Resource Allocation Between Persistent and Transient Flows

Supratim Deb, Ayalvadi Ganesh, and Peter Key

***Abstract*—The flow control algorithms currently used in the Internet have been tailored to share available capacity between users on the basis of the physical characteristics of the network links they use rather than the characteristics of their applications. However, real-time applications typically have very different requirements from file transfer or Web browsing, and treating them identically can result in a perception of poor quality of service even when adequate bandwidth is available. This is the motivation for differentiated services. In this paper, we explore service differentiation between persistent (fixed duration) and transient (fixed volume) flows, and also between transient flows of markedly different sizes; the latter is stimulated by current discussion on Web mice and elephants. We propose decentralized bandwidth allocation algorithms that can be implemented by end-systems without requiring the support of a complex network architecture, and show that they achieve performance very close to what is achievable by the optimal centralized scheme.**

***Index Terms*—Bandwidth allocation, decentralized control, service differentiation, shortest processing time, weighted processor sharing.**

I. Introduction

FLOW CONTROL in the Internet is currently performed by TCP. This protocol performs two important functions: it detects and responds to congestion, and it aims to achieve a *fair* allocation of network capacity among users. As TCP is oblivious to the nature of the application transferring data, it aims to achieve fairness at a rather fine-grained level of instantaneous capacity allocation. This has worked well in an environment consisting of fairly homogeneous applications, with similar characteristics and requirements. However, as the diversity of applications on the Internet increases, there is a need for additional mechanisms to provide a quality of service (QoS) appropriate to the application and specific to its requirements. A one-size-fits-all approach can result in applications seeing a poor QoS even when adequate capacity is available to provide each application with a good QoS. This motivated service differentiation efforts such as Diffserv.

In this paper, we consider specifically the problem of sharing network capacity between file transfers and real-time traffic such as Internet telephony or video conferencing. Real-time flows are usually long-lived and can be treated as persistent sources for purposes of analysis, in a way we make precise in the next section. They have very different quality of service requirements from file transfers. Whereas what matters for file transfers is usually the transfer time, or equivalently, average throughput over the entire transfer period, real-time flows typically care about the rate they receive at each instant in time (or, more precisely, averages over time periods much shorter than the lifetime of the connection). The value of capacity allocated to a user can be described mathematically by a utility function which captures elements of the quality of service perceived by the user [24]. The resource allocation problem can be cast as one of maximizing the aggregate utility of all users.

We model the utility for a file transfer as the negative of the time taken to complete the transfer. For real-time traffic, we assume that the total utility obtained is the integral of an instantaneous utility over the lifetime of the connection; the instantaneous utility, in turn, is modeled as an increasing and concave function of the rate received by the flow at that instant. This model is popular in the literature, and sources with such a utility function are termed *elastic*. There has been considerable recent work on sharing capacity between persistent elastic users [15], [17], [19], [21]. However, the problem of combining such sources with transient sessions such as file transfers has received little attention. One recent study [16] suggests that, when the two traffic types share a network, file transfers should receive priority.

A second, related problem we address is that of sharing network capacity between file transfers of markedly different sizes. TCP does not differentiate on the basis of file size. We ask whether it is possible to improve performance for small files without significantly degrading it for large files. This question assumes particular importance in the context of the finding by a number of researchers that file sizes on the Web have a heavy-tailed distribution [4]: when file sizes vary over several orders of magnitude, treating all file transfers identically may not be appropriate.

The main contributions of this paper are as follows. We consider a single bottleneck link of capacity Nc shared by N persistent flows, and by transient flows arriving at the points of a Poisson process of rate $N\lambda$. In Section II, we propose an objective function which takes into account both the utility of persistent flows and the mean delay of transient flows in the system. We obtain bounds on the optimal cost in Section III. In Section IV, we show that certain easy-to-implement sub-optimal-policies are close to optimal in the large N regime. In Section V, motivated by recent work on biasing in favor of short jobs in a queueing system [2], [10], we propose a weighted bandwidth sharing policy which favors small file transfers and study its performance analytically and through simulations. We conclude in Section VI.

Manuscript received December 30, 2002; approved by IEEE/ACM Transactions on Networking Editor S. Low.

S. Deb was with the Laboratory for Information and Decision Systems, MIT, Cambridge, MA 02139 USA. He is now with Bell-Labs Research India, Bangalore 560017, India (e-mail: supratim@lucent.com).

A. Ganesh and P. Key are with Microsoft Research, Cambridge, CB3 0FB U.K. (e-mail: ajg@microsoft.com; peterkey@microsoft.com).

Digital Object Identifier 10.1109/TNET.2005.845544

II. Model and Problem Formulation

We consider a single link shared by two different kinds of flow. Flows of the first kind each remain in the system for a "fixed" duration (the duration may be random, but is independent of the capacity allocated), while flows of the second kind each have a fixed volume of data to transfer. The former typically describe real-time streaming media while the latter correspond to file transfers. These two kinds of applications have different quality of service requirements, which will be reflected in our model. For simplicity, we shall assume that the number of fixed-duration flows remains constant; we shall refer to such flows as *persistent*, and we shall refer to fixed-volume flows as *short* or *transient* flows. Note that it is not important for the validity of this assumption that individual "persistent" flows be long-lived relative to individual "short" flows. What is important is that the total number of persistent flows should remain nearly constant over the typical lifetime of a short flow.

Consider a single link of capacity Nc shared by a fixed number, N, of persistent flows and a variable number of short-lived flows. We are typically interested in high-capacity links multiplexing a large number of flows, i.e., in a large N regime. The persistent flows are modeled as belonging to one of J classes, with a proportion α_j in class j, and with each flow in class j having an increasing and strictly concave instantaneous utility function u_j. If Nx is the aggregate rate allocated to the persistent flows, then the aggregate utility received by these flows can be modeled as $Nu(x)$, where

$$u(x) = \sup\left\{\sum_{j=1} \alpha_j u_j(x_j) : \sum_{j=1} \alpha_j x_j = x\right\}.$$

In words, $Nu(x)$ is a sup convolution of the individual utilities; it is the utility that would be achieved by sharing the capacity Nx optimally among the persistent flows. It can be shown that $u(\cdot)$ is also increasing and strictly concave. Henceforth, we shall simply assume that the aggregate utility of persistent flows can be modeled as $Nu(x)$ for some increasing and strictly concave u. The total utility over a time period is given by the integral of the instantaneous utility over that period.

Such concave utility functions were introduced by Shenker [24] to describe elastic users, and have subsequently been used in [11], [12], [20], and [26], for example. Elastic traffic is normally thought of as delay-insensitive, or data transfer, whereas here we are characterizing *real-time* traffic as elastic. This is appropriate for real-time traffic which has adaptive codecs, or which seeks to mimic the behavior of congestion-controlled data, such as "TCP-friendly" rate control schemes [22]. Indeed, a simple approximation to TCP is to use a utility function of the form $u_j(x) = -1/(T_j^2 x)$ where T_j is the round-trip time for a class j flow [17].

We also note that it is enough if the utility functions are concave in some neighborhood of the typical "operating point." If concavity does not hold in the region of interest, then the optimal way to allocate capacity is to perform admission control and provide adequate capacity to the admitted sources. This is a separate topic which we do not treat in this paper. Concavity reflects the diminishing marginal utility of bandwidth; equivalently, it asserts that a fixed constant rate is preferable to a fluctuating rate with the same mean. It also helps to capture the notion of fairness in a utilitarian setting, wherein we define *optimality* in terms of social welfare; that is, an *optimal* allocation is one that maximizes aggregate utility. To see this, note that concavity implies via Jensen's inequality that, in an optimal allocation, all users with the same utility function will get the same amount of bandwidth.

The assumption that the number of persistent flows is fixed is not essential. One alternative would be model these flows as arriving according to a Poisson process of rate $N\nu$ and remaining in the system for a fixed (random) duration with finite mean, $1/\mu$. If these durations (service times) are independent and identically distributed, then the number of persistent flows in the system can be described by an $M/G/\infty$ queue. It is well known for this queue that, for arbitrary service time distributions, the number of flows in the system in equilibrium has a Poisson distribution with mean $N\nu/\mu$. For large N, the distribution concentrates around its mean value, with typical fluctuations being of order $\sqrt{N}$. This observation provides some justification for modeling the number of persistent flows in a large system as constant, irrespective of the duration of a single flow.

The problem of sharing capacity optimally between persistent sources has been studied in [15] and [17]–[19], for example. Therefore, in this paper, we assume that any capacity share allocated to persistent flows is optimally shared between them, and concentrate on how to partition the available capacity between persistent and transient flows, as well as how to share it among the transient flows. To simplify technicalities in the analysis below, we assume in addition that $u(\cdot)$ is differentiable.

Short flows correspond to file transfers. They arrive into the system at the points of a Poisson process of rate $N\lambda$ and leave when the file transfer is complete. The file sizes are independent and identically distributed (iid) random variables. We make no assumption about the file size distribution other than that it has a finite mean, f. Without loss of generality, we take $c = 1$ and $f = 1$ for notational convenience, so that λ denotes the normalized load offered by the short flows. We shall assume that $\lambda < 1$; this is necessary to ensure that there is a policy which makes the system stable, i.e., prevents the number of backlogged short flows increasing without bound. There is a unit holding cost per unit time for each short flow in the system. The goal is to maximize the time average of $Nu(x(t)) - n(t)$, where $n(t)$ denotes the number of short flows in the system at time t. To this end, we introduce the performance objective

$$J_\pi(n, \mathbf{f}) = \liminf_{T\to\infty} \frac{1}{T} \times E\left[\int_0^T [Nu(x(t)) - n(t)]\, dt | n(0) = n, f_1, \ldots, f_n\right] \quad (1)$$

where f_i denotes the residual workload of the ith short flow in the system at time 0. We seek a policy π that maximizes this objective. A policy specifies, at each time t, the rate $Nx(t)$ allocated to persistent flows, as well as how to share the remaining capacity, $N(1 - x(t))$, among the short flows. We shall restrict attention to stationary policies under which the stochastic process $(x(t), n(t))$ is ergodic, so that the limit exists in the right-hand side of (1), though it could possibly be $-\infty$. Note that the restriction to such policies is not empty. For example, the policy π that allocates a fixed capacity Nx to persistent flows at all times t, for some $x \in (0, c-\lambda)$, and shares the residual capacity equally among all short flows in the system, has this property: it is clearly stationary, and ergodicity follows from well known properties of the $M/G/1-PS$ queue, where PS stands for processor sharing. Thus, $J_\pi(n, \mathbf{f})$ denotes the long-term average of $Nu(x(t)) - n(t)$ under a policy π when there are initially n short flows in the system, with respective residual workloads $f_1, \ldots, f_n$. Let τ_k denote the sojourn time of the kth short flow to enter the system. By Little's law the time average of $n(t)$ is the same as λ times the average value of τ_k, where these averages are defined as

$$\overline{n} = \lim_{T\to\infty} \frac{1}{T} \int_0^T n(t)dt \quad \overline{\tau} = \lim_{k\to\infty} \frac{1}{k} \sum_{j=1}^{k} \tau_j$$

respectively, along each sample path where the limits exist. The equality $\overline{n} = \lambda\overline{\tau}$ holds for every such sample path. By the assumed ergodicity of $n(t)$, these limits exist, and have the same value, on all but a measure zero subset of sample paths. Thus, we can view the objective as maximizing the utility of real-time flows, subject to a bound on the mean sojourn time of file transfers. The objective function above is precisely the Lagrangian for this optimization problem, where λ can be interpreted as the Lagrange multiplier.

