{f,m\}}^{net}$ is a constant that depends on the server and traffic characteristics at the different hops in the end-to-end network path.

In this paper, our objective is to compute an upper bound on the network unfairness measure, $U_{H,\{f,m\}}^{net}$.

It may seem tempting to reason that, since the throughput of a flow is determined by the rate allocated at a “bottleneck” server along its path, the end-to-end fairness guarantee for two flows would be the same as the fairness guarantee provided by the bottleneck server. Unfortunately, such a reasoning is incorrect. This is because, end-to-end throughput received by a flow at short time-scales depends on the queuing delay suffered by its packets. At short time-scales, individual packets of flows may encounter queuing delay at *several* servers, even when at large time-scales, a single bottleneck server may determine the average bandwidth allocated to a flow. Therefore, the end-to-end fairness guarantee is not the same as the fairness guarantee provided by any single server.

Note that we have assumed that r_f , the rate reserved for flow f , is the same at all routers on its path. There are many ways to compute r_f . For instance, r_f could be the minimum rate acceptable to the end-user, or it could be the rate determined according to the *max-min* fair rate allocation scheme—such a scheme allocates to a flow a rate at each router that is no more than the fair rate allocated to that flow at its bottleneck router. The problem of determining the rate r_f is orthogonal to the problem of end-to-end fairness analysis, and is outside the scope of this paper.

3 End-to-end Fairness Analysis: Challenges

As mentioned earlier, designers of individual fair scheduling algorithms generally prove fairness bounds ($U_{j,\{f,m\}}$) with respect to throughput achieved by different flows only at a *single* server [4, 14]; the literature, however, does not contain analyses that prove fairness bounds ($U_{H,\{f,m\}}^{net}$) for the *end-to-end* throughput achieved by flows in a network of fair servers. There are two inherent difficulties in extending existing single-server analyses to end-to-end network analysis.

1. The literature contains single-server fairness analyses only for specific scheduling algorithms [4, 5, 14]. In a wide area network, however, each router may employ a different scheduling algorithm. Hence, an end-to-end

fairness analysis should be applicable to such heterogeneous networks.

2. Most single-server analyses of fair scheduling algorithms presented in the literature derive fairness bounds only over intervals during which the concerned flows are simultaneously and continuously backlogged. Due to the variability in the delay experienced by packets at a server, traffic gets distorted as it traverses through the network. Therefore, flows may not be simultaneously or continuously backlogged at all the servers along the path and during all time intervals, even if they are at the first server. Hence, it is not straightforward to apply existing single-server analyses to determine bounds on end-to-end network fairness.

We address these limitations in two steps. First, we define a general class of *Fair Throughput (FT)* servers; we show that most of the known fair scheduling algorithms belong to this class (Section 4). Second, we develop an analysis methodology for deriving end-to-end fairness bounds for a network of FT servers, where different nodes may employ different scheduling algorithms from the FT class (Section 5).

4 The Class of Fair Throughput Servers

Recall that the fairness guarantee in Definition 1 is applicable only to time intervals during which both flows are continuously backlogged. As discussed in Section 3, there may be time intervals during which one or both of the flows may not be continuously backlogged at subsequent servers. To facilitate fairness analysis during such time intervals at subsequent servers, we define the class of *Fair Throughput (FT)* scheduling algorithms. An algorithm in the FT class provides a stronger notion of the per-node fairness guarantee—if a flow m is continuously backlogged during an interval, then the normalized throughput received by *any* other flow f (whether continuously backlogged or not during the interval) does not exceed the normalized throughput received by flow m by more than a bounded quantity. We formalize this notion below.

Definition 3 The scheduling algorithm at node j belongs to the class of *Fair Throughput servers* if in any time interval $[t_1, t_2]$ during which a flow m is continuously backlogged, the number of bits transmitted by the server for any flow f and flow m , $W_{f,j}(t_1, t_2)$ and $W_{m,j}(t_1, t_2)$ respectively, satisfy:

$$\frac{W_{f,j}(t_1, t_2)}{r_f} \leq \frac{W_{m,j}(t_1, t_2)}{r_m} + I_{j,m,f}$$

where $I_{j,m,f}$ is a constant that depends on the server and traffic characteristics at node j .

From Definition 1 and Definition 3, it is easy to observe that all FT algorithms also provide fairness guarantees with $U_{j,\{f,m\}} = \max(I_{j,m,f}, I_{j,f,m})$.

The definition of the class of FT scheduling algorithms is fairly general, and most well-known fair scheduling algorithms—such as Generalized Processor Sharing (GPS) [19], Self-clocked Fair Queuing (SCFQ) [10], Start-time Fair Queuing (SFQ) [14], Worst-case Fair Weighted Fair Queuing (WF²Q) [4]—belong to this class. We derive the $I_{j,m,f}$ values for these algorithms in Appendix A, and summarize them in Table 1.

