Protecting Small Flows from Large Ones for Quality of Service

R. G. Addie, Oleksiy Yevdokimov, Stephen Braithwaite and David Millsom

Abstract

A simple strategy for improving Quality of Service is to protect small flows from large ones, by finding a small number of large flows, separating them from the remaining traffic, and giving the small flow traffic priority over these large flows. This is a very powerful technique because it is known that a small proportion of all flows (e.g. 0.1%) contain a large proportion of all bytes (e.g. 20%). Many questions arise when this method of managing performance is considered. These questions are investigated by means of a mathematical model of traffic and of queueing in routers. One particular question is particularly emphasized in this paper: how should the size of flows be measured?

1 Introduction

Quality of service (QoS) in the Internet has been a topic of ongoing interest since before the Internet was even implemented. The QoS of the current Internet is remarkably good. However it is still not good enough to sustain all the services that we would like to provide via the Internet. Voice over Internet protocol (VOIP) services are a good example of this: although the Internet is used extensively to provide VOIP communication, the performance provided in this way is not always satisfactory. The technique considered in this paper addresses this problem in a simple manner, not requiring changes to Internet protocols or architecture.

It is likely that new services (video-conferencing, IP TV) which require good performance and which place greater demands for bandwidth on the Internet will become popular in the future. We therefore need a way to meet these demands efficiently, while meeting a QoS target.

Many strategies have been put forward to address the issue of QoS, notably the Integrated Services architecture for the Internet (IntServ) [4] and the Differentiated Services architecture (DiffServ) [12]. These strategies require the deployment of mechanisms in routers for resource reservation, in the case of IntServ, or to classify and police traffic, in the case of DiffServ. If DiffServ is used, traffic will be separated into different classes, and users who want better performance will be able to pay for it.

In this paper we explore a strategy – Large Flow Discrimination (LFD) – which does not require reservation, tagging of packets, or classification of traffic, or policing, because the only traffic which receives different performance does so because of its statistical characteristics (size). Instead of differentiating *for* service requests on the basis of *need*, we differentiate *against* service requests on the basis of *greed*.

Many questions arise: 1. Does this method provide sufficient Quality of Service for the small flows? 2. How many large flows need to be treated separately? 3. How frequently do new large flows arrive and depart? 4. How should we measure the size of a flow? 5. Will some small flows be mixed up with large flows and receive bad performance? 6. How bad is performance for small flows at present? 7. How sensitive are the benefits of LFD to assumptions regarding Internet traffic? 8. How quickly do we need to identify large flows in order to manage their impact on small flows? 9. What will happen to a large flow which needs high QoS; in particular, a large continuous bit-rate flow which needs low jitter? 10. Doesn't DiffServ already solve the QoS problem?

Although we give tentative answers to most of these questions in the concluding remarks, the question we focus on is *how to measure flow size*.

The remainder of this paper is as follows: in Section 2 we review what is known about Internet traffic; in Section 3 we define the LFD strategy; in Section 4 we present a new method for measuring the size of an Internet flow; in Section 5 we estimate the complexity of the proposed algorithms; in Section 6 we present an estimate of the performance that will be experienced by small and large flows in the Internet if LFD is used. In Section 7 we return to consideration of the questions which were asked, concerning the LFD method, in the previous paragraph.

2 Internet Traffic

Internet traffic has been found to be *long-range dependent* and *self-similar* [9]. A simple explanation of this which is widely accepted is that Internet traffic is made up of flows

and the byte counts of these flows have a heavy tailed distribution, such as the Pareto distribution [10, 3, 8].

It is common to assume that the arrival times of flows forms a Poisson process, i.e. a process of arrival times which is completely uncorrelated. Measurements have shown that this is not the case, however it has been shown in [7] that even if flow arrivals are correlated, this effect on overall characteristics of traffic is secondary by comparison with the heavy-tailed character of the individual flows. A Poisson arrival process of heavy-tailed flows remains, therefore, a satisfactory basic model of Internet traffic.

If d is Pareto distributed:

$$\Pr(d > x) = \begin{cases} \left(\frac{x}{\delta}\right)^{-\gamma}, & x \geq \delta, \\ 1, & \text{otherwise.} \end{cases} \tag{1}$$

A typical choice for parameters is $\gamma = 1.1$, $\delta = 1$. With these settings, for example, 50% of flows will have their size less than or equal to $0.5^{-\frac{1}{1.1}} = 2^{\frac{1}{1.1}}$.