III. Bounds on the Optimal Value Function

We now derive an upper bound on the value of the objective function $J_\pi(n, \mathbf{f})$ that is achievable by any policy π, expressed in Theorem 3.1 below. By comparing this with the value attained by specific sub-optimal policies, we shall show in the next section that those policies are close to optimal.

Let $A(t)$ denote the number of short flows arriving during the time interval $[0, t]$, $B(t) = b_1 + b_2 + \cdots + b_{A(t)}$ the total work brought in by these flows and $W(t)$ the total work due to short flows backlogged at time t. Since $A(t)$ is a Poisson process and the b_i are iid with mean 1, independent of $A(t)$, we have

$$\lim_{t\to\infty} \frac{1}{t} A(t) = N\lambda \text{ a.s.} \quad \lim_{t\to\infty} \frac{1}{t} B(t) = N\lambda \text{ a.s.} \tag{2}$$

where a.s. stands for "almost surely." We now characterize the minimum capacity that has to be allocated to the short flows.

Lemma 3.1: Suppose that, for some strictly increasing function $h : \mathbb{R}_+ \to \mathbb{R}_+$ satisfying $h(x)/x \to \infty$ as $x \to \infty$, we have $E[h(b_1)] < +\infty$. Let $(n, \mathbf{f})$ be an arbitrary initial condition. Suppose that π is a policy which, for some $\epsilon > 0$, satisfies

$$\lim_{T\to\infty} \frac{1}{T} \int_0^T x(t)dt > 1 - \lambda + \epsilon \text{ a.s}$$

Then, $\limsup_{T\to\infty} n(T) = +\infty$, and so $J_\pi(n, \mathbf{f}) = -\infty$.

Remarks: Since $h(x)/x$ can go to infinity arbitrarily slowly, the assumption that $E[h(b_1)] < \infty$ is only slightly stronger than requiring that b_1 have a finite mean. For example, it would suffice if $E[b_1^{1+\epsilon}] < \infty$ for some $\epsilon > 0$. Alternatively, if the file size distribution has a Pareto density $f(x) = c(a + x)^{-\alpha}$, then the assumption holds provided $\alpha > 2$, i.e., the mean file size is finite; here c and a are positive constants.

The lemma says that any policy π which allocates capacity less than $N(\lambda - \epsilon)$ on average to short flows will cause the number of short flows backlogged to grow to infinity. This is not altogether trivial: we cannot rule out, a priori, the possibility that the amount of backlogged work grows to infinity while the number of backlogged flows remains bounded, in expectation.

Proof: Let $W(0) = f_1 + \cdots + f_n$ denote the total residual work due to short flows initially in the system at time 0. For the policy π, we have

$$\begin{aligned} &\lim_{t\to\infty} \frac{1}{t} [W(t) - W(0)] \\ &\quad \geq \lim_{t\to\infty} \frac{1}{t} \left[B(t) - \int_0^t N(1 - x(s))\, ds \right] \\ &\quad > N\epsilon \text{ a.s.} \end{aligned} \tag{3}$$

The first inequality says that the work $W(t) - W(0)$ backlogged between times 0 and t is at least the total work $B(t)$ arriving during this time, less the maximum work that could have been completed, namely, the total capacity $\int_0^t N(1 - x(s))ds$ allocated to short flows during this period.

Observe from (2) and (3) that, for any $\alpha \in (0, 1)$, we have for T sufficiently large that

$$P\left(W(t) - W(0) > N\epsilon t \text{ and } A(t) < N(\lambda+\epsilon)t \; \forall\, t \geq T\right) > \alpha. \tag{4}$$

Denote $E[h(b_1)]$ by m. We have by Markov's inequality and the monotonicity of h that $P(b_1 > z) = P(h(b_1) > h(z)) \leq m/h(z)$ for all $z > 0$. Fix $M > 0$, arbitrarily large. Now, by the independence of the b_i

$$\begin{aligned} &P\left(b_i \leq (N\epsilon t/M) \; \forall\, i = 1, \ldots, \lfloor N(\lambda + \epsilon)t \rfloor\right) \\ &\qquad \geq \left(1 - \frac{m}{h(N\epsilon t/M)}\right)^{N(\lambda+\epsilon)t}. \end{aligned} \tag{5}$$

But the right-hand side goes to 1 as $t \to \infty$ since $h(N\epsilon t/M)$ grows faster than t, by assumption. Now, if the work backlogged at time t is no smaller than $N\epsilon t$, and if no flow contributes more than $N\epsilon t/M$, then the number of backlogged flows must be at least M. Thus, it is immediate from (4) and (5) that, given any $M > 0$ and $\alpha \in (0, 1)$, it holds for all t sufficiently large that

$$P(n(t) > M) > \alpha. \tag{6}$$

Therefore, by the Borel–Cantelli lemma, $n(t) > M$ infinitely often, i.e., $\limsup_{t\to\infty} n(t) \geq M$. Since this holds for arbitrarily large M, the first claim of the claim is established. The second claim follows because $J_\pi(n, \mathbf{f}) \leq Nu(1) - \limsup_{t\to\infty} n(t)$, as $u(x(t)) \leq u(1)$ for all t. This completes the proof of the lemma. ■

We now look at the performance of policies which allocate at least $N\lambda$ to the short flows.

Lemma 3.2: Let $(n, \mathbf{f})$ be an arbitrary initial condition and let π be any policy for which

$$\overline{x} := \lim_{T\to\infty} \frac{1}{T}\int_0^T x(t)dt \leq 1-\lambda \text{ a.s.}$$

the existence of the limit $\overline{x}$ is part of the assumption. Then, $J_\pi(n,\mathbf{f}) \leq Nu((1-\lambda))$.

Proof: Since u was assumed to be concave, we obtain from Jensen's inequality and the nonnegativity of $n(t)$ that

$$\frac{1}{T}\int_0^T [Nu(x(t)) - n(t)]\,dt \leq Nu\left(\frac{1}{T}\int_0^T x(t)dt\right)$$

for every $T > 0$. Taking expectations and using Jensen's inequality once more, we get

$$E\left(\frac{1}{T}\int_0^T [Nu(x(t)) - n(t)]\,dt\right) \leq Nu\left(E\left[\frac{1}{T}\int_0^T x(t)dt\right]\right). \tag{7}$$

Since $(1/T)\int_0^T x(t)dt$ converges to $\overline{x}$ almost surely, and since $x(t) \in [0,1]$ for all t, it follows by dominated convergence that

$$E\left[\frac{1}{T}\int_0^T x(t)dt\right] \to \overline{x}$$

and by the continuity and monotonicity of u that

$$u\left(E\left[\frac{1}{T}\int_0^T x(t)dt\right]\right) \to u(\overline{x}) \leq u(1-\lambda).$$

Substituting this in (7) yields the claim of the lemma, by the definition of J_π. ■

We can now state the following upper bound on the value of any ergodic policy, which is immediate from the above two lemmas.

Theorem 3.1: Suppose the file size distribution satisfies the assumption in Lemma 3.1. Then, for any initial condition $(n, \mathbf{f})$ and any ergodic policy π

$$J_\pi(n,\mathbf{f}) \leq Nu(1-\lambda).$$

In the next section, we shall consider specific sub-optimal policies. By comparing them with the upper bound derived above, we shall show that they are close to optimal, for arbitrary file size distributions. It is difficult to determine the optimal policy in the general setting considered so far. When file sizes are exponentially distributed, [6] and [7] show how to calculate the optimal policy using value iteration, and describe some structural properties of the optimal policy.

IV. Sub-Optimal Policies

In the remainder of this paper, we assume that the utility function u is strictly concave, increasing and twice continuously differentiable. We describe two simple policies below and show that they are close to optimal, in an asymptotic sense as N tends to infinity.

Before we describe the sub-optimal policies, we state our method of evaluating a sub-optimal policy. The question we ask is, how much worse than optimal is a given sub-optimal policy? One way to quantify this is to ask how large a capacity $N\hat{c}$ is needed, so that the total utility achieved using the given sub-optimal policy on a link of capacity N is at least the utility achieved using the optimal policy on a link of capacity $N\hat{c}$. Thus, $(1-\hat{c})$ is the additional capacity required per-persistent flow, for a sub-optimal policy to do as well as the optimal policy. We show that, with a proper choice of parameters, the sub-optimal policies described in this section do as well as the optimal policy for $1-\hat{c} = O(1/\sqrt{N})$.

A. Static Policy

A fixed amount of bandwidth $N\underline{c}$ is reserved for the persistent sources and the remainder, $N(1-\underline{c})$, is shared equally among the transient flows. We shall denote this policy $\pi_S(\underline{c})$, using the subscript S to signify that the bandwidth partitioning is static. This policy involves logically partitioning the link between persistent and transient flows and using a flow-control mechanism that shares capacity equally among the transient flows. The logical partitioning can be implemented, for example, by maintaining separate buffers for traffic from persistent and transient sources and serving these buffers at the specified rates, $N\underline{c}$ and $N(1-\underline{c})$, using a weighted round-robin policy. This can be thought of as a rather simple special case of the Diffserv architecture [3]. With regard to sharing the allocated capacity equally among flows of the same type, TCP approximately achieves this if all flows have the same access bandwidth and round-trip time. Another alternative is to use the scheme of [9] with all flows of a given type (transient or persistent) having the same willingness-to-pay parameter.

Now, irrespective of the file size distribution, the number of short flows in progress evolves like the queue size in an $M/GI/1-PS$ queue, with load $\rho = \lambda/(1-\underline{c})$. The equilibrium queue length distribution is geometric with parameter ρ (see [13], for example), and so the mean number of short flows in progress is $E_\pi[n] = \rho/(1-\rho)$. We thus obtain the following.