We expect that any fair scheduling algorithm that provides per-node fairness guarantee (Definition 1) can be shown to belong to the FT class. This is because, during sub-intervals in which f is also backlogged, the difference in normalized throughput received by flows f and m is bounded (due to the fairness guarantee provided by the fair scheduling algorithm). On the other hand, during sub-intervals in which f is *not* backlogged, its throughput is zero, which can not exceed the throughput received by flow m . A formal proof for this assertion, however, is beyond the scope of this paper.

5 End-to-end Analysis of FT Networks

Given a network of FT servers, our objective is to derive an upper-bound on $\left| \frac{W_{f,H}(t_1,t_2)}{r_f} - \frac{W_{m,H}(t_1,t_2)}{r_m} \right|$, the difference in normalized throughput received during any time interval $[t_1, t_2]$ by flows f and m that traverse the same end-to-end path of H servers, assuming only that the flows are continuously-backlogged at the first server. In the following, we first present the methodology for conducting the end-to-end fairness analysis (Section 5.1), and then present the formal lemmas, theorem, and their proofs (Section 5.2).

5.1 Analysis Methodology

As mentioned in Section 3, one of the main challenges in extending single-server fairness analysis to an end-to-end analysis is that flows may not remain continuously or even simultaneously backlogged at subsequent servers. The question we would like to answer is: *given that the first server provides a fairness guarantee to the two backlogged flows, can we say something similar about the throughput received at the second, third, fourth, and so on, servers?* If we can relate the fairness guarantee provided at a server to the fairness guarantee provided at the *previous* server, then by using a recursive argument, we would be able to answer the above question in the affirmative. In particular, we need to answer the transformed question: *given $U_{j,\{f,m\}}^{net}$, the bound on difference in normalized throughput achieved at j^{th} server by flows f and m , can we compute $U_{j+1,\{f,m\}}^{net}$, the bound on difference in their normalized throughput at the $(j+1)^{\text{th}}$ server?*

For simplicity of exposition, while discussing the methodology in this section, we assume zero propagation latencies on the links connecting servers. For any time interval $[t_1, t_2]$ at the $(j+1)^{\text{th}}$ server, we consider the following two cases:

- If *none* of the flows are backlogged at the *time instants*

t_1 and t_2 , then the throughput received by the two flows in the interval $[t_1, t_2]$ is the *same* as the throughput they receive at the previous server in this time interval (assuming zero propagation latencies). This is because, no packets that were received before t_1 get served in $[t_1, t_2]$ (since there is no backlog at t_1), and no packets received during $[t_1, t_2]$ get served after t_2 (no backlog at t_2 either).

Therefore, for such time intervals, the difference in normalized throughput of the two flows at server $j+1$ has the same upper bound as that for server j .

- If either one of the flows is backlogged at server $j+1$ at t_1 or t_2 , then the number of packets of that flow served during $[t_1, t_2]$ at server $j+1$ may be different from those served at server j . This is because, some packets that are backlogged at t_1 may get served in $[t_1, t_2]$, and some packets that arrive from server j in $[t_1, t_2]$ may not get served and may remain backlogged at t_2 . The throughput of the flow during $[t_1, t_2]$ at server $j+1$ is therefore determined not only by its throughput at server j , but also the backlogs at server $j+1$ at times t_1 and t_2 . It follows that to derive the fairness guarantee of server $j+1$ during such time intervals, we additionally need to bound the difference in the normalized backlogs of the two flows, at both t_1 and t_2 .

Let us consider a time instant t_0 , smaller than t_1 , at which none of the flows are backlogged at server $j+1$. The backlog of either flow at t_1 (or t_2) can be computed as the number of packets that arrive in $[t_0, t_1]$ (or $[t_0, t_2]$) minus the number of packets that are served in the same time interval. From the fairness guarantee of server j , we know that the difference in (normalized) number of packets that arrive at server $j+1$ in $[t_0, t_1]$ (or $[t_0, t_2]$) for the two flows is bounded. It follows that to bound the difference in normalized backlogs at t_1 (or t_2) at server $j+1$, we need to bound the difference in normalized throughput during $[t_0, t_1]$ (or $[t_0, t_2]$).

To compute the difference in normalized throughput of flows f and m at server $j+1$ during $[t_0, t_1]$ (or $[t_0, t_2]$), we consider the following two scenarios:

- *Both flows are backlogged at t_1 (or t_2).*

Lemma 1 establishes a lower bound on the normalized throughput during $[t_0, t_1]$ of either flow (say, f) in terms of the normalized throughput of the other (flow m). The proof methodology for this lemma is based on the following two observations. Let $t' \in [t_0, t_1]$ be the *last* time instant before which f is non-backlogged (see Figure 1).