Counting bytes instead of flows reveals a much heavier tail. Let D denote the length of the flow containing a byte chosen at random. Then

$$\begin{aligned} \Pr(D > y) &= \begin{cases} \int_\delta^y x d\left(\frac{x}{\delta}\right)^{-\gamma} / \int_\delta^\infty x d\left(\frac{x}{\delta}\right)^{-\gamma}, & y \geq \delta, \\ 1, & \text{otherwise,} \end{cases} \\ &= \begin{cases} \left(\frac{y}{\delta}\right)^{1-\gamma}, & y \geq \delta, \\ 1, & \text{otherwise.} \end{cases} \end{aligned} \tag{2}$$

As a consequence, a high proportion of bytes are transported in a very small proportion of flows. Denote by ℓ_1, the length of a flow such that 20% of all bytes are in larger flows. Then $\ell_1 = 5^{10} = 9765625$. From (1) the proportion of flows larger than this is $9765625^{-1.1} \approx 0.00000002 = 0.000002\%$. So 0.000002% of flows carry 20% of the bytes. A similar calculation shows that 10^{-9}% of flows contain 10% of all bytes. In general p proportion of bytes are carried in

$$\text{Prop}_\gamma = p^{\frac{\gamma}{\gamma-1}} \tag{3}$$

of the largest flows. The feature that a small number of large flows contain a remarkably large proportion of bytes is well established [13, 6].

3 A Strategy for Providing QoS by Size Differentiation

The proposed LFD strategy measures the aggregate flow using a *flow-size measure*. If the aggregate flow is sufficiently small we do nothing, and the traffic will receive good performance. If the aggregate flow is large, we need to find the specific flow, or flows, causing the aggregate flow to be large so we divide the aggregate flow into two halves (or 32 32^{nd}s). At least one of the sub-aggregate flows must also be large (our measure of flow-size must have this property – this will be shown in Section 4). We continue in this way, as described in [1], until we have found one or more individual flows which are large. We then treat these flows separately, giving the bytes in them strictly lower priority than the bytes in the remaining flows. This strategy relies on some characteristics of the flow-size measure which will be further discussed in the next section.

If LFD is used, instead of individually monitoring and managing many flows, it is only necessary to monitor and manage a small number of large flows, together with one aggregate of the small flows. When a large flow arrives, but before it has been specifically identified, aggregates made up of small *and* large flows must be managed also.

If a large flow which is being treated separately ceases to be a large flow, which we discover by applying the flow-size measure, we will absorb this flow back into the aggregate of small flows, and cease to monitor it.

The advantages of this strategy include:

(i) the small flows experience a lightly loaded link;

(ii) the large flows experience greater delay, and jitter, however since these flows are large, and hence cannot be serviced with great speed anyway, the extra delay and jitter should be barely noticable.

A quantitative measure of the improvement in performance experienced by the short flows will be provided in Section 6.

4 How to measure flow size

Neither the *rate*, nor the *total bytes* in a flow are a good measure of its size. Even if the rate of a flow could be measured instantaneously, discriminating against a high rate flow would change it into a low rate flow, rendering the strategy unstable. Even if the total bytes in a flow was announced at the start of every flow, discriminating against large flows would prevent continuous bit-rate (CBR) flows from receiving good QoS.

So we need a measure in-between rate and byte-count. In this measure, VOIP flows should be small and if the complete aggregate is less than a certain size, 1 for example, performance delivered by a router should be satisfactory.

A flow-size measure, $\rho_t(\cdot)$, at time t, for example, should have the following characteristics (Φ, Ψ denote flows, a a positive number):

(i) homogeneity:

$$\rho_t(a \times \Phi) = a\rho_t(\Phi); \tag{4}$$

(ii) the triangle inequality:

$$\rho_t(\Phi + \Psi) \leq \rho_t(\Phi) + \rho_t(\Psi); \tag{5}$$

(iii) performance: if $\rho_t(\Phi) < 1$, Φ will experience no loss and minimal delay at time t ;

(iv) VOIP: a flow aggregate, Ψ, made up of a VOIP service has $\rho_t(\Psi)$ small for all t;

(v) persistence: if $\rho_t(\Phi) = B$, $\rho_s(\Phi) \approx B$ for s near t also.

Property (5) ensures when a large aggregate flow is divided into parts, at least one of the parts is also large, according to the flow-size measure. This property ensures that the divide-and-rule method for identifying a large flow will succeed.