Lemma 4.1: Let $a > 0$ be arbitrary, and let $\underline{c} = 1-\lambda - a/\sqrt{N}$. Then

$$E_{\pi_S(\underline{c})}[Nu(x(n)) - n] = Nu(1-\lambda) - O(\sqrt{N}).$$

Proof: The load in the queue serving short flows is

$$\rho = \frac{N\lambda}{N(1-\underline{c})} = \frac{\lambda}{\lambda + \left(a/\sqrt{N}\right)}$$

and so the mean number of short flows in the system is given by

$$E_{\pi_S(\underline{c})}[n] = \frac{\rho}{1-\rho} = \frac{\lambda}{a}\sqrt{N}. \tag{8}$$

On the other hand, each persistent flow receives capacity $\underline{c}$, and so the mean utility of persistent flows is given by

$$\begin{aligned} E_{\pi_S(\underline{c})}[Nu(x)] &= Nu(\underline{c}) = Nu\left(1-\lambda-a/\sqrt{N}\right) \\ &= Nu(1-\lambda) - a\sqrt{N}u'(y), \end{aligned} \tag{9}$$

for some y in the interval $[1-\lambda-a/\sqrt{N}, 1-\lambda]$. Now, by (8) and (9)

$$E_{\pi_S(\underline{c})}[Nu(x) - n] = Nu(1-\lambda) - [au'(y) + \lambda/a]\sqrt{N}.$$

Since the continuous function u' is bounded on the compact interval $[1-\lambda-(a/\sqrt{N}), 1-\lambda]$, the claim of the lemma is established. ■

How much worse than optimal is the static policy? As noted before, we ask, how large a capacity $N\hat{c}$ is needed, so that the total utility achieved using the static policy on a link of capacity N is the same as the utility achieved using the optimal policy on a link of capacity $N\hat{c}$. Comparing Lemmas 3.2 and 4.1, it is clear that $\hat{c} = 1 - O(1/\sqrt{N})$. In so far as N is large in the typical operating regime of interest, this shows that the static policy is close to optimal.

We now discuss the intuition behind the choice of parameters in the static policy. Recall that $N\lambda$ is the rate at which work is brought in by short flows, and $N\lambda/\rho$ is the capacity allocated to them. The choice $\rho = 1 - a/\sqrt{N}$ corresponds to allocating short flows just a little more than the minimum required to ensure that all short flows eventually leave the system. In other words, the queue of short flows operates in a heavy traffic regime. Nevertheless, in this regime, the number of short flows present in the system is of order $\sqrt{N}$ on average. Hence, each short flow typically gets $O(\sqrt{N})$ bandwidth whereas each persistent flow, of which there are N, gets only $O(1)$ bandwidth. Thus, from the perspective of an individual short flow, it receives higher priority than persistent flows, though short flows on aggregate don't receive higher priority. The point to note is that, if all short flows are to be served, then the average service rate allocated to them should be at least the rate at which they are bringing in work. In a large system, this rate is large compared to the work brought in by a single flow. Thus, by serving short flows at this rate, the number of short flows and their mean sojourn time is kept small, and consequently they perceive a good quality of service. At the same time, the aggregate rate allocated to short flows is close to the minimum possible, so the persistent flows achieve very nearly the maximum utility they can get in any stable system.

We shall observe the same qualitative features in the weighted processor sharing policy we consider in the next subsection. This, too, is a simple policy that is practicable in decentralized systems, and can achieve near-optimal performance.

In fact, the near-optimality of the static policy can be expected to hold even if the arrival process of short flows is not Poisson, but a general renewal process. Recall that the arrivals constitute a renewal process if the inter-arrival times are iid, with arbitrary distribution. The number of short flows in the system thus evolves like a $GI/GI/1 - PS$ queue. Moreover, this queue is operated in a heavy traffic regime: the service capacity $N(1 - \underline{c})$ is chosen so that the load on the queue, given by $\rho^N = (N\lambda)/(N(1 - \underline{c}))$ satisfies

$$\rho^N = 1 - a/\sqrt{N}, \quad \text{i.e.,} \quad \sqrt{N}(1 - \rho^N) \to a.$$

We have superscripted ρ by N to make explicit that we are considering a sequence of systems indexed by N. Let $n^N(t)$ denote the queue size (number of short flows in progress) at time t in the system indexed by N, and define the scaled queue size process $\hat{n}^N(t) = n^N(Nt)/\sqrt{N}$. It is shown by Gromoll [8] in this scaling regime that, if the inter-arrival times have $2+\epsilon$ finite moments and the service times have $4 + \epsilon$ finite moments, for some $\epsilon > 0$, then the scaled queue size process $\hat{n}^N(\cdot)$ converges in distribution (as $N \to \infty$) to a reflected Brownian motion with negative drift, which we denote $R(\cdot)$. Now, for the process $R(\cdot)$, $(1/T)\int_0^T R(t)dt$ converges almost surely to a finite limit as $T \to \infty$. This leads us to conjecture that the same holds true for the scaled process $\hat{n}^N(\cdot)$, and hence that, for the unscaled process

$$\frac{1}{T}\int_0^T n^N(t)dt \to b_N\sqrt{N} \quad \text{as } T \to \infty \tag{10}$$

where b_N is a sequence of constants converging in turn to a constant b as $N \to \infty$. Note that Gromoll has established convergence of $\hat{n}^N(\cdot)$ to $R(\cdot)$ in distribution but not the convergence of moments, so (10) is indeed a conjecture. If the conjecture holds, then it follows as in the Poisson case studied in Lemma 4.1 that the static policy achieves a value for the objective function that is within $O(\sqrt{N})$ of $Nu(1 - \lambda)$, which is an upper bound on the optimal value. Thus, the static policy is close to optimal.

Implementation of the static policy requires that bandwidth partitioning be carried out by network routers. It also requires knowledge of the rate at which work is brought in by short flows, or a fairly accurate estimate of this rate.[1] In contrast, the weighted processor sharing policy discussed next can be implemented at end systems (although it may require information from the routers as to the number of persistent flows in progress, N), and does not require knowledge of the traffic characteristics of short flows.

B. Weighted Processor Sharing

Suppose each persistent source has weight 1 and each file transfer in progress has weight w, and that capacity is shared between users in proportion to their weights. In particular, each file transfer in progress gets the same share of capacity. We shall call this the weighted processor sharing or WPS policy, and denote it $\pi_W(w)$. We will show that, with an appropriate choice of w, the total utility achieved using weighted processor sharing policy on a link of capacity N is the same as the utility achieved using the optimal policy on a link of capacity $N\hat{c}$ if $1 - \hat{c} = O(1/\sqrt{N})$.

Now, irrespective of the file size distribution, the number of file transfers in progress can be modeled by a symmetric queue, and has the invariant distribution of a birth-death process with constant birth rate $N\lambda$, and state-dependent death rate $N\mu_n = N(1 - x(n))$ (see [13, Lemma 3.9]). Here $Nx(n)$ is the aggregate capacity allocated to persistent sources when n short flows are in progress. The invariant distribution for this process is given by

$$\pi(n) = \pi(0)\frac{\lambda^n}{\mu_1 \cdots \mu_n}, \qquad \sum_{n=0}^{\infty}\pi(n) = 1. \tag{11}$$

Let $K := N/w$. Then

$$x(n) = \frac{N}{N + nw} = \frac{K}{K + n} \quad \text{and} \quad \mu_n = 1 - x(n) = \frac{n}{K + n}$$

from which it follows that

$$\pi(n) = \pi(0)\lambda^n\prod_{i=1}^{n}\frac{K + i}{i} = \pi(0)\frac{\Gamma(K + n + 1)}{\Gamma(K + 1)\Gamma(n + 1)}\lambda^n \tag{12}$$

where $\pi(0)$ is determined by the requirement that $\pi(n)$ sums to one. Here, $\Gamma(\cdot)$ is the Gamma function, defined on $(0, \infty)$

[1] An adaptive scheme that adjusts the bandwidth partitioning based on measurements of the load due to short flows may be able to achieve comparable performance. Investigating this is a topic for future research.

by $\Gamma(a) = \int_0^\infty x^{a-1}e^{-x}dx$; $\Gamma(a) = (a-1)!$ if a is a positive integer. The Gamma function satisfies the recursion $\Gamma(a) = (a-1)\Gamma(a-1)$ for all real $a > 1$. Note that

$$\begin{aligned}\sum_{n=0}^{\infty} \frac{\Gamma(K+n+1)}{\Gamma(K+1)\Gamma(n+1)}\lambda^n &= \sum_{n=0}^{\infty} \frac{(-K-1)(-K-2)\dots(-K-n)}{n!}(-\lambda)^n \\ &= \frac{1}{(1-\lambda)^{K+1}} \qquad (13)\end{aligned}$$

by the generalized binomial expansion of $(1-\lambda)^{-(K+1)}$. Now, substituting (13) in (12) and using the fact that $\sum_n \pi(n) = 1$, we obtain $\pi(0) = (1-\lambda)^{K+1}$. Substituting this in (12) yields

$$\pi(n) = \frac{\Gamma(K+n+1)}{\Gamma(K+1)\Gamma(n+1)}\lambda^n(1-\lambda)^{K+1}. \qquad (14)$$

We shall use this to obtain an analogue of Lemma 4.1, showing that the weighted processor sharing policy is nearly optimal if w is chosen appropriately. We first need some technical results.

The generating function of n can be computed from (14) as follows:

$$G(z) := E[z^n] = \sum_{n=0}^{\infty} \pi(n)z^n = \left(\frac{1-\lambda}{1-\lambda z}\right)^{K+1}. \qquad (15)$$

The mean number of short flows in the system is given by

$$\begin{aligned}E_{\pi_W(w)}[n] &= G'(1) = (K+1)\lambda \left.\frac{1-\lambda}{(1-\lambda z)^2}\right|_{z=1} \\ &= (K+1)\frac{\lambda}{1-\lambda}. \qquad (16)\end{aligned}$$

Now, by the recursion $\Gamma(a+1) = a\Gamma(a)$, we have

$$\frac{\pi(n+1)}{\pi(n)} = \frac{K+n+1}{n+1}\lambda$$

so that $\pi(n+1) > \pi(n)$ if and only if $(K+n+a)\lambda > n+1$. Rearranging terms, the maximum value of $\pi(n)$ is seen to be attained at $n^* := \lfloor K\lambda/(1-\lambda)\rfloor$. The next lemma says that, if w is chosen so that K is large, then the probability distribution π concentrates around its mode, n^*.

Lemma 4.2: Let $a > 0$ be arbitrary and let $w = a\sqrt{N}$, so that $K = \sqrt{N}/a$. Let n^* be the integer part of $K\lambda/(1-\lambda)$. (The dependence of w, K and n^* on N has not been made explicit in the notation). For any $\delta > 0$, there are constants $c_1, c_2 > 0$, not depending on N, such that

$$\sum_{m=(1+\delta)n^*}^{\infty} \pi(n) \le c_1 e^{-c_2\sqrt{N}}, \qquad \sum_{m=0}^{(1-\delta)n^*} \pi(n) \le c_1 e^{-c_2\sqrt{N}}$$

for all N sufficiently large.

The proof is relegated to the Appendix.