1. The throughput of flow f in $[t_0, t']$ at server $j+1$ is the same as its throughput in the same time interval at server j (since no backlogs

| FT algorithm | $U_{j,\{f,m\}}$ | $I_{j,m,f}$ |
|-------------------|--|--|
| GPS | 0 | 0 |
| SCFQ | $\frac{l_f^{max}}{r_f} + \frac{l_m^{max}}{r_m}$ | $\frac{l_f^{max}}{r_f} + \frac{l_m^{max}}{r_m}$ |
| SFQ | $\frac{l_f^{max}}{r_f} + \frac{l_m^{max}}{r_m}$ | $\frac{l_f^{max}}{r_f} + \frac{l_m^{max}}{r_m}$ |
| WF ² Q | $l^{max} \left(\frac{1}{r_f} + \frac{1}{r_m} - \frac{1}{C} \right)$ | $\frac{l^{max}}{r_m} + l_f^{max} \left(\frac{1}{r_f} - \frac{1}{C} \right)$ |

Table 1: Unfairness measures for some FT algorithms

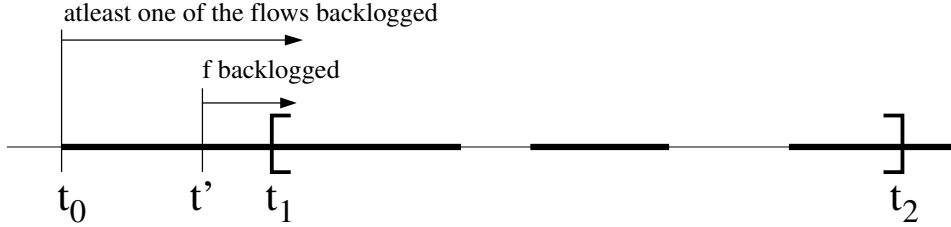


Figure 1: Reference for Lemma 1

at either t_0 or t'). At server j , the throughput of flow f is lower bounded in terms of the throughput of flow m (due to the fairness guarantee of server j). Further, the throughput of flow m at server $j + 1$ during $[t_0, t']$ (no backlog at t_0) cannot exceed the corresponding throughput at server j .

2. Flow f is continuously backlogged in $[t', t_1]$ and Definition 3 can be applied to this interval to establish a lower bound on throughput of flow f in terms of throughput of flow m .
- Only one of the flows is backlogged at t_1 (or t_2).

As described above, Lemma 1 can derive a lower bound on the normalized throughput during $[t_0, t_1]$ of the backlogged flow (say, f) in terms of the throughput of the other (flow m).

To compute the reverse relation, that is, a lower bound on throughput of flow m , first observe that during $[t_0, t_1]$, the throughput of flow m at server $j + 1$ is the same as its throughput at server j (since no backlogs at either t_0 or t_1). At server j , the throughput of flow m is lower bounded in terms of the throughput of flow f (due to the fairness guarantee of server j). Further, the throughput of flow f at server $j + 1$ during $[t_0, t_1]$ (no backlog at t_0) cannot exceed its throughput at server j . These observations are used in Lemma 2 to compute a lower bound on the throughput of flow m in terms of that of flow f .

Therefore, by using Lemma 1 and Lemma 2, we can compute upper bounds on the difference in normalized throughput of the two flows during both $[t_0, t_1]$ and $[t_0, t_2]$. The sum of these two bounds then gives a candi-

date value of $U_{j+1,\{f,m\}}^{net}$, the upper bound on the difference in normalized throughput during $[t_1, t_2]$. Lemma 3 helps tighten this value, based on the fairness guarantee of the FT algorithm at server j .

5.2 Formal Analysis and Proofs

Using the methodology described above, we derive the end-to-end fairness guarantee of a network of FT servers in Theorem 1. For the remainder of the analysis, we remove the simplistic assumption (made in Section 5.1) of non-zero link propagation latencies. We use $\pi_j(t)$ to denote the propagation latency experienced—on the link connecting server j and $j + 1$ —by the last packet of either flow received at server $j + 1$ by time t .

Lemma 1 *If flow f is backlogged at server $j + 1$ at time t and if t_0 is a time instant smaller than t such that neither f nor m is backlogged at t_0^- , and at least one is backlogged at t_0 , then:*

$$\frac{W_{f,j+1}(t_0, t)}{r_f} \geq \frac{W_{m,j+1}(t_0, t)}{r_m} + a' - I_{j+1,f,m}$$

where $a' = W_{f,j}(t_0 - \pi_j(t_0), t' - \pi_j(t'))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t' - \pi_j(t'))/r_m$, and $t' \in [t_0, t]$ is the latest time instant at which f becomes backlogged.

Proof: Since packets arrive at server $j + 1$ in the same order they are transmitted at server j , it follows that the packets received at server $j + 1$ during $[t_0, t']$ are the packets transmitted from server j during $[t_0 - \pi_j(t_0), t' - \pi_j(t')]$.

Since f is not backlogged at t'^- and t_0^- , we have: $W_{f,j+1}(t_0, t') = W_{f,j}(t_0 - \pi_j(t_0), t' - \pi_j(t'))$. Further, since flow m is not backlogged at t_0^- , we have: $W_{m,j+1}(t_0, t') \leq W_{m,j}(t_0 - \pi_j(t_0), t' - \pi_j(t'))$.