4.1 Different Measures of Size

A candidate as a measure of flow size is the level of a virtual buffer, or a token bucket, at a certain moment t, which can be defined as $B_{t,\Phi} = \sup_{s \le t} \Phi(t,s) - (t-s)R_B$, where R_B is the rate of the buffer. The following inequality (a convexity property) holds for the level of a virtual buffer:

$$B_{t,a\Phi_1+b\Phi_2} \le aB_{t,\Phi_1} + bB_{t,\Phi_2}, \qquad (6)$$

where Φ_1, Φ_2 are flow aggregates and a, b are real positive numbers with $a+b=1$, $a,b \ge 0$. At the same time $B_{t,\Phi}$ is *not* a good measure of flow size since the characteristics (i) and (ii) do not hold.

For example, it is not unlikely that an aggregate flow which generates non-zero buffer levels in a buffer B, might be decomposed into two aggregate flows which pass through the same buffer without causing the buffer to fill at any time.

The first flow-size measure we wish to consider, $\rho_{B,t}(\cdot)$, relates closely to a buffer and the link into which it feeds, of rate r. We define the size $\rho_{B,t}(\Phi)$ of a flow aggregate Φ as the smallest number $a > 0$ such that the scaled flow Φ/a can pass through the link and buffer without *overflowing* the buffer at time t. This flow-size measure has the homogeneity property because when we rescale a flow, the smallest number $a > 0$ which causes an overflow is simply rescaled appropriately to cause an overflow. Using (6) and a basic result from [14] we can show that the triangle inequality, (5), is true for this measure $\rho_{B,t}$.

If a flow has size $\rho_{B,t}(\Phi) < 1$, it means that the flow will pass through the buffer without overflowing at time t. This shows Property (iii), although only at time t. Since a VOIP flow has low bit-rate and very little random variation, it is clear that Property (iv) holds.

The last property, (v), is difficult to evaluate definitively, however, we can adjust the degree of persistence of the measure of flow size introduced in the next section, a pragmatic implementation of a measure similar to $\rho_{B,t}$, to any desired level.

```
B[0] = 0; x[0] = 0; r = 0; t = 0;

while(t<T) {
  t = t + 1;
  b[t] = bytes arriving in interval t;
  r = phi*B[t];
  B[t+1] = max(beta*B[t],x[t] + b[t] - r);
  x[t+1] = max(0,x[t] + b[t] - r);
}

return B[T] / Actual_Buffer_size;
```

Figure 1. A Pragmatic Flow Size Algorithm

4.2 A pragmatic algorithm

In this section we define an algorithm for measuring flow size similar to the one from the previous section but which is easier to compute. It is shown in Figure 1. It may be interpreted as a virtual buffer in which the speed of the link and the capacity of the buffer increase as needed and then sink back exponentially when this extra capacity is not needed. It has two parameters, ϕ, the ratio between buffer capacity and link rate, and β, the decay-rate of buffer capacity when it is not full.

Homogeneity of the flow size algorithm of Figure 1 is easily verified by checking that if we replace $b[t]$ by $a \times b[t]$ for all t, the values of $B[t]$ and $x[t]$ will all be scaled by the same factor, at all stages of the algorithm. The other key property we need to verify is that $B[t]$, considered as a function on flows, satisfies the triangle inequality.

Convexity of $B[t]$ and $x[t]$ as functions of b ($b[0],\ldots,b[t]$), can be shown by induction. Clearly $B[0] = 0$ and $x[0] = 0$ are (trivially) convex functions of b. Now, assuming $B[t]$ and $x[t]$ are convex functions of b, observe that $B[t+1]$ is the maximum of two other functions which are, by the inductive hypothesis, convex functions of b. So, $B[t+1]$ is also a convex function on the same space. The same argument applies to $x[t+1]$ as a function of b, since it is also expressed as the maximum of two convex functions of b.

5 Algorithmic Complexity

The processing required in LFD is made up of the following components:

(i) Per-packet processing, which may be confined to classification (by means of a routing table), and byte counting. The complexity depends upon the number of large flows which, should be quite small, e.g. less than 10.

(ii) Flow-size calculation, which can occur somewhat less often than at the arrival of every packet. The algorithm of Figure 1 requires 4 additions and 2 multiplications, and two `max` operations, per iteration.

(iii) When a large flow arrives or departs, the routing tables for packet classification will be updated. The precise source-destination address and port pair needs to be found.