Notice that the bandwidth allocated to persistent flows by the WPS policy decreases to zero as the number of short flows in the system increases to infinity. If $u(x) \to -\infty$ as $x \to 0$, then the contribution to $E_{\pi_W(w)}[u(x)]$ due to such events could be arbitrarily large and negative. In order to avoid this, we need the following technical assumption which controls the speed at which $u(x) \to -\infty$ as $x \to 0$.

Assumption A: Suppose the utility function $u(\cdot)$ is such that, for some $\eta > 0$ and some $\delta > 0$, we have

$$\int_{\delta}^{\infty} e^{-\eta x} u\left(\frac{1}{x}\right) dx > -\infty.$$

Note that if the above inequality holds for some $\delta > 0$, then it holds for all $\delta > 0$. Equivalently, $\liminf_{x\to\infty} e^{-\eta x}u(1/x) > -\infty$ for some $\eta > 0$.

This assumption says that $-u(1/x)$ does not grow superexponentially in x as $x \to 0$. It is not terribly restrictive; for example, it is satisfied by utility functions of the form $u(x) = x^{1-\beta}/(1-\beta)$ for $\beta > 0$, and also by $u(x) = \log(x)$. It is also satisfied by any utility function which is bounded below.

The next lemma establishes the near-optimality of the WPS policy.

Lemma 4.3: Let $a > 0$ be arbitrary, and let $w = a\sqrt{N}$ be the relative weight assigned to each short flow. If $u(\cdot)$ satisfies Assumption A, then

$$E_{\pi_W(w)}\left[Nu\left(x(n)\right) - n\right] = Nu(1-\lambda) - O(\sqrt{N}).$$

The proof can be found in the Appendix.

It is easy to see that the capacity $N\hat{c}$ required for an optimal policy to outperform weighted processor sharing satisfies $\hat{c} = 1 - O(1/\sqrt{N})$. This is similar to what we obtained for the static policy and shows that WPS is asymptotically optimal in the same sense.

An advantage of the WPS policy over the static policy is that the optimal choice of w requires knowledge only of N and not of the load offered by short flows, λ. Moreover, the knowledge of N can be imperfect: if we have an estimate of N which is only within a multiplicative constant of its actual value, the resulting choice of weights is still near-optimal, and the deviation from optimality is only of order $1/\sqrt{N}$. Thus, the weighted-PS policy is robust and well-suited to practical implementation. Finally, it can be implemented by end systems rather than the network, for example by having end systems use a weighted analogue of TCP with weights chosen as above. An alternative implementation would be to use a willingness-to-pay scheme, as described in [15], with a willingness-to-pay parameter proportional to the weights above. It is still an open problem as to how to estimate N from the end systems.

C. Numerical Results

The analytical results derived above show that the simple policies we have proposed for sharing bandwidth between persistent and transient flows are nearly optimal in large systems, multiplexing a large number of persistent flows. In this section, we explore how large N needs to be for the analysis to be valid, and find that it applies even at very small values of N. In order to obtain numerical results, we need to explicitly specify a utility function $u(\cdot)$; we take $u(x) = -1/x$, which is the form of utility implicitly maximized by TCP [17].

Recall that the static policy assigns bandwidth $\underline{c} = 1 - \lambda - a/\sqrt{N}$ to each persistent flow, where a is a parameter to be chosen. By (8), the mean number of short flows in the system is $\lambda\sqrt{N}/a$. Hence, the value of the objective function for the static policy is

$$E_{\pi_S(\underline{c})}\left[Nu(x) - n\right] = \frac{-N}{1-\lambda-\left(\frac{a}{\sqrt{N}}\right)} - \frac{\lambda}{a}\sqrt{N}.$$

It can be verified that the objective is maximized by taking $a = \sqrt{N\lambda}(1-\lambda)/(\sqrt{N}+\sqrt{\lambda})$; correspondingly, we have

$$E_{\pi_S}\left[Nu\left(x(n)\right)\right] = -\frac{N+\sqrt{N\lambda}}{1-\lambda}, \qquad E_{\pi_S}[n] = \frac{\sqrt{N\lambda}+\lambda}{1-\lambda}. \tag{17}$$

The WPS policy assigns weight $w = a\sqrt{N}$ to each short flow. Consequently, $K = N/w = \sqrt{N}/a$ and, by (16), the mean number of short flows in the system is $(K+1)(\lambda/(1-\lambda))$. We now evaluate the mean utility of persistent flows. Recall that, when there are n short flows in the system, the persistent flows each receive bandwidth $x(n) = K/(K+n)$ and hence derive utility $u(x(n)) = -(1+(n/K))$. Thus, their mean utility is $-(1+(E[n]/K))$. Using (16), we obtain after simplification that

$$E_{\pi_W}\left[Nu\left(x(n)\right) - n\right] = \frac{-N}{1-\lambda} - \frac{\lambda}{1-\lambda}\left[a\sqrt{N} + \frac{\sqrt{N}}{a} + 1\right].$$

It is clear that the maximum value of the objective function is attained when $a = 1$, i.e., each transient flow is given a weight $w = \sqrt{N}$ relative to each persistent flow. For $a = 1$, we have

$$E_{\pi_W}\left[Nu\left(x(n)\right)\right] = -\frac{N+\sqrt{N}\lambda}{1-\lambda}, \qquad E_{\pi_W}[n] = \frac{\sqrt{N}\lambda+\lambda}{1-\lambda}. \tag{18}$$

In Fig. 1, we plot the mean utility of a persistent flow and the mean number of transient flows in the system for (i) the static policy, as given by (17), (ii) weighted PS, as given by (18), (iii) naive PS, which gives an equal share of bandwidth to each flow, persistent or transient, and (iv) the optimal policy, obtained numerically using the techniques described in [6], [7]. If all flows use TCP, then case (iii) approximates the bandwidth shares they obtain. We chose the system parameters $c = 1$, $f = 1$, $N = 25$, and varied the arrival rate, $N\lambda$, of short flows so that the normalized load offered by short flows, $\lambda f/c$, spans the interval [0.1,0.7].

The plots show that both the static and weighted-PS policies achieve near-optimal performance, both for persistent and transient flows. In other words, the static and the weighted-PS policies considered do not just approximate the optimal policy in terms of total utility, but also in terms of utility for each class of flows. On the other hand, the policy which gives equal bandwidth share to all flows has a much larger mean number of short flows in the system. Indeed, the mean number of short flows with the static policy and the weighted-PS policy is $O(\sqrt{N})$ while it is $O(N)$ with a naive processor sharing policy. The last plot in the figure shows the percentage capacity overprovisioning required for the static and weighted-PS policies to do as well as the optimal policy, and confirms that the deviation from optimality is small.

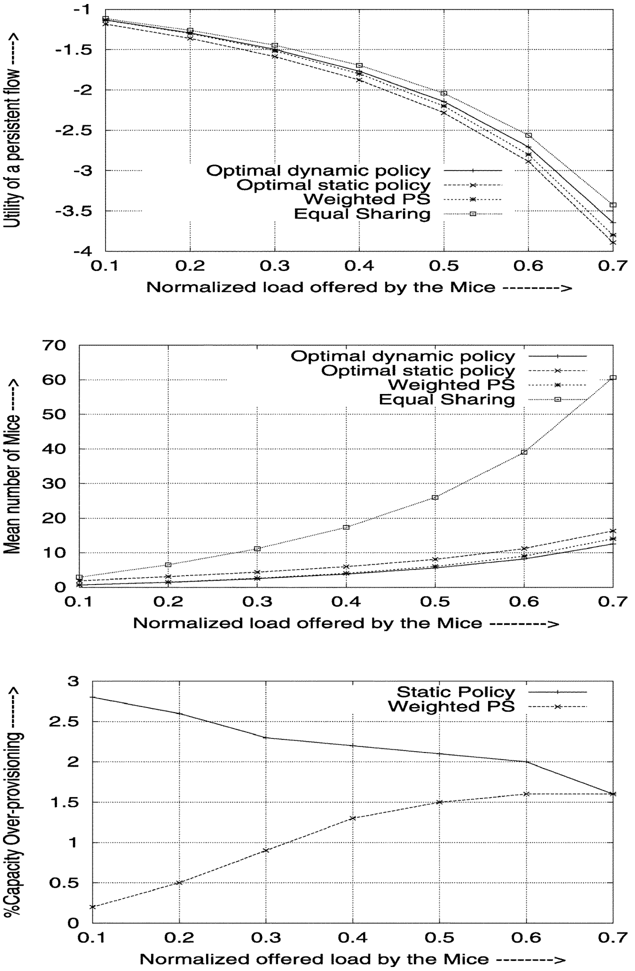

Fig. 1. The top and the middle panel show average utility of a persistent flow, and mean number of transient flows for different policies, respectively. The bottom panel shows the % overprovisioning required for static and WPS policies to outperform optimal dynamic allocation. System parameters: $c = 1$, mean file size $= 1$, $u(x) = -1/x$, $N = 25$.

V. Bandwidth Sharing Between Transient Flows

We continue to work with the optimization problem posed in Section II. There, we considered how to split capacity between persistent and transient flows but did not consider further how the capacity allocated to transient flows should be shared between them. We now consider how capacity should be shared between file transfers when the sizes of the files being transferred might vary over several orders of magnitude. The objective is to minimize the number of file transfers in progress (equivalently, the mean holding cost or mean sojourn time). If file sizes are exponentially distributed and the allocation decision has to be made without knowing the sizes of all file transfers in progress, then it not matter how this allocation is made; any allocation that does not leave capacity idle achieves the same mean number in system. If file sizes are not exponentially distributed, then this is no longer true; for example, if file sizes are heavy-tailed, the first-come-first-served policy performs worse than processor-sharing. Finally, if the amount remaining to be transferred is known, then a simple interchange argument shows that the optimal policy is to give priority to the file with shortest remaining processing time (SRPT). This policy has been proposed in the context of Web servers [2], [10]. However, it is not suited to our problem for a couple of reasons. First, it needs a centralized controller to assign priority (or a distributed leader election protocol, which imposes a high overhead). Second, while the concept is clear for a single bottleneck link or resource, it does not generalize easily to multiple bottlenecks. This motivates us to consider a generalization of the weighted PS policy introduced in the previous section, adapting it to provide a bias toward the file-transfers with small residual file-sizes.

We now consider a weighted processor sharing policy where each transient flow chooses its own weight based on its residual file size. Suppose the weights are chosen according to

$$w_x = w_{\min} + (w_{\max} - w_{\min}) \exp(-ax) \tag{19}$$

where w_x denotes the weight assigned to a flow with residual file size x, and $w_{\min}$, $w_{\max}$ and a are system parameters. The weight assigned to each persistent flow is 1. The link capacity N is shared between flows in proportion to their weights, i.e., flow i receives capacity $w_i N/W$, where W denotes the sum of w_i over all flows in the system, both persistent and transient. In practice, (19) would be implemented with the weights chosen at closely spaced discrete epochs, but we assume an idealized continuous time implementation for the purpose of analysis.