Since $a' = W_{f,j}(t_0 - \pi_j(t_0), t' - \pi_j(t'))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t' - \pi_j(t'))/r_m$. Then, we have:

$$\frac{W_{f,j+1}(t_0, t')}{r_f} \geq \frac{W_{m,j+1}(t_0, t')}{r_m} + a' \quad (3)$$

Since flow f is continuously backlogged during $[t', t]$, and server $j + 1$ belongs to the FT class, we have from Definition 3:

$$\frac{W_{m,j+1}(t', t)}{r_m} \leq \frac{W_{f,j+1}(t', t)}{r_f} + I_{j+1,f,m} \quad (4)$$

From (3) and (4), we get:

$$\frac{W_{f,j+1}(t_0, t)}{r_f} \geq \frac{W_{m,j+1}(t_0, t)}{r_m} + a' - I_{j+1,f,m}$$

■

Lemma 2 *If flow m is not backlogged at server $j+1$ at time t and if t_0 is a time instant smaller than t such that neither f nor m is backlogged at t_0^- , and at least one is backlogged at t_0 , then:*

$$\frac{W_{m,j+1}(t_0, t)}{r_m} \geq \frac{W_{f,j+1}(t_0, t)}{r_f} - a$$

where $a = W_{f,j}(t_0 - \pi_j(t_0), t - \pi_j(t))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t - \pi_j(t))/r_m$.

Proof: Since packets arrive at server $j + 1$ in the same order they are transmitted at server j , it follows that the packets received at server $j + 1$ during $[t_0, t]$ are the packets transmitted from server j during $[t_0 - \pi_0, t - \pi]$.

Since m is not backlogged at t and t_0^- , we have: $W_{m,j+1}(t_0, t) = W_{m,j}(t_0 - \pi_0, t - \pi)$. Further, since f is not backlogged at t_0^- , we have: $W_{f,j+1}(t_0, t) \leq W_{f,j}(t_0 - \pi_0, t - \pi)$.

Since $a = W_{f,j}(t_0 - \pi_0, t - \pi)/r_f - W_{m,j}(t_0 - \pi_0, t - \pi)/r_m$, we have:

$$\frac{W_{m,j+1}(t_0, t)}{r_f} \geq \frac{W_{f,j+1}(t_0, t)}{r_m} - a$$

■

Lemma 3 *If during a time interval $[t_0, t_1]$, the numbers of bits transmitted by a server for flow f and m satisfy (1) with unfairness measure H' , and if $a_1 = W_{f,j}(t_0, t_1)/r_f - W_{m,j}(t_0, t_1)/r_m$, and $a_2 = W_{f,j}(t_0, t_2)/r_f - W_{m,j}(t_0, t_2)/r_m$, where $t_0 \leq t_1 \leq t_2 \leq t$, then*

$$-H' \leq a_2 - a_1 \leq H'$$

Proof:

$$\begin{aligned} \frac{W_{f,j}(t_1, t_2)}{r_f} &= \frac{W_{f,j}(t_0, t_2)}{r_f} - \frac{W_{f,j}(t_0, t_1)}{r_f} \\ &= \frac{W_{m,j}(t_0, t_2)}{r_m} + a_1 - \frac{W_{m,j}(t_0, t_1)}{r_m} - a_2 \\ &= \frac{W_{m,j}(t_1, t_2)}{r_m} + a_1 - a_2 \end{aligned}$$

Since the number of bits transmitted during the interval $[t_1, t_2]$ should satisfy (1) with unfairness measure H' , it follows that: $-H' \leq a_2 - a_1 \leq H'$. ■

Theorem 1 *If the throughput obtained by two flows, f and m , at the first node in a network of FT servers satisfies (1), and if they share the same end-to-end path of H hops, then during any time interval $[t_1, t_2]$, the end-to-end throughput of the flows are related as:*

$$\begin{aligned} &\left| \frac{W_{f,H}(t_1, t_2)}{r_f} - \frac{W_{m,H}(t_1, t_2)}{r_m} \right| \\ &\leq U_{1,\{f,m\}} + \sum_{h=2}^H (I_{h,f,m} + I_{h,m,f}) \quad (5) \end{aligned}$$

Proof: The proof is by induction on j , the number of hops.

Base Case: $j = 1$. By assumption, the throughput of the two flows at the first node satisfies (1), which is the same as (5) for the base case ($H = 1$).

Induction Hypothesis: Assume (5) is true for all servers $1, \dots, j$.

Induction Step: We need to show that (5) holds at server $j + 1$. Without loss of generality, assume that $\frac{W_{f,j+1}(t_1, t_2)}{r_f} \geq \frac{W_{m,j+1}(t_1, t_2)}{r_m}$. It follows that to prove (5), we only need to show that:

$$\begin{aligned} \frac{W_{f,j+1}(t_1, t_2)}{r_f} &\leq \frac{W_{m,j+1}(t_1, t_2)}{r_m} + U_{1,\{f,m\}} \\ &\quad + \sum_{h=2}^{j+1} (I_{h,f,m} + I_{h,m,f}) \quad (6) \end{aligned}$$

Let $t_0 \leq t_1$ be the *largest* time instant such that none of the two flows are backlogged at server $j + 1$ at t_0^- (see Figure 2).