6 Flow level performance

Three quite different examples are considered in this section: a link with speed 128 kbits/s, a link with speed 10 Mbits/s, and a link with speed 10 Gbits/s. In each case, the critical performance measure is the probability that small flows are excessively delayed. We measure delay of a flow by the ratio of flow completion time (FCT) [5] to its minimal possible value, taking into account just the one link. We refer to this ratio as Relative Response Time (RRT) [11]. For the slow link, we consider, as our test of good performance, $P(RRT > 3)$, for the medium speed link, $P(RRT > 5)$, and for the fast link, $P(RRT > 10)$.

The results are presented in Table 1. For each column a target utilization level for the link is selected, then the remaining entries in the column are derived using formulae already derived together with the formulae for system performance which are derived in Subsection 6.1. If the resulting performance levels which result are not suitable, the target utilization has been adjusted up or down until the desired result is obtained. Table 1 shows the observed results at the conclusion of this iterative procedure.

The rate of arrival of the large flows, which need to be controlled in order to achieve LFD, has been calculated using (3). The average number of large flows has been calculated as $-\log(1-\rho_L)$, as justified in Subsection 6.1, where ρ_L is the utilization due to large flows, relative to the capacity of the system after the short flows have been removed (using a reduced load approximation). For example, in the case of the 10 Mbit/s link, the utilization due to large flows is 25%. If we treat the short flows as a constant bit-rate load on the link, this large flow load represents a proportion $25/65 = 38.5\%$ of the remaining capacity of the link. The mean number of large flows is therefore $-\log(0.615) = 0.486$.

The probabilities of RRT exceeding satisfactory levels, under either LFD or without LFD, has been calculated using the results of the following subsection. In summary, the results show that using LFD provides satisfactory RRT for short flows at utilization levels 20-30% higher than without LFD. Large flow delays also will be improved under LFD because for all flows, no matter how large, there is a yet larger flow which can potentially damage performance. To test the sensitivity of these results to our modeling assumptions, a second table, Table 2, of system performance estimates has been calculated using the assumption that $\gamma = 1.2$ in place of the choice $\gamma = 1.1$.

The last rows of Tables 1 and 2 have been calculated by means of (2). These sizes should be regarded as indicative only since they have been calculated under the assumption that flows have been ranked by total size and that this is Pareto distributed. In operation, flows will be ranked according to flow-size computed dynamically. If flows were delivered in bulk, instantaneously, to the Internet, at the first router the dynamic ranking would be very close to their ranking by total bytes. However, at subsequent routers the flows will be more spread out and the dynamic ranking will diverge increasingly from their byte-count ranking.

Although it is reasonable to expect that the flow-sizes computed dynamically will closely correlated with flow-sizes measured in number of bytes, little more can be said about computed flow-sizes in networks until measurements in real networks have been undertaken.

6.1 Performance Model

The performance model we use to derive the results in Table 1 has been constructed on the assumption that the parameter $\gamma \approx 1$. As $\gamma \longrightarrow 1$, the range of flow sizes becomes more and more diverse, and it becomes more and more likely that adjacent or nearby flows will be of different lengths. Another way to express this is that it becomes less likely that a shorter flow will overlap with the beginning or end of a larger one.

Let us assume that there are *no overlaps of smaller flows with the beginnings or endings of larger flows*. This situation is depicted in Figure 2. The lines in this figure indicate the periods of time during which flows are being served by the outgoing link. We are interested in two alternatives: the one where shorter flows receive strict priority over long ones (Shortest Job First – SJF), and the other where all flows receive an equal amount of bandwidth (Fair Queueing – FQ).

Figure 2. Non-overlapping network flows

To see why overlaps can be ignored, choose a particular flow length, ℓ and consider the *larger* flows which contain the end point of this flow. These have a Pareto distribution with minimum size δ and shape parameter $\gamma - 1$. The mean length of these flows is infinite, and therefore the proportion of them in which the starting point occurs before the start of the larger flow is zero and so the probability that an overlap occurs between the short flow and the start of a long flow, is zero.