We shall assume that W is constant over time. Such an assumption is plausible in a large system operating in a steady-state regime. Indeed, we saw in Section IV-B that, when the weights of all short flows are equal to $\sqrt{N}$ irrespective of their residual file sizes, the distribution of the number of short flows in the system, $\pi(n)$, is given by (14). Using Stirling's approximation for the Gamma function in (14), it can be further shown that $\pi(n)$ is approximately Gaussian with mean $\sqrt{N}\lambda/(1-\lambda)$ and variance $\sqrt{N}\lambda/(1-\lambda)^2$. Each of the N persistent flows has weight 1. Thus, in this system, the aggregate weight W is approximately Gaussian with mean $N/(1-\lambda)$ and variance $N^{3/2}\lambda/(1-\lambda)^2$. For large N, the standard deviation, being $O(N^{3/4})$, is much smaller than the mean, which implies that W is approximately constant. A precise statement is that the random variable W/N converges in probability to the constant, $1/(1-\lambda)$.

Now note that if the total weight W is assumed to be a constant, then the total weight contributed by short flows evolves as a Poisson shot noise. A Poisson shot noise is the response of a linear system to a train of impulses arriving at the points of a Poisson process [5]. In our case, consider the arrivals with file-sizes in the interval $[x, x+dx)$ which arrive at a rate $N\lambda dF(x)$. The evolution of weight of any arrival with initial file-size x is precisely the impulse response associated with file sizes in the interval $[x, x+dx)$. Thus, if we can characterize the evolution of weight for any given file size (which is a deterministic evolution owing to the constant W assumption), then, summing up the weights due to all possible arrivals and all possible file-sizes, and the total weight due to the persistent flows, gives the total weight at any time. This should be equal to the total weight W which gives a fixed point equation for W. From this, one can compute the various quantities of interest. The detailed analysis is provided in Appendix C. We simply state the main results under a constant W assumption in the following.

A. Approximate Analysis of the Scheme

Our goal is to derive expressions for the sojourn time of a flow and the mean number of short flows in the system. The simplifying assumption behind the results of this subsection is the following.

Assumption B: The total weight W contributed by the transient flows and the persistent flows remains constant over time.

We remind the reader that there are N persistent flows, the capacity of the system is N, the mean file size of the transient flows is 1, and the arrival rate of transient flows is λ.

Let x denote the initial size of a file, and $x(t)$ its residual size t time units after its arrival into the system, where t is smaller than its sojourn time in the system. We have

$$x(0) = x, \qquad \frac{d}{dt}x(t) = -\frac{N w_{x(t)}}{W} \tag{20}$$

where $w_{x(t)}$ is specified in terms of $x(t)$ via (19). Let

$$T(x) = \inf\{t > 0 : x(t) = 0, x(0) = x\}$$

denote the sojourn time of the file with initial size x. We want to derive expressions for $T(x)$ and for the mean number of transient flows in the system. To do that, we need an expression for W.

Proposition 5.1: Under Assumption B, the total weight W, we have

$$W = \frac{N}{1-\lambda}.$$

The proof is in Appendix C. The idea of the proof is as follows. We find an expression for $w(x,t)$, the weight contributed by a short flow with initial size x after spending t units of time in the system. This expression depends on W. We then obtain the total weight in steady state due to all arrivals by integrating this expression with respect to the file size distribution. Adding this to the total weight of persistent flows should yield W. This gives a fixed point equation for W, which we solve.

We now use Proposition 5.1 to compute $T(x)$ and the mean number of transient flows in the system.

Proposition 5.2: Let $\gamma = w_{\max}/w_{\min}$. We have the following under Assumption B:

1) The sojourn time of a file with initial size x, $T(x)$, is given by

$$T(x) = \frac{1}{a w_{\min}(1-\lambda)} \log\left[1 + \frac{e^{ax}-1}{\gamma}\right]. \tag{21}$$

2) The mean number of transient flows in the system, $E[n]$, is given by

$$E[n] = \frac{N\lambda}{a w_{\min}(1-\lambda)} E\left[\log\left[\frac{\gamma - 1 + e^{aX}}{\gamma}\right]\right] \tag{22}$$

where X is the random variable from which the file-sizes of the transient flows are sampled. □

The proof is in Appendix C.

The (unweighted) processor sharing policy is recovered in the limit $a \to 0$, in which case $T(x) = x/((1-\lambda)w_{\max})$. The sojourn time of a file is thus proportional to its size, which is desirable in terms of fairness but has the disadvantage that small files see poor performance.

Having quantified the parameters of interest, we next show that the proposed dynamic weighted processor sharing scheme has, on the one hand, the desirable near optimality described in Section IV-B, and on the other hand, also provides a bias toward short flows by giving them larger throughput.

B. Near Optimality of the Scheme

Observe that the bandwidth share of the persistent flows is roughly N/W which is $(1-\lambda)$. Thus the utility of the persistent flows is roughly $Nu(1-\lambda)$ in the regime we are considering. Using the same argument as in Section IV-B, if we show that the mean number of short flows is roughly $O(\sqrt{N})$, then the minimum bandwidth $N\hat{c}$ required by an optimal policy to achieve the same cost as achieved by the dynamic weighted processor sharing policy will be given by $\hat{c} = 1 - O(1/\sqrt{N})$. This can be achieved in two ways.

If we set $w_{\min} = k_1\sqrt{N}$ and $w_{\max} = \gamma k_1 \sqrt{N}$, it can be seen from the expression of $E[n]$ that $E[n] = O(\sqrt{N})$, and thus

$$E\left[Nu\left(x(n)\right) - n\right] \approx Nu(1-\lambda) - O(\sqrt{N}).$$

Alternatively, suppose we set $w_{\min} = 1$ and $\gamma = k\sqrt{N}$. This has the interesting interpretation that an infinitely large transient flow is not distinguished from a persistent flow and thus gets the same weight as a persistent flow. So far we have not assumed anything about the file size distribution other than its finite mean. To show that the above choice of weights yields near optimality, we further assume that the file size distribution has a finite moment generating function (this is true if the file sizes are bounded). To see that $E[n] = O(\sqrt{N})$, note that we have

$$\begin{aligned}\lim_{N\to\infty}\frac{E[n]}{\sqrt{N}} &= \lim_{N\to\infty}\frac{\sqrt{N}\lambda}{aw_{\min}(1-\lambda)}E\left[\log\left[\frac{k\sqrt{N}-1+e^{aX}}{k\sqrt{N}}\right]\right]\\ &= \frac{\lambda}{a(1-\lambda)}E\left[\lim_{N\to\infty}\sqrt{N}\log\left[1+\frac{e^{aX}-1}{k\sqrt{N}}\right]\right]\\ &= \frac{\lambda}{ak(1-\lambda)}E\left[e^{aX}-1\right] < \infty\end{aligned}$$

where the exchange of limit and expectation can be justified using the monotone convergence theorem since $y\log(1+(1/y))$ is increasing in y. We clearly have $E[n] = O(\sqrt{N})$ in this case. Thus we have shown that, if we simply choose $\gamma = k\sqrt{N}$ and $w_{\min} = 1$ ($w_{\max} = k\sqrt{N}$), we can still have near optimality of the proposed dynamic processor sharing, in the sense that the additional capacity per persistent flow required in order to do as well as the optimal allocation is $O(1/\sqrt{N})$.

C. Bias Toward Short Flows

We next show that the proposed scheme is indeed biased toward short files. In order to quantify the extent to which it favors short flows, we compute the ratio of sojourn times for two different files, of sizes f_1 and f_2. With plain sharing, this ratio is $T(f_1)/T(f_2) = f_1/f_2$. Denoting the ratio $w_{\max}/w_{\min}$ by γ, we obtain for the scheme proposed above that

$$\frac{T(f_1)}{T(f_2)} = \frac{\log\left[1+(e^{af_1}-1/\gamma)\right]}{\log\left[1+(e^{af_2}-1/\gamma)\right]}. \tag{23}$$

We observe that if f_1 and f_2 are both large relative to $1/a$ and if, moreover, e^{af_i}/γ is much bigger than 1 for i =1, 2, then $T(f_1)/T(f_2) \approx f_1/f_2$. In other words, the *throughput*, defined as the ratio of file size to sojourn time, is roughly constant for large files, meaning that the scheme approximates processor sharing at large file sizes. Likewise, if f_1 and f_2 are both small relative to $1/a$, then again $T(f_1)/T(f_2) \approx f_1/f_2$. Finally, suppose f_1 is large and f_2 is small relative to $1/a$. Then, by (23)

$$\frac{T(f_1)}{T(f_2)} \approx \frac{af_1 - \log\gamma}{\frac{af_2}{\gamma}} \approx \frac{\gamma f_1}{f_2} = \frac{w_{\max}}{w_{\min}}\frac{f_1}{f_2}.$$

In other words, the small files get a throughput approximately γ times greater, or stretch (defined as $T(f)/f$) $1/\gamma = w_{\min}/w_{\max}$ smaller than a large file. Loosely speaking, files much smaller than $1/a$ are "mice," files much larger than $1/a$ are "elephants," all mice are treated roughly equally, as are all elephants, but mice are favored over elephants. Note that this is achieved without explicitly splitting files into classes, but simply by having them choose individual weights based on their residual file sizes.

The degree to which mice are favored is determined by the ratio $\gamma = w_{\max}/w_{\min}$. It needs to be kept in mind that this is under the assumption that W is constant, which is not valid if there are no persistent flows. A model with no persistent flows and with an SRPT service discipline has been studied in [2], where it is shown that the stretch of long flows remains bounded. The intuition is that there will be epochs when the long flow is competing with very few or no short flows, at which times it is not handicapped by its small weight. A similar intuition applies to our model.

D. Simulation Results With the Proposed Scheme

We now present simulation results to show that the proposed dynamic weighted processor sharing scheme can improve the throughput of the transient flows without penalizing the persistent flows when compared to systems using weighted processor sharing with static weights.