Difference in normalized throughput during $[t_0, t_1]$:

Consider the following two cases at t_1 :

- If flow f is backlogged at t_1 (see Figure 2), then from Lemma 1, there exists $t_3 \in [t_0, t_1]$, such that:

$$\frac{W_{f,j+1}(t_0, t_1)}{r_f} \geq \frac{W_{m,j+1}(t_0, t_1)}{r_m} + a_3 - I_{j+1,f,m}$$

where $a_3 = W_{f,j}(t_0 - \pi_j(t_0), t_3 - \pi_j(t_3))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t_3 - \pi_j(t_3))/r_m$.

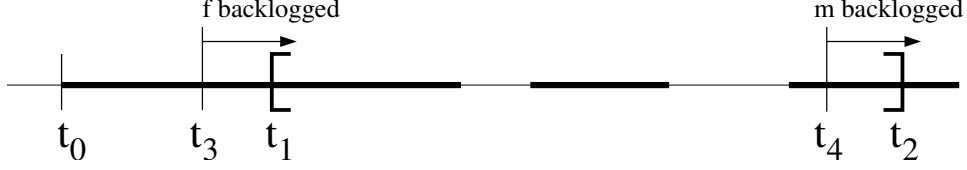


Figure 2: Reference for proof of Theorem 1

- If flow f is *not* backlogged at t_1 , then from Lemma 2, we have:

$$\begin{aligned} \frac{W_{f,j+1}(t_0, t_1)}{r_f} &\geq \frac{W_{m,j+1}(t_0, t_1)}{r_m} + a_1 \\ &\geq \frac{W_{m,j+1}(t_0, t_1)}{r_m} + a_1 - I_{j+1,f,m} \end{aligned}$$

$$\text{where } a_1 = W_{f,j}(t_0 - \pi_j(t_0), t_1 - \pi_j(t_1))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t_1 - \pi_j(t_1))/r_m.$$

Therefore, in either case, there exists some $t' \in [t_0, t_1]$, such that:

$$\frac{W_{f,j+1}(t_0, t_1)}{r_f} \geq \frac{W_{m,j+1}(t_0, t_1)}{r_m} + a' - I_{j+1,f,m} \quad (7)$$

$$\text{where } a' = W_{f,j}(t_0 - \pi_j(t_0), t' - \pi_j(t'))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t' - \pi_j(t'))/r_m.$$

Difference in normalized throughput during $[t_0, t_2]$:

Consider the following two cases at t_2 :

- If flow m is backlogged at t_2 (see Figure 2), then from Lemma 1, there exists $t_4 \in [t_0, t_2]$, such that:

$$\frac{W_{m,j+1}(t_0, t_2)}{r_m} \geq \frac{W_{f,j+1}(t_0, t_2)}{r_f} - a_4 - I_{j+1,m,f}$$

$$\text{where } a_4 = W_{f,j}(t_0 - \pi_j(t_0), t_4 - \pi_j(t_4))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t_4 - \pi_j(t_4))/r_m.$$

- If flow m is *not* backlogged at t_2 , then from Lemma 2, we have:

$$\begin{aligned} \frac{W_{m,j+1}(t_0, t_2)}{r_m} &\geq \frac{W_{f,j+1}(t_0, t_2)}{r_f} - a_2 \\ &\geq \frac{W_{f,j+1}(t_0, t_2)}{r_f} - a_2 - I_{j+1,m,f} \end{aligned}$$

$$\text{where } a_2 = W_{f,j}(t_0 - \pi_j(t_0), t_2 - \pi_j(t_2))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t_2 - \pi_j(t_2))/r_m.$$

Therefore, in either case, there exists some $t'' \in [t_0, t_2]$, such that:

$$\frac{W_{m,j+1}(t_0, t_2)}{r_m} \geq \frac{W_{f,j+1}(t_0, t_2)}{r_f} - a'' - I_{j+1,m,f} \quad (8)$$

$$\text{where } a'' = W_{f,j}(t_0 - \pi_j(t_0), t'' - \pi_j(t''))/r_f - W_{m,j}(t_0 - \pi_j(t_0), t'' - \pi_j(t''))/r_m.$$

Difference in normalized throughput during $[t_1, t_2]$:

From (7) and (8), we get:

$$\begin{aligned} \frac{W_{f,j+1}(t_1, t_2)}{r_f} &= \frac{W_{f,j+1}(t_0, t_2)}{r_f} - \frac{W_{f,j+1}(t_0, t_1)}{r_f} \\ &\leq \frac{W_{m,j+1}(t_0, t_2)}{r_m} + a'' + I_{j+1,m,f} \\ &\quad - \frac{W_{m,j+1}(t_0, t_1)}{r_m} - a' + I_{j+1,f,m} \end{aligned}$$

From the induction hypothesis and Lemma 3, we know that: $-(U_{1,\{f,m\}} + \sum_{h=2}^j (I_{h,f,m} + I_{h,m,f})) \leq a'' - a' \leq U_{1,\{f,m\}} + \sum_{h=2}^j (I_{h,f,m} + I_{h,m,f})$. Therefore, we get:

$$\begin{aligned} \frac{W_{f,j+1}(t_1, t_2)}{r_f} &\leq \frac{W_{m,j+1}(t_1, t_2)}{r_m} + U_{1,\{f,m\}} \\ &\quad + \sum_{h=2}^{j+1} (I_{h,f,m} + I_{h,m,f}) \end{aligned}$$

Thus, the induction step is proved for server $j + 1$.