Under these assumptions, the distribution of the number of simultaneous flows under SJF will be close to Poisson. The mean duration of a flow of length ℓ is not affected by whether larger flows occur at the same time, and is not *sig-*

	Link 1	**Link 2**	**Link 3**
Speed	128 kbps	2 Mbps	10 Gbits/s
Total Utilization (ρ)	50%	60%	70%
Large flow utilization	30%	25%	20%
Performance using LFD	$P(RRT_s > 3) \approx 0.002$	$P(RRT_s > 5) \approx 0.004$	$P(RRT_s > 12) \approx 0.004$
Performance not using LFD	$P(RRT_s > 3) \approx 0.12$	$P(RRT_s > 5) \approx 0.23$	$P(RRT_s > 12) \approx 1$
Rate of arrival of large flows	0.0006	0.0013	0.6
Mean number of large flows	0.47	0.49	0.51
Length of large flows	6 Mbytes	40 Mbytes	400 Mbytes

Table 1. Table showing performance of LFD, assuming $\gamma = 1.1$

	Link 1	**Link 2**	**Link 3**
Speed	128 kbps	2 Mbps	10 Gbits/s
Total Utilization (ρ)	50%	60%	60%
Large flow utilization	30%	25%	10%
Using LFD	$P(RRT_s > 3) \approx 0.002$	$P(RRT_s > 5) \approx 0.004$	$P(RRT_s > 12) \approx 0.004$
No LFD	$P(RRT_s > 3) \approx 0.12$	$P(RRT_s > 5) \approx 0.23$	$P(RRT_s > 12) \approx 0.14$
Rate of arrival of large flows	0.348	14	29
Mean number of large flows	0.47	0.49	0.22
Length of large flows	16 kbytes	40 kbytes	4 Mbytes

Table 2. Table showing performance of LFD, assuming $\gamma = 1.2$

nificantly affected by whether, and how many, shorter flows have occurred at the same time, because these shorter flows will have only a minor impact on the duration of the longer flow. Therefore, the total number of flows which intersect this moment in time can be regarded as the outcome of an infinite collection of independent events, and is therefore Poisson distributed.

We know that the probability of zero simultaneous flows must be $1-\rho$, but in a Poisson distribution with mean μ, the probability of zero is $e^{-\mu}$, so the mean number of the Poisson distribution must be $-\log(1-\rho)$. It follows that the probability of k simultaneous flows, assuming the queue discipline is SJF, is:

$$P_{\mathrm{SJF}}(k) = \frac{(-\log(1-\rho))^k (1-\rho)}{k!}, \qquad k \geq 0. \quad (7)$$

As we switch between from SJF to FQ the flow lengths as depicted in Figure 2 should increase, depending upon how many flows overlap, and potentially an overlap between a flow and the end of a larger flow could be introduced, but we will assume that this doesn't happen for the same reason as before. If two flows overlap, the overlap interval should therefore increase by 2. This increase in the duration of two flows at a time will decrease the proportion of time occupied by only one flow at a time to $(-\log(1-\rho))(1-\rho) - \frac{1}{2}(-\log(1-\rho))^2(1-\rho)$.

If three flows overlap, the length of their overlap will increase by a factor of 3. The *number* of overlaps between 3 simultaneous flows will already have been increased by a factor of 2, because of the increase in length of 2-overlaps caused by the switch from SJF to FQ, so the combined effect, of switching from SJF to FQ will decrease the proportion of time occupied by two flows at once, by $2 \times 3 \times P_{\mathrm{SJF}}(3)$.

So, ignoring end-overlaps, $P_{\mathrm{FQ}}(2) = (-\log(1-\rho))^2(1-\rho) - \left(1-\frac{1}{3!}\right)(-\log(1-\rho))^3(1-\rho)$, and in general

$$P_{\mathrm{FQ}}(k) = k!P_{\mathrm{SJF}}(k) - (k+1)!P_{\mathrm{SJF}}(k+1) + P_{\mathrm{SJF}}(k+1),$$

$k > 0$. It follows that

$$\begin{aligned} P_{\mathrm{FQ}}(\geq k) &= k!P_{\mathrm{SJF}}(k) + P_{\mathrm{SJF}}(\geq k+1) \\ &= (-\log(1-\rho))^k(1-\rho) + P_{\mathrm{SJF}}(\geq k+1), \quad (8) \end{aligned}$$

$k \geq 1$.

Equation (8) also gives us the distribution of RRT for the shortest flows under FQ since during the lifetime of such flows, the number of the other flows also active will most likely stay the same. It is known that the mean RRT is the same for flows of different lengths and simulations show that the entire distribution of RRT is much the same for long flows as for short ones. This model of the RRT of flows passing through a system where the FQ discipline is used has been used to estimate the tail probabilities of RRT shown in Tables 1 and 2.