We simulate a system with capacity $C = 100$ carrying $N = 25$ persistent flows, each of which has weight 1 and has the utility function $u(x) = -1/x$. File transfers arrive at rate λ, and file sizes have the Pareto distribution, $P(\text{file}-\text{size} > x) = 1/(1+(x/f))^2$, $x \geq 0$, with mean file size $f = 100$. We take $a = 1/f$, $w_{\max} = 50$ and $w_{\min} = 10$. Performance measures for processor sharing with the scheme described above, and processor sharing with constant (file-size independent) weights $\overline{w}$ for two different weights, 10 and 50, are shown in Fig. 2. The top panel shows the utility received by the persistent flows under each policy, while the bottom panel shows the throughput of transient flows. The simulation results are based on 12 000 events (file arrivals) with a burn-in period of 500 time units for the system to reach stationarity, and, are averaged over multiple

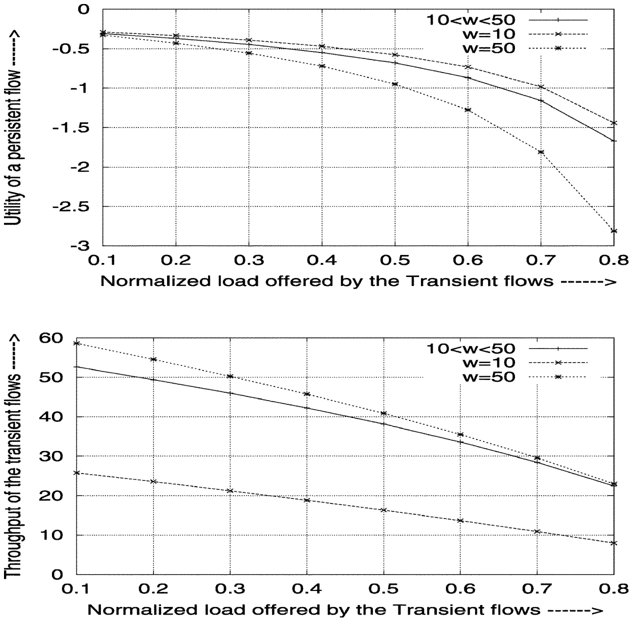

Fig. 2. Comparison of average utility of a persistent flow (top), and average throughput of transient flows (bottom) with three schemes: in one the weights of the short flows are varied between 10 $(w_{\min})$ and 50 $(w_{\max})$ according to the scheme discussed in this section, and the other two are with fixed weights $(\overline{w})$ as 10 and 50, respectively. The parameters are capacity $C = 100$, mean file size $f = 100$, $u(x) = -1/x$, and number of persistent flows $N = 25$. The arrival rate λ is varied along the x-axis.

runs. Clearly, when $\overline{w} = 50$, the average throughput of the transient flows goes up but at the cost of a reduced utility for the persistent flows. When $\overline{w} = 10$, the persistent flows perform better but the average throughput of the short flows decreases a lot. However, by using the processor sharing described in this section, wherein the weights of transient flows are varied dynamically, the transient flows can achieve a large throughput without starving the persistent flow much.

E. A Distributed Implementation

We assumed above that bandwidth is shared exactly according to the proposed weighted scheme. We now propose a mechanism to approximate this by a suitable modification to the TCP flow control protocol. Recall that TCP is a window flow control mechanism which operates as follows. In the slow start phase, each received acknowledgment increases the window size by one packet, while in the congestion avoidance phase W acknowledgments increase the window size from W to $W+1$. Further each lost packet or negative acknowledgment reduces the window size by half. We propose the following modification, which we shall refer to below as Dynamic-TCP.

Slow start: If the current window size is W and the residual file size is f_r, then a successful acknowledgment updates the window size as

$$W \leftarrow W + \left(1 + \left(\frac{w_{\max}}{w_{\min}} - 1\right) \exp\left(-af_r\right)\right).$$

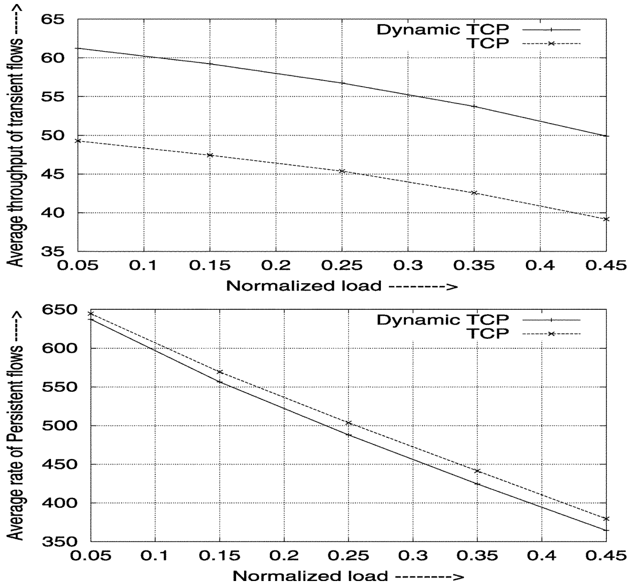

Fig. 3. Plots showing comparison between Dynamic-TCP and Standard-TCP for average session-throughput seen by the transient flows and the time-average rate of all the persistent flows. The plots are for mean file size $= 40$, number of persistent flows $= 10$. Arrival rate is varied along the x-axis.

Congestion avoidance: If the current window size is W and the residual file size is f_r, then a successful acknowledgment updates the window size as

$$W \leftarrow W + \frac{\left(1 + \left(\frac{w_{\max}}{w_{\min}} - 1\right) \exp(-af_r)\right)}{\lfloor W \rfloor}.$$

Window decrease: If the current window size is W and the residual file size is f_r, then a lost or marked packet (in an ECN compatible system) reduces the window size as

$$W \leftarrow W \times \left(\frac{1}{2} + \frac{1}{2}\exp(-af_r)\right)$$

The main idea is to allow small files to increase their window size more rapidly and decrease it more slowly. In the above, $w_{\max}$, $w_{\min}$, a are system parameters. We now present a very preliminary evaluation of this algorithm.

We consider a single link of capacity 800 packets per second carrying 10 persistent flows. Transient flows arrive as a Poisson process and have Pareto file sizes with ccdf $1/(1 + x/f)^2$, where the mean file size is $f = 40$. The link buffer employs Random Early Marking [1]: when the number of packets in the link buffer is q, an arriving packet is marked with probability $(1 - \exp(-\gamma q))$. We take $\gamma = 0.005$. Each persistent flow has a round trip delay of 80 ms and adapts its rate using a standard ECN compatible TCP. The round trip delays of the transient flows are uniformly distributed in the interval [60 200] ms. We compare the scenarios where the transient flows use Dynamic-TCP and where they use standard ECN compatible TCP. For Dynamic-TCP, we use $w_{\max}/w_{\min} = 5$ and $a = 0.02$; the latter choice corresponds to assuming that files smaller than 50 packets are "mice." Fig. 3 shows that Dynamic-TCP achieves significantly better mean throughput for transient flows at the price of a small reduction in throughput for the persistent flows. The results presented are very preliminary and a much more thorough investigation is needed to validate these findings. It also remains to address the question of what incentive users have to adopt the proposed weight parameters. We have not considered this, but assumed that all users comply. The purpose here has simply been to show that the proposed scheme can, in principle, be implemented at end systems. A great deal of further work is required on the practicalities.

VI. Concluding Remarks

We considered the problem of optimal bandwidth allocation in a system consisting of both persistent and transient flows. Treating all transient flows as identical, we first described simple algorithms that achieve a nearly optimal partitioning of the available bandwidth between the persistent and transient sources. We then studied the problem of how to share the bandwidth allocated to transient flows among file transfers of different sizes and described a distributed scheme that is biased in favor of "mice" over "elephants."

The optimal choice of parameters for the policies described in Section IV requires knowledge of system parameters, which is often unrealistic. We believe that comparable performance can be achieved by adaptive policies that tune their parameters based on measurements, but this is a topic for future research. Extending our single-bottleneck analysis to networks is also an open problem. Finally, a detailed implementation of the proposed algorithms at end-systems and their evaluation is a topic for future work.

Appendix

A. Proof of Lemma 4.2

Lemma: Let $a > 0$ be arbitrary and let $w = a\sqrt{N}$, so that $K = \sqrt{N}/a$. Let n^* be the integer part of $K\lambda/(1-\lambda)$. (The dependence of w, K and n^* on N has not been made explicit in the notation). For any $\delta > 0$, there are constants $c_1, c_2 > 0$, not depending on N, such that

$$\sum_{m=(1+\delta)n^*}^{\infty} \pi(n) \le c_1 e^{-c_2\sqrt{N}} \qquad \sum_{m=0}^{(1-\delta)n^*} \pi(n) \le c_1 e^{-c_2\sqrt{N}}$$

for all N sufficiently large.

Proof: For ease of computation, we will treat $K\lambda/(1-\lambda)$ as an integer so that $n^* = K\lambda/(1-\lambda)$. The same argument goes through when $K\lambda/(1-\lambda) - 1 < n^* \le K\lambda/(1-\lambda)$.

Observe that if $i \ge \delta n^*$, then

$$\begin{aligned}\frac{\pi(n^*+i+1)}{\pi(n^*+i)} &= \left(1 + \frac{K}{n^*+i+1}\right)\lambda \\ &\le \left(1 + \frac{K}{(1+\delta)n^*+1}\right)\lambda \\ &\le \left(1 + \frac{1-\lambda}{(1+\delta)\lambda}\right)\lambda \\ &= \frac{1+\delta\lambda}{1+\delta}\end{aligned}$$

where we have substituted $n^* = K\lambda/(1-\lambda)$ to obtain the second inequality. Hence

$$\sum_{m=(1+\delta)n^*}^{\infty} \pi(n) \le \pi\left((1+\delta)n^*\right) \sum_{m=0}^{\infty} \left(\frac{1+\delta\lambda}{1+\delta}\right)^m \tag{24}$$

$$= \frac{1+\delta}{\delta(1-\lambda)} \pi\left((1+\delta)n^*\right). \tag{25}$$

We also have by (14) and Stirling's formula that

$$\pi\left((1+\delta)n^*\right) \le \left(\frac{K+(1+\delta)n^*}{K}\right)^K \times \left(\frac{K+(1+\delta)n^*}{(1+\delta)n^*}\right)^{(1+\delta)n^*} \lambda^{(1+\delta)n^*}(1-\lambda)^{K+1}.$$

Taking logarithms, we get

$$\log \pi\left((1+\delta)n^*\right) \le -\left(K + (1+\delta)n^*\right) \times H\left(\frac{K}{K+(1+\delta)n^*}; 1-\lambda\right)$$

where $H(q;p) := q\log(q/p) + (1-q)\log((1-q)/(1-p))$ denotes the relative entropy or Kullback-Leibler divergence of the Bernoulli distribution with parameter q relative to the Bernoulli distribution with parameter p. Substituting for n^* yields

$$\log \pi\left((1+\delta)n^*\right) \le -\frac{K(1+\delta\lambda)}{1-\lambda} H\left(\frac{1-\lambda}{1+\delta\lambda}; 1-\lambda\right).$$

Now the relative entropy $H(q;p)$ is strictly positive if $q \ne p$, and $K = \sqrt{N}/a$, so we have

$$\pi\left((1+\delta)n^*\right) \le e^{-c_2\sqrt{N}}$$

for some constant $c_2 > 0$. Combining this with (24) yields the first claim of the lemma. The proof of the second claim is similar and is omitted. ■

B. Proof of Lemma 4.3

Lemma: Let $a > 0$ be arbitrary, and let $w = a\sqrt{N}$ be the relative weight assigned to each short flow. If $u(\cdot)$ satisfies Assumption A, then

$$E_{\pi_W(w)}\left[Nu\left(x(n)\right) - n\right] = Nu(1-\lambda) - O(\sqrt{N}).$$

Proof: We shall denote by E_π expectations with respect to the invariant distribution π induced by the policy $\pi_W(w)$. We can decompose $E_\pi[u(x(n))]$ as

$$E_\pi\left[u\left(x(n)\right)\right] = \sum_{k=0}^{\beta K-1} \pi(k)u\left(x(k)\right) + \sum_{k=\beta K}^{\infty} \pi(k)u\left(x(k)\right) \tag{26}$$

where $\beta > 0$ is an arbitrary constant, and $x(k) = K/(K+k)$ is the per-flow bandwidth allocated to each persistent flow when there are k transient flows in the system. We want to show that $E_\pi[u(x(n))]$ is not much smaller than $u(1-\lambda)$, i.e., we seek a lower bound on $E_\pi[u(x(n))]$. Now, if u is nonnegative, it is clear that ignoring the second term in the sum above gives a lower bound. Hence, we shall only consider the case where $u(x)$

is negative for small enough x, and show in this case that the second term above is negligible.