Hence, the theorem follows by mathematical induction. \blacksquare
The following corollary follows from Definitions 1, 2, 3, and Theorem 1.

Corollary 1 A network of FT servers provides an end-to-end fairness guarantee with $U_{H,\{f,m\}}^{net} = U_{1,\{f,m\}} + \sum_{h=2}^j (I_{h,f,m} + I_{h,m,f})$.

Corollary 1 states that the unfairness measure for a network of servers is a linear function of the unfairness measures of the individual servers. An interesting property of the guarantee is that unlike end-to-end guarantees on delay [13] and throughput [15], which in addition, are linear functions of link propagation latencies, the bound on end-to-end fairness in (5) is independent of link propagation latencies. It follows that if the number and types of servers on a typical path in a wide-area network is similar to those in a local-area network, then the end-to-end fairness guarantees of the two networks would also be similar.

It is important to observe that the analysis of Theorem 1 can be used to derive end-to-end fairness guarantees even for

flows that are *not* continuously backlogged at the first servers. The only condition that needs to be satisfied at the first server is that the difference in normalized throughput of flow f and m at the first server is bounded. This may be true even if the flows are not continuously backlogged, for instance in the case when the difference in (normalized) number of bits that arrive from the respective sources is bounded. To obtain the end-to-end fairness guarantee for such cases, $U_{1,\{f,m\}}$ is replaced in (5) by the bound on difference in normalized throughput at the first server.

6 Related Work

A large number of fair scheduling algorithms have been proposed over the last decade, and the literature is abundant in analyses that establish the fairness properties of these algorithms at a single server [4, 5, 10, 12, 14, 19, 22]. However, the literature does not contain any end-to-end fairness analyses for a network of fair servers.

A large number of powerful and general techniques have also been developed in the recent past for conducting end-to-end analyses of the network properties exported by large classes of scheduling algorithms [1, 2, 3, 7, 13, 17, 20]. Unfortunately, most of these end-to-end analyses are restricted to deriving guarantees on *absolute* metrics—such as delay, throughput, jitter—that are defined for a single flow. Fairness is a *relative* metric, which is defined in relation to the service received by other flows. End-to-end analysis techniques of the past—for instance those based on the definition of service curves—have not been developed for such relative service guarantees.

7 Concluding Remarks

Fairness of bandwidth allocation among competing flows is an important property of packet scheduling algorithms. A fair scheduling algorithm allows routers (1) to provide throughput guarantees to backlogged flows at short time-scales, independent of past usage of the link bandwidth by the flows; and (2) to allocate idle link capacity to competing flows in proportion to their weights (or reserved rates). Although the single-server fairness properties of several fair scheduling algorithms are well-understood, the literature does not contain any end-to-end fairness analysis of a network of such fair servers.

In this paper, we present the *first* end-to-end fairness analysis of a network of fair servers. We first argue that it is difficult to extend existing single-node fairness analysis to an end-to-end analysis of a network where each node may employ a different fair scheduling algorithm. We then present a two-step approach for end-to-end fairness analysis of heterogeneous networks. First, we define a class of scheduling algorithms, referred to as the *Fair Throughput (FT)* class, and prove that most known scheduling algorithms belong to this class. Second, we develop an analysis methodology for de-

termining the end-to-end fairness bounds for a network of FT servers. Our analysis is general and can be applied to heterogeneous networks where different nodes employ different scheduling algorithms from the FT class.