The entries in these tables for RRT experienced by the short flows in the case where LFD *is* used have been calculated by applying the above model to just the short-flow traffic. Since the short-flow traffic has a lower variance than traffic with a complete range of flow lengths, these are conservative estimates of the true probabilities of RRT exceedence.

Note that if $\rho > 1 - e^{-1} = 0.632$, $-\log(1-\rho) > 1$ and (8) no longer defines a distribution function. This can be explained by the failure of our assumption that under SJF the duration of a long flow at a given time t is independent from the number of shorter flows which overlap with this moment.

6.2 Simulations

A special purpose simulator of a router with the FQ and SJF disciplines has been implemented. This simulator treats each flow, or burst, as a job with a Pareto distributed length.

Comparison of the theory of the performance of FQ from Section 6 with simulations is shown in Figure 3. Since the processes being simulated are long-range dependent, the accuracy achievable by means of simulation is severely limited. Although hundreds of millions of flows have been simulated, the discrepancy between theory and simulation at $k = 1$ in this figure (where we know the theory is exact) is already noticeable. The method used in [2] to improve the accuracy of simulations of a long-range dependent process could be used here, but has not so far been undertaken. It is to be expected that the simulations will provide answers somewhat lower than the correct values. Consequently the discrepancy between theory and simulations depicted in Figure 3 should be expected and is more likely to be caused by the difficulty of simulation than inaccuracy of the model.

7 Conclusion

Now that we have developed and validated a mathematical model of the way the FQ and SJF queueing disciplines will work in practise, we are ready to return to the questions raised in the introduction.

Q1: Does this method really provide sufficient Quality of Service for the small flows? Yes. By withdrawing a very small proportion of very large flows from competition with the small ones, they can get much better performance.

Q2: How many large flows need to be treated separately? Rarely more than 10, although during transients, when large flows are being identified, the need for fast response might dictate that more aggregate flows are monitored simultaneously.

Q3: How frequently do new large flows arrive and depart? Less than 1 per second, except in the most extreme case considered, the 10 Gbit/s link, with $\gamma = 1.2$, in which case a rate of 1 large flow per millisecond is expected.

Q4: How should we measure the size of a flow? A fast, simple algorithm for measuring flow-size was identified. Tests in a router are needed.

Q5: Will some small flows be mixed up with large flows and receive bad performance? This question needs further study.

Q6: How bad is performance for small flows anyway, if we do nothing? The advantage of LFD over FQ appears to be similar to a change of link speed by 10% – 30%, depending upon the size of the link, and the type of traffic.

Q7: How sensitive are the supposed benefits of the Large Flow Discrimination (LFD) to the assumptions regarding Internet traffic? The key assumption is that flows have a heavy-tailed distribution. This is well-established in the literature on traffic measurements [13, 6]. We have varied the parameter γ to test the sensitivity to this assumption. When $\gamma = 1.2$, the benefits of LFD are reduced for large links. However, the benefits of LFD remain very significant for small links.

Q8: How quickly do we need to identify large flows in order to manage their impact on small flows? This question requires further study.

Q9: What will happen to a large flow which needs high QoS; in particular, a large continuous bit-rate flow which needs low jitter? There is a certain size *below which* flows can be guaranteed satisfactory QoS in all senses, including jitter. Above this size, flows will still receive good flow completion times, but guaranteeing jitter will be difficult. Sizes below which a flow can be regarded as small vary depending on link capacity (See Table 1).

Q10: Doesn't DiffServ already solve the QoS problem? It remains to be seen whether DiffServ can be deployed sufficiently widely and thoroughly to achieve a satisfactory approximation to guaranteed Quality of Service. Also, LFD can be used for any DiffServ class to provide better flow completion times for the flows *within* a class, and to protect the short flows in each class from accidental or inadvertent abuse by other users.

By discriminating against large flows we can achieve significantly better performance than fair queueing, for all flow sizes. The problem of allocating weights to different flows, as in weighted fair queueing, has been bypassed. Since the large flows identify themselves, we do not need tagging of packets, or policing of flows. And since the number of flows, and their arrival rate, are low, the algorithms to implement LFD in routers are fast and scalable. The LFD queue discipline shows great promise for improving the flow level performance of routers, which needs to be further investigated by experiments, simulations, and mathematical modeling.