Observe that

$$\frac{\pi(k+1)}{\pi(k)} = \frac{K+k+1}{K+1}\lambda \le \frac{(1+\beta)\lambda}{\beta} \quad \forall\, k \ge \beta K.$$

Since $\lambda < 1$, this ratio is strictly smaller than 1 for large enough β. We shall denote its logarithm by $-\gamma$, so that $\gamma > 0$. Proceeding as in the proof of Lemma 4.2, we have

$$\pi(\beta K + m) \le e^{-\gamma m}\pi(\beta K) \le e^{-c\sqrt{N}} e^{-\gamma m} \qquad (27)$$

where c is a positive constant and $\gamma = -\log[(1+\beta)\lambda)/\beta] > 0$.

Next, observe that $x(k) \to 0$ as $k \to \infty$ and so, for β sufficiently large, $u(x(k)) < 0$ whenever $k \ge \beta K$. For sequences a_N, b_N, we shall write $a_N \asymp b_N$ to mean that the ratio a_N/b_N is bounded away from zero and infinity, uniformly in N. Now, by (27)

$$\begin{aligned}\sum_{k=\beta K}^{\infty} \pi(k) u\left(x(k)\right) &\ge e^{-c\sqrt{N}} \sum_{m=0}^{\infty} e^{-\gamma m} u\left(\frac{K}{K+\beta K+m}\right)\\ &\asymp e^{-c\sqrt{N}} \int_{x=0}^{\infty} e^{-\gamma x} u\left(\frac{K}{K+\beta K+x}\right) dx.\end{aligned}$$

Noting that $K = N/w = \sqrt{N}/a$, and making the change of variables $y = x/K$, we get

$$\begin{aligned}&\sum_{k=\beta K}^{\infty} \pi(k) u\left(x(k)\right)\\ &\quad\ge c_1\sqrt{N} e^{-c\sqrt{N}} \int_{0}^{\infty} e^{-\frac{\gamma\sqrt{N}}{a}y} u\left(\frac{1}{1+\beta+y}\right) dy\\ &\quad\ge c_1\sqrt{N} e^{-c\sqrt{N}} \left[\int_{0}^{1+\beta} e^{-\frac{\gamma\sqrt{N}}{a}y} u\left(\frac{1}{1+\beta+y}\right) dy\right.\\ &\qquad\left. + \int_{1+\beta}^{\infty} e^{-\frac{\gamma\sqrt{N}}{a}y} u\left(\frac{1}{2y}\right) dy\right].\end{aligned}$$

Since $u(1/(1+\beta+y))$ is bounded on the compact interval $[0, 1+\beta]$, the first integral in the last line above goes to zero as $N \to \infty$ by dominated convergence. For large enough N, the second integrand is bounded above by $e^{-\eta y}u(1/2y)$, which is integrable over $[1+\beta, \infty)$ by Assumption A. Hence, the second integral also goes to zero by dominated convergence.

We have thus shown that $\sum_{k=\beta K}^{\infty} \pi(k)u(x(k))$ is either positive or goes to zero faster than $e^{-c\sqrt{N}}$. In either case, the second term in the sum in (26) can be ignored in deriving a lower bound on $E_\pi[u(x(n))]$. It remains to evaluate $E_\pi[u(x(n))1_{(n<\beta K)}]$.

Expanding $u(x)$ in a Taylor series around $1-\lambda$, we get

$$u(x) = u(1-\lambda) + (x-1+\lambda)u'(1-\lambda) + \frac{(x-1+\lambda)^2}{2} u''\left(y(x)\right) \quad (28)$$

for some $y(x)$ lying between x and $1-\lambda$. Now

$$\begin{aligned}E_\pi\left[u(1-\lambda)1_{(n<\beta K)}\right] &= u(1-\lambda)\left[1 - \sum_{k=\beta K}^{\infty} \pi(k)\right]\\ &= u(1-\lambda)(1 - O(e^{-c\sqrt{N}}) \quad (29)\end{aligned}$$

for some constant $c > 0$, by Lemma (4.2). We also have

$$\begin{aligned}E_\pi\left[x(n)\right] &= E_\pi\left[\frac{K}{K+n}\right] = \sum_{k=0}^{\infty} \pi(k) \frac{K}{K+k} \qquad (30)\\ &= \sum_{k=0}^{\infty} \frac{\Gamma(K+k+1)}{\Gamma(K+1)\Gamma(k+1)} \frac{K}{K+k} \lambda^k (1-\lambda)^{K+1}\\ &= (1-\lambda)^{K+1} \sum_{k=0}^{\infty} \frac{\Gamma(K+k)}{\Gamma(K)\Gamma(k+1)} \lambda^k = 1-\lambda. \qquad (31)\end{aligned}$$

We have used the relation $\Gamma(a+1) = a\Gamma(a)$ to obtain the third equality, and (13) to obtain the last equality. Therefore

$$\begin{aligned}&E_\pi\left[\left(x(n) - 1 + \lambda\right)1_{(n<\beta K)}\right]\\ &\quad= \sum_{k=\beta K}^{\infty} \pi(k)\left(1 - \lambda - \frac{K}{K+k}\right)\\ &\quad= O(e^{-c\sqrt{N}}) \qquad (32)\end{aligned}$$

by Lemma (4.2), since $K/(K+k) \le 1$ for all k.

Now, if $k < \beta K$, then $x(k) \ge 1/(1+\beta)$. The continuous function $u''(\cdot)$ is bounded below on the compact set $[1/(1+\beta), 1]$, say by $-\kappa$ (u'' is negative since u is concave). Thus

$$\begin{aligned}&E_\pi\left[\left(x(n) - 1 + \lambda\right)^2 u''\left(y\left(x(n)\right) 1_{(n<\beta K)}\right]\\ &\qquad\ge -\kappa E_\pi\left[\left(\frac{K}{K+n} - 1 + \lambda\right)^2 1_{(n<\beta K)}\right].\end{aligned}$$

Now, analogous to (31), we have

$$E_\pi\left[\frac{K(K-1)}{(K+n)(K+n-1)}\right] = (1-\lambda)^2.$$

Moreover

$$\begin{aligned}&\frac{K^2}{(K+k)^2} - \frac{K(K-1)}{(K+k)(K+k-1)}\\ &\quad= \frac{Kk}{(K+k)^2(K+k-1)}\\ &\quad\le \frac{1}{K^2} = \frac{a^2}{N} \quad \forall\, k\end{aligned}$$

so $E_\pi[(K/(K+n))^2] = (1-\lambda)^2 + O(1/N)$. Combining this with (31) yields

$$E_\pi\left[\left(\frac{K}{K+n} - (1-\lambda)\right)^2\right] = O\left(\frac{1}{N}\right).$$

Since $x(n) = K/(K+n)$ is bounded by 1, it now follows from Lemma 4.2 that

$$\begin{aligned}&E_\pi\left[\left(x(n) - 1 + \lambda\right)^2 u''\left(y\left(x(n)\right) 1_{(n<\beta K)}\right]\\ &\quad\ge -\kappa\left[O(\frac{1}{N} + O(e^{-c\sqrt{N}})\right]\\ &\quad= O\left(\frac{1}{N}\right). \qquad (33)\end{aligned}$$

Now it follows from (28), (29), (32), and (33) that

$$E_\pi\left[u\left(x(n)\right)1_{(n<\beta K}\right] = u(1-\lambda) - O\left(\frac{1}{N}\right)$$

and so, by (26)

$$E_\pi\left[Nu\left(x(n)\right)\right] \ge Nu(1-\lambda) - O(1).$$

We also have by (16) that $E_\pi[n] = O(K) = O(\sqrt{N})$. This establishes the claim of the lemma. □ ■

C. Derivations Pertaining to Section V-A

Let x denote the initial size of a file, and $x(t)$ its residual size t time units after its arrival into the system, where t is smaller than its sojourn time in the system. We have

$$x(0) = x, \quad \frac{d}{dt}x(t) = -\frac{Nw_{x(t)}}{W} \tag{34}$$

where $w_{x(t)}$ is specified in terms of $x(t)$ via (19). Using (19), we can rewrite the above as

$$\frac{d}{dt}w_{x(t)} = \frac{Na}{W}\left(w_{x(t)} - w_{\min}\right) w_{x(t)}.$$

The solution of this differential equation is

$$1 - \frac{w_{\min}}{w_{x(t)}} = \left[1 - \frac{w_{\min}}{w_x}\right] \exp\left(\frac{Naw_{\min}}{W}t\right). \tag{35}$$

Let $T(x) = \inf\{t > 0 : x(t) = 0, x(0) = x\}$ denote the sojourn time of the file. Thus, $T(x) = \inf\{t > 0 : w_{x(t)} = w_{\max}, x(0) = x\}$ and it satisfies

$$1 - \frac{w_{\min}}{w_{\max}} = \left[1 - \frac{w_{\min}}{w_x}\right] \exp\left(\frac{Naw_{\min}}{W}T(x)\right)$$

which yields

$$T(x) = \frac{W}{Naw_{\min}} \log\left[1 + \frac{w_{\min}}{w_{\max}}(e^{ax} - 1)\right]. \tag{36}$$

Recall that, by the sojourn time formula given by (36), a file transfer of size x originated at time $-t$ is there in the system at time zero if $T(x) > t$ which upon simplification yields $x > g(t)$, where

$$g(t) = \frac{1}{a}\log\left(1 + \frac{w_{\max}}{w_{\min}}\left[\exp\left(\frac{Naw_{\min}}{W}t\right) - 1\right]\right). \tag{37}$$

Let $\overline{F}$ denote the complementary file size distribution, i.e., $\overline{F}(x) = 1 - F(x)$ is the probability that the size of a randomly chosen file is bigger than x. Then, the mean number of file transfers in progress at any time is given by

$$E[n] = N\lambda \int_0^\infty \int_{g(t)}^\infty dF(x)dt$$

We are now in a position to prove Proposition 5.1 and Proposition 5.2. We restate them below for convenience.