References

- [1] R. Agrawal, R.L. Cruz, C.M. Okino, and R. Rajan. A Framework for Adaptive Service Guarantees. In *Proceedings of Allerton Conference on Comm., Control, and Comp., Monticello, IL*, Sept 1998.
- [2] R. Agrawal, R.L. Cruz, C.M. Okino, and R. Rajan. Performance Bounds for Flow Control Protocols. In *IEEE/ACM Transactions on Networking*, volume 7, pages 310–323, June 1999.
- [3] J.C.R. Bennett, K. Benson, A. Charny, W.F. Courtney, and J.Y. LeBoudec. Delay Jitter Bounds and Packet Scale Rate Guarantee for Expedited Forwarding. to appear in *IEEE/ACM Transactions on Networking*.
- [4] J.C.R. Bennett and H. Zhang. WF^2Q : Worst-case Fair Weighted Fair Queuing. In *Proceedings of INFOCOM'96*, pages 120–127, March 1996.
- [5] J.C.R. Bennett and H. Zhang. Hierarchical Packet Fair Queuing Algorithms. In *IEEE/ACM Transactions on Networking*, volume 5, pages 675–689, Oct 1997.
- [6] Z. Cao, Z. Wang, and E. Zegura. Rainbow Fair Queuing: Fair Bandwidth Sharing Without Per-Flow State. In *Proceedings of IEEE INFOCOM*, March 2000.
- [7] C.S. Chang. Performance Guarantees in Communication Networks. *Spring-Verlag, New York, NY*, 2000.
- [8] A. Clerget and W. Dabbous. TUF: Tag-based Unified Fairness. In *Proceedings of IEEE INFOCOM*, April 2001.
- [9] A. Demers, S. Keshav, and S. Shenker. Analysis and Simulation of a Fair Queuing Algorithm. In *Proceedings of ACM SIGCOMM*, pages 1–12, September 1989.
- [10] S.J. Golestani. A Self-Clocked Fair Queuing Scheme for High Speed Applications. In *Proceedings of INFOCOM'94*, 1994.
- [11] P. Goyal. Packet Scheduling Algorithms for Integrated Services Networks. *PhD thesis, University of Texas at Austin, Austin, TX*, August 1997.
- [12] P. Goyal and H.M. Vin. Fair Airport Scheduling Algorithms. In *Proceedings of International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'97)*, pages 273–282, May 1997.

- [13] P. Goyal and H.M. Vin. Generalized Guaranteed Rate Scheduling Algorithms: A Framework. In *IEEE/ACM Transactions on Networking*, volume 5, pages 561–571, August 1997. Also available as technical report TR95-30, Department of Computer Sciences, The University of Texas at Austin.
- [14] P. Goyal, H.M. Vin, and H. Cheng. Start-time Fair Queuing: A Scheduling Algorithm for Integrated Services Packet Switching Networks. In *Proceedings of ACM SIGCOMM'96*, pages 157–168, August 1996.
- [15] J. Kaur and H.M. Vin. Core-stateless Guaranteed Throughput Networks. *Technical Report TR-01-47, Department of Computer Sciences, University of Texas at Austin*, November 2001.
- [16] L. Kleinrock. *Queueing Systems, Volume 1: Theory*. John Wiley & Sons, New York, NY, 1975.
- [17] J.Y. LeBoudec and P. Thiran. *Network Calculus. Spring-Verlag Lecture Notes in Computer Science*, 2050, July 2001.
- [18] R. Pan, B. Prabhakar, and K. Psounis. CHOKe, A Stateless Active Queue Management Scheme for Approximating Fair Bandwidth Allocation. In *Proceedings of IEEE INFOCOM*, March 2000.
- [19] A.K. Parekh. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks*. PhD thesis, Department of Electrical Engineering and Computer Science, MIT, 1992.
- [20] D. Stiliadis and A. Verma. Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms. In *IEEE/ACM Transactions on Networking*, volume 6, pages 611–624, October 1998.
- [21] I. Stoica. Stateless Core: A Scalable Approach for Quality of Service in the Internet. *PhD thesis, Carnegie Mellon University, Pittsburgh, PA*, December 2000.
- [22] I. Stoica, H. Zhang, and T.S.E. Ng. A Hierarchical Fair Service Curve Algorithm for Link-Sharing, Real-Time and Priority Services. In *Proceedings of ACM SIGCOMM, Cannes, France*, pages 249–262, 1997.
- [23] H. Zhang and S. Keshav. Comparison of Rate-Based Service Disciplines. In *Proceedings of ACM SIGCOMM*, pages 113–121, August 1991.
- [24] L. Zhang. VirtualClock: A New Traffic Control Algorithm for Packet Switching Networks. In *Proceedings of ACM SIGCOMM'90*, pages 19–29, August 1990.

A Algorithms in the FT Class

A.1 SCFQ [10]

Let $v(t_1, t_2)$ be the difference in system virtual times at t_1 and $t_2 \geq t_1$. Let $v_f(t_1, t_2)$ be the difference in virtual time of flow f at t_1 and t_2 .

Since flow m is continuously backlogged throughout $[t_1, t_2]$, we have from Corollary 4 of [10]:

$$\frac{W_m(t_1, t_2)}{r_m} \geq v(t_1, t_2) - \frac{l_m^{max}}{r_m} \quad (9)$$

Consider flow f . Break $[t_1, t_2]$ into sub-intervals belonging to two sets: B , the set of sub-intervals during which f is continuously backlogged, and NB , the set of sub-intervals during which f is not backlogged. From Definition 5, the differential service lag function for flow f , δ_f , is defined as:

$$\begin{aligned} \delta_f(t_1, t_2) &= v(t_1, t_2) - v_f(t_1, t_2) \\ &= v(t_1, t_2) - \sum_{[t', t''] \in B} v_f(t', t'') \\ &\quad - \sum_{[t', t''] \in NB} v_f(t', t'') \\ &= v(t_1, t_2) - \sum_{[t', t''] \in B} W_f(t', t'') \\ &\quad - \sum_{[t', t''] \in NB} v(t', t'') \end{aligned}$$

where we use Definition 3 and 4 of [10]. From Lemma 2 of [10], we know that $v(t)$ is a non-decreasing function of time. Therefore,

$$\begin{aligned} \delta_f(t_1, t_2) &\leq v(t_1, t_2) - \sum_{[t', t''] \in B} W_f(t', t'') \\ &\leq v(t_1, t_2) - W_f(t_1, t_2) \end{aligned} \quad (10)$$