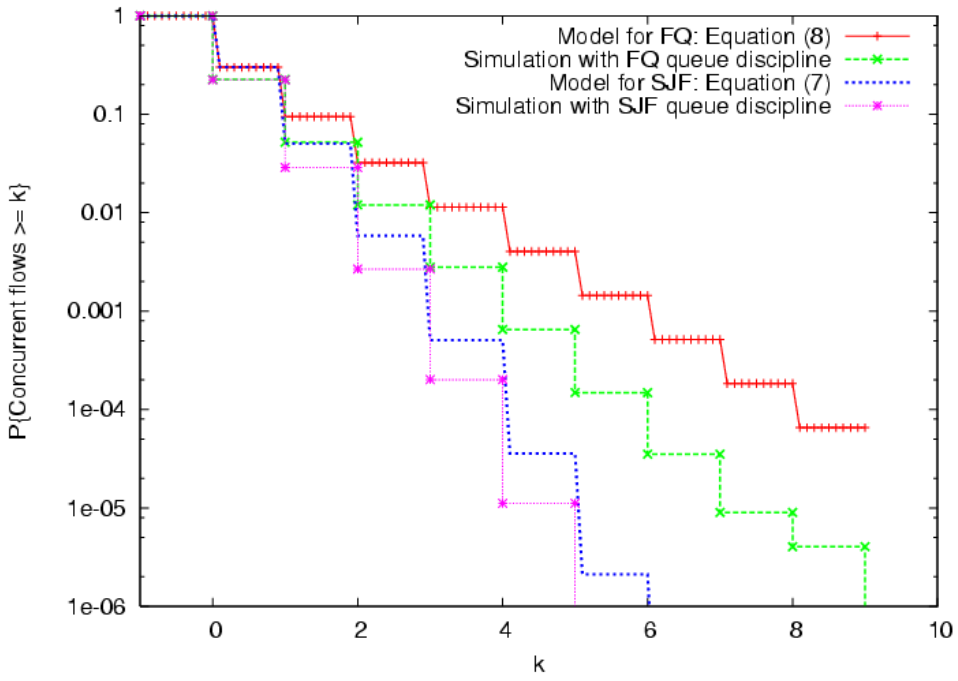

Figure 3. Comparison of predicted concurrent flows distribution from theory and simulations for FQ and SJF queue disciplines

References

[1] R. G. Addie, S. Braithwaite, J. das Gupta, and J. Leis. Aggregate flows – for efficient management of large flows in the internet. In *Proceedings of the 5th Workshop on the Internet, Telecommunications and Signal Processing*, December 2006.

[2] R. G. Addie, T. M. Neame, and M. Zukerman. Performance evaluation of a queue fed by a Poisson Pareto burst process. *Computer Networks*, 40:377–397, October 2002.

[3] R. G. Addie, M. Zukerman, and T. M. Neame. Fractal traffic: Measurements, modelling and performance evaluation. In *Proceedings, IEEE Infocom 1995*. IEEE, April 1995.

[4] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview. Technical Report RFC 1633, IETF, June 1994.

[5] N. Dukkipati and N. McKeown. Why flow-completion time is the right metric for congestion control. *ACM SIGCOMM Computer Communication Review*, 36(1):59 – 62, January 2006.

[6] L. Guo and I. Matta. The war between mice and elephants. In *Proceedings of ICNP'2001: The 9th IEEE International Conference on Network Protocols*, pages 180–188, November 2001.

[7] N. Hohn, D. Veitch, and P. Abry. Cluster processes, a natural language for network traffic. *IEEE Transactions on Signal Processing, Special Issue on Signal Processing in Networking*, 51(8):2222–2249, 2003.

[8] N. Hohn, D. Veitch, and P. Abry. The impact of the flow arrival process in internet traffic. In *Proc. IEEE ICASSP*, pages VI 37–40, 2003.

[9] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2:1–15, 1994.

[10] N. Likhanov and R. R. Mazumdar. Cell loss asymptotics in buffers fed with a large number of independent stationary sources. In *Proceedings of IEEE Infocom '98*, 1998.

[11] D. McNickle and R. G. Addie. Comparing protocols for differential service in the internet. In *Proceedings of IEEE TENCON 2005*. IEEE, IEEE, December 2005.

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

[13] V. Paxson and S. Floyd. Wide-area traffic: The failure of poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995.

[14] A. P. Robertson and W. J. Robertson. *Topological Vector Spaces*. Cambridge Univesity Press, 1966.