Proposition: Under Assumption B, the total weight W, which is assumed to be a constant must satisfy

$$W = \frac{N}{1-\lambda}.$$

Proof: Let $w(x,t)$ be the weight contributed by a file initiated at time $-t$ of initial size x. Clearly, the total weight contributed by all the flows at time 0 is the sum of the weight contributed by the persistent flows and the weight contributed by the transient flows at time 0. We thus have

$$W = N + N\lambda \int_0^\infty \int_{g(t)}^\infty w(x,t)dF(x)dt \tag{38}$$

which provides a fixed point equation for W (since $g(t)$ depends on W). To prove Proposition 5.1, it is thus enough to show that the fixed point (38) has a unique solution given by

$$W = \frac{N}{1-\lambda}.$$

First note that the condition (37), can be equivalently written as

$$t < \frac{W}{Naw_{\min}} \log\left[\frac{\gamma - 1 + e^{ax}}{\gamma}\right]$$

where $\gamma = w_{\max}/w_{\min}$. Thus, we can change the order of integration in (38) and write the weight contributed by the short flows as

$$N\lambda \int_{x=0}^{\infty} \int_{t=0}^{\frac{W}{Naw_{\min}t}\log\left[\frac{\gamma-1+e^{ax}}{\gamma}\right]} \frac{w_{\min}(\gamma-1+e^{ax})}{\gamma-1+e^{ax}-(\gamma-1)e^{\frac{Naw_{\min}}{W}t}}dt \times dF(x).$$

By observing that

$$\int \frac{\alpha}{\alpha - \beta e^{\theta t}}dt = t - \frac{\log(\alpha - \beta e^{\theta t})}{\theta}$$

we can write the fixed point equation as

$$\begin{aligned} W &= N + N\lambda \\ &\quad \times \int_{x=0}^{\infty} \int_{t=0}^{\frac{W}{Naw_{\min}}\log\left[\frac{\gamma-1+e^{ax}}{\gamma}\right]} \frac{w_{\min}(\gamma-1+e^{ax})}{\gamma-1+e^{ax}-(\gamma-1)e^{\frac{Naw_{\min}}{W}t}} \\ &\qquad \times dt\, dF(x) \\ &= N + N\lambda \\ &\quad \times \int_{x=0}^{\infty} \frac{W}{Na}\left(\log\left[\frac{\gamma - 1 + e^{ax}}{\gamma}\right]\right. \\ &\qquad \left. + ax - \log\left[\frac{\gamma - 1 + e^{ax}}{\gamma}\right]\right) dF(x) \\ &= N + W\lambda EX \end{aligned}$$

where EX is the expected file size. But since we assumed $EX = 1$, we have $W = N/(1-\lambda)$ ■

Proposition: Let $\gamma = w_{\max}/w_{\min}$. We have the following under Assumption B:

1) The sojourn time of a file with initial size x, $T(x)$, is given by

$$T(x) = \frac{1}{aw_{\min}(1-\lambda)} \log\left[1 + \frac{e^{ax} - 1}{\gamma}\right]. \tag{39}$$

2) The mean number of transient flows in the system, $E[n]$, is given by

$$E[n] = \frac{N\lambda}{aw_{\min}(1-\lambda)} E\left[\log\left[\frac{\gamma - 1 + e^{aX}}{\gamma}\right]\right] \tag{40}$$

where X is the random variable from which the file-sizes of the transient flows are sampled. □

Proof: The expression for $T(x)$ follows immediately from (36) after plugging in the expression for W. Thus, the mean sojourn time is given by $E[T(X)]$, where X is the random varibale from which the file sizes are sampled. The mean number of transient flows can be now obtained using Little's formula as $E[n] = N\lambda E[T(X)]$.

As long as the $EX < \infty$, it is easy to see that $E[T(X)]$ exists since $\log[(\gamma - 1 + e^{ax})/\gamma] < ax$ for $\gamma > 1$. ■

References

[1] S. Athuraliya, D. E. Lapsley, and S. H. Low, "Random early marking for Internet congestion control," in *Proc. IEEE GLOBECOM*, Rio de Janeiro, Brazil, 1999, pp. 1747–1752.

[2] N. Bansal and M. Harchol-Balter, "Analysis of SRPT scheduling: investigating unfairness," in *Proc. ACM Sigmetrics*, 2001, pp. 279–290.

[3] S. Blake, M. Carlson, E. Davies, B. Ohlman, D. Verma, Z. Wang, and W. Weiss, "A framework for differentiated services," presented at the 42nd Internet Engineering Task Force (IETF) Meeting, Chicago, IL, 1998.

[4] M. E. Crovella and A. Bestavros, "Self-similarity in World Wide Web traffic: evidence and possible causes," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 835–846, Dec. 1997.

[5] D. J. Daley and D. Vere-Jones, *An Introduction to the Theory of Point Processes*. New York: Springer-Verlag, 1988.

[6] S. Deb, A. J. Ganesh, and P. B. Key, "Resource allocation with persistent and transient flows," presented at the Networking 2002 Conf., Pisa, Italy, 2002.

[7] ——, "Resource allocation with persistent and transient flows," Microsoft Research, Tech. Rep. MSR-TR-2001-114, 2001.

[8] C. Gromoll, "Diffusion approximation for a processor sharing queue in heavy traffic," *Ann. Appl. Prob.*, vol. 14, pp. 555–611, 2004.

[9] R. J. Gibbens and F. P. Kelly, "Resource pricing and the evolution of congestion control," *Automatica*, vol. 35, pp. 1969–1985, 1999.

[10] M. Harchol-Balter, B. Schoeder, N. Bansal, and M. Agarwal, "Size-based scheduling to improve web performance," *ACM Trans. Comput. Syst.*, vol. 21, no. 2, pp. 207–233, 2003.

[11] P. Hurley, J. Y. le Boudec, and P. Thiran, "A note on the fairness of additive increase and multiplicative decrease," in *Proc. 16th Int. Teletraffic Congr. (ITC-16)*, Jun. 1999, pp. 467–478.

[12] F. P. Kelly, "Mathematical modeling of the Internet," in *Mathematics Unlimited—2001 and Beyond*, B. Engquist and W. Schmid, Eds. Berlin, Germany: Springer-Verlag, 2001.

[13] ——, *Reversibility and Stochastic Networks*. New York: Wiley, 1979.

[14] L. Kleinroch, *Queueing Systems, Volume 2: Computer Applications*. New York: Wiley-Interscience, 1975.

[15] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, pp. 237–252, 1998.

[16] P. Key and L. Massoulié, "User policies in a network implementing congestion pricing," in *Proc. Workshop on Internet Service Quality Economics (ISQE)*, Cambridge, MA, Dec. 1999, pp. 189–197.

[17] S. Kunniyur and R. Srikant, "End-to-end congestion control: utility functions, random losses and ECN marks," in *Proc. IEEE INFOCOM*, Tel Aviv, Israel, Mar. 2000, pp. 1323–1332.

[18] R. J. La and V. Anantharam, "Charge-sensitive TCP and rate control in the Internet," in *Proc. IEEE INFOCOM*, 2000, pp. 1166–1175.

[19] Low and D. E. Lapsley, "Optimization flow control—I: basic algorithm and convergence," *IEEE/ACM Trans. Netw.*, vol. 7, no. 6, pp. 861–875, Dec. 1999.

[20] L. Massoulié and J. Roberts, "Bandwidth sharing: objectives and algorithms," *IEEE/ACM Trans. Netw.*, vol. 10, no. 3, pp. 320–328, Jun. 2002.

[21] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556–567, Oct. 2000.

[22] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A model based TCP-friendly rate control protocol," presented at the Int. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Basking Ridge, NJ, Jun. 1999.

[23] S. M. Ross, *Introduction to Stochastic Dynamic Programming*. New York: Academic, 1983.

[24] S. Shenker, "Fundamental design issues for the future Internet," *IEEE J. Select. Areas Commun.*, vol. 13, no. 7, pp. 1176–1188, Sep. 1995.

[25] H. C. Tijms, *Stochastic Models: An Algorithmic Approach*. Chichester, U.K.: Wiley, 1994.

[26] M. Vojnovic, J. Y. Le Boudec, and C. Boutremans, "Global fairness of additive-increase and multiplicative-decrease with heterogeneous round trip times," in *Proc. IEEE INFOCOM*, vol. 3, Mar. 2000, pp. 1303–1312.

Supratim Deb received the Bachelor of Engineering degree in electrical and electronics engineering from the Birla Institute of Technology and Science, Pilani, India, in 1997 and the Master of Engineering degree in telecommunication from the Indian Institute of Science, Bangalore, India in 1999. He received the Ph.D. degree in electrical engineering from the University of Illinois at Urbana-Champaign in 2003.

He worked with the IBM India Research Lab for a short span during 1999-2000. Following his Ph.D. in 2003, he worked as a post-doctoral associate at the Massacusetts Institute of Technology, Cambridge, before joning Bell-Labs Research India, Bangalore, in 2005. His research interests include Internet congestion control, resource allocation in wireless and wireline networks, network coding, and distributed systems.

Ayalvadi Ganesh graduated from the Indian Institute of Technology, Madras, in 1988. He received the M.S. and Ph.D. degrees in electrical engineering from Cornell University, Ithaca, NY, in 1991 and 1995, respectively. His Ph.D. thesis was on the use of large deviation techniques in queueing theory.

He was with Edinburgh University, Birkbeck College, London, U.K., and Hewlett-Packard's Basic Research Institute in Mathematical Sciences (BRIMS) before joining Microsoft Research in March 1999, as a Researcher in the Networking group. He was also a Fellow of King's College, Cambridge, from 2000 to 2004. His research interests are application of random graph models to peer-to-peer networks, resource allocation, control and pricing in the Internet, and large deviation theory and applications to queueing theory and statistical inference.

Peter Key received the B.A. degree in mathematics from Oxford University, Oxford, U.K., in 1978, and the M.Sc. degree in statistics from University College, London, U.K., in 1979. He was a Research Assistant in the Statistics and Computer Science Department of Royal Holloway College, London University, until 1982. He received the Ph.D. degree in 1985 with a thesis on Bayesian forecasting.

He joined BT Labs (formerly British Telecom) in 1982, working in the field of teletraffic engineering and performance evaluation, where he was involved with the development and introduction of Dynamic Alternative Routing (DAR). After leading a mathematical services group, he became involved with network reliability. In 1992, he ventured into the ATM arena, to manage an ATM performance group. He joined Microsoft Research's European Research Centre in Cambridge in 1998. He is an inventor on a number of patents concerned with dynamic routing and call admission control. In 1999, he was technical co-chair of the 16th International Teletraffic Congress. He is also a Visiting Fellow at the Statistical Laboratory, Cambridge.

Dr. Key is a Fellow of the Royal Statistical Society.