From Theorem 1 of [10], we know that $\delta_f(t_1, t_2) \geq \frac{l_f^{max}}{r_f}$. Therefore, we get:

$$v(t_1, t_2) \geq W_f(t_1, t_2) - \frac{l_f^{max}}{r_f} \quad (11)$$

From (9) and (11) we get:

$$\frac{W_f(t_1, t_2)}{r_f} \leq \frac{W_m(t_1, t_2)}{r_m} + \frac{l_m^{max}}{r_m} + \frac{l_f^{max}}{r_f}$$

Therefore, for an SCFQ server, $U_{j, \{f, m\}} = I_{j, m, f} = \frac{l_m^{max}}{r_m} + \frac{l_f^{max}}{r_f}$.

A.2 SFQ [14]

Let v_1 and v_2 , respectively, be the virtual times at t_1 and t_2 . Since flow m is backlogged throughout $[t_1, t_2]$, we have from Lemma 1 of [14]:

$$W_m(t_1, t_2) \geq r_m(v_2 - v_1) - l_m^{max} \quad (12)$$

From Lemma 2 of [14], we have for flow f :

$$W_f(t_1, t_2) \leq r_f(v_2 - v_1) + l_f^{max} \quad (13)$$

From (12) and (13), we get:

$$\begin{aligned} \frac{W_m(t_1, t_2)}{r_m} + \frac{l_m^{max}}{r_m} &\geq v_2 - v_1 \geq \frac{W_f(t_1, t_2)}{r_f} - \frac{l_f^{max}}{r_f} \\ \Rightarrow \frac{W_f(t_1, t_2)}{r_f} &\leq \frac{W_m(t_1, t_2)}{r_m} + \frac{l_m^{max}}{r_m} + \frac{l_f^{max}}{r_f} \end{aligned}$$

Therefore, for an SFQ server, $U_{j,\{f,m\}} = I_{j,m,f} = \frac{l_m^{max}}{r_m} + \frac{l_f^{max}}{r_f}$.

A.3 WF²Q [4]

According to Theorem 1 in [4], the work done for a flow f at a WF²Q server, W_f , is related in the following ways to the work done for the flow in a corresponding GPS [16] server, W_f^{GPS} :

$$W_f^{GPS}(t_1, t_2) - W_f^{WF^2Q}(t_1, t_2) \leq l^{max} \quad (14)$$

$$W_f^{WF^2Q}(t_1, t_2) - W_f^{GPS}(t_1, t_2) \leq \left(1 - \frac{r_f}{C}\right)l_f^{max} \quad (15)$$

Given two flows f and m in a GPS server, of which flow m is continuously backlogged during a time interval $[t_1, t_2]$ (and flow f is not necessarily backlogged), we have the following:

$$\frac{W_f^{GPS}(t_1, t_2)}{r_f} \leq \frac{W_m^{GPS}(t_1, t_2)}{r_m}$$

Using (14) and (15), we get:

$$\begin{aligned} \frac{W_f^{WF^2Q}(t_1, t_2) - \left(1 - \frac{r_f}{C}\right)l_f^{max}}{r_f} &\leq \frac{W_f^{GPS}(t_1, t_2)}{r_f} \\ &\leq \frac{W_m^{GPS}(t_1, t_2)}{r_m} \\ &\leq \frac{W_m^{WF^2Q}(t_1, t_2) + l^{max}}{r_m} \end{aligned}$$

Therefore,

$$\frac{W_f^{WF^2Q}(t_1, t_2)}{r_f} \leq \frac{W_m^{WF^2Q}(t_1, t_2)}{r_m} + \frac{l^{max}}{r_m} + \frac{l_f^{max}}{r_f} - \frac{l_f^{max}}{C}$$

Therefore, for a WF²Q server:

$$\begin{aligned} I_{j,m,f} &= \frac{l^{max}}{r_m} + \frac{l_f^{max}}{r_f} - \frac{l_f^{max}}{C} \\ U_{j,\{f,m\}} &= \max \left\{ \frac{l^{max}}{r_f} + l_m^{max} \left(\frac{1}{r_m} - \frac{1}{C} \right), \right. \\ &\quad \left. \frac{l^{max}}{r_m} + l_f^{max} \left(\frac{1}{r_f} - \frac{1}{C} \right) \right\} \\ &\leq l^{max} \left(\frac{1}{r_m} + \frac{1}{r_f} - \frac{1}{C} \right) \end{aligned}$$