Snapshot Simulation of Internet Traffic: Fast and Accurate for Heavy-tailed Flows

R. G. Addie,
University of Southern Queensland, Australia.
addie@usq.edu.au

ABSTRACT

Simulations involving processes at very different time scales can be so slow to converge that starting in one state and waiting for a representative sample of the state space to be explored is not feasible. Under these circumstances we need to find a different way to explore a representative range of states in order to obtain valid results in a reasonable time. Internet traffic is an example of this situation. This is due to the fact that it is made up of clearly identifiable flows and a significant proportion of overall bytes occur in long-lived *flows, whose overall duration will in many cases be longer than can be simulated.*

In this paper we develop a method which constructs a "randomly selected state" of Internet traffic from scratch – snapshot simulation*. The technique is applied to a realistic model of Internet traffic and is used to confirm theoretical results for the number of active flows in a router which adopts either Fair Queueing or Shortest Job First as its queueing discipline. Snapshot simulations are also compared to conventional simulations of the same systems and it is shown that although convergence to the same results appears to be occurring, satisfactory accuracy cannot be obtained in a reasonable time by the conventional simulations. Simulation of a practical high-performance queue discipline, SJF-n, in which only the largest n flows at any time are de-prioritized, is also simulated and shown to give good performance for quite low n.*

1. INTRODUCTION

An important measure of performance in the Internet is the duration of time between the initiation of a request and the complete delivery of a response. This is a complex event, including a number of related components, and it is therefore appropriate to break it into simpler components. It would be a good step if we merely knew the distribution of the time between the initiation of a flow, and its completed delivery. This measure of Internet performance is known as *flow completion time*, and it has been observed that analytic formulae for flow completion time distributions are not presently available [1].

When the queue discipline is Fair Queueing (FQ), flow completion times for shorter flows are largely determined by the number of active longer flows present during the lifetime of a flow. For other queue disciplines, also, a complete knowledge of the distribution of the number of active flows gives us critical insight into the behaviour of flow completion times. For this reason we concentrate, in this paper, on simulation techniques which allow us to estimate the distribution of the *number of active flows*.

Analytic modeling and simulation of Internet traffic behaviour currently focusses on *scenarios* because long-range dependence of traffic means that accurate simulation of a realistic traffic model requires simulation of processes at very different time scales. Simulation of processes at dramatically different time scales is infeasible unless some technique for avoiding the massive complexity of simulating the short time-scale processes for the entire duration of a process occurring at the long time scale is used.

To address this problem of having to simulate a mountain of detail for the considerable duration of the important long flows, the technique of Time-stepped hybrid simulation was introduced in [2]. In this approach, details in the simulation of the TCP protocol are appropriately reduced, and by introducing *chunks* of time, simulation of individual packets is also largely avoided. The approach of the present paper is quite different in that the level of detail simulated varies depending upon the arrival time of a flow. Although the simulations presented in this paper are predominately at a higher level of detail, so that individual packets are not simulated, such details could in principal be included for *some* packets, the ones most relevant to the moment under observation.

We confine out attention in this paper to the queueing disciplines Fair Queueing (FQ) [3], Shortest Job First (SJF) [4] and the discipline SJF-n which gives low priority to the n largest flows, and treats the remainder according to FQ. These disciplines are defined for arbitrary flows (not necessarily subdivided into packets) competing for the resources of an outgoing link at a router. If packets are taken into account, FQ must be implemented by a round-robin service quantum for each competing flow, and likewise SJF cannot not allow delivery of a packet from a flow to be interrupted, even when a shorter flow arrives. Even this model of queueing in a router is an oversimplification since the process of allocating resources between flows in a router relies on active queue management (AQM) in the router and the operation of the TCP and UDP protocols in individual hosts. However, at this stage, incorporating these details has not been undertaken. The fact that queue disciplines approximating FQ and SJF-n can be implemented by an appropriate AQM is discussed elsewhere (in particular, see [4]).

In a wide range of situations SJF, or the closely related Shortest Remaining Processing Time First (SRPT) [5], have been shown to be optimal. So results for SJF can show us how much worse than optimal FQ is. The discipline SJF-n on the other hand is a discipline which we shall see performs nearly as well as the optimal

QoSim 2008, March 3, 2008, Marseille, France

SJF discipline and yet is practical to implement (as investigated in [6]); this discipline is difficult to analyze except by simulation. It is explained below, also, how systems with other queue disciplines can be simulated by the same technique.

The behavior of flows in the Internet depends upon the traffic conditions that they encounter in a complex manner. For a good understanding of the performance experienced by flows we need a model which includes sufficient details of this complex interaction between flows, and the resources they are accessing. At one extreme, we have the situation where all flows sharing a certain link are simultaneously bottlenecked by this resource (the *fully bottlenecked case*). At another extreme, the flows appearing at this certain link are all bottlenecked at other locations (the *uncongested case* – since there is no congestion at the router being observed). A realistic model allows for both cases simultaneously: flows which are bottlenecked here, and therefore compete with other for the capacity of the link, and flows which are bottlenecked elsewhere, and therefore behave in a manner independent from the other flows. All the models simulated in this paper are of the fully bottlenecked variety – all flows are congested at the router under study – however cases where some or all of the flows experience congestion elsewhere can readily be simulated by the same methods.

Up to now, analytic solutions have been developed only for separate models of the uncongested case [7] and the fully bottlenecked case which can be analysed as a process sharing queue using the invariance result obtained in [8], which is derived also in [9]. The simulation of the uncongested case presented in [7] makes use of a technique related to snapshot simulation in order to enable satisfactory results in reasonable simulation times. Conventional simulations were started in a carefully selected variety of initial states. The initial states varied with respect to the number of "long flows" present, where a "long flow" was defined to be a flow which persists for the entire duration of the simulation. The simulations in [7] were still very long and nevertheless could provide confidence in regard to accuracy for a limited range of system parameters.

Naive simulations, which do not make use of any importance sampling technique, have also been used but without providing any confidence as to their accuracy. It is also common practise to simulate *scenarios*, in which some specific configuration of long flows is assumed to provide the background for other shorter flows which compete for resources, e.g. with NS2 [10]. Such simulations can make no claim to characterizing *typical characteristic* behaviour of the Internet.

The simulations carried out in this paper are in accordance with an overall statistical model of flows (as presented in Subsection 2.1) but are much faster than traditional simulations (comparisons are presented in Section 4). There is no inherent restriction on the level of detail which *could* be included. Although the simulations here include no more than basic details, at the heart of the snapshot simulation method lies a conventional simulation which can include arbitrary details.

The difficulty of derivation of analytical results, for either the uncongested or the fully bottlenecked cases, and the approximations required to justify their derivation demand that their validity and accuracy be checked by simulation. In addition, the more practical case where some flows are uncongested and others are bottlenecked, which at present has not been treated analytically, can be tackled by the simulations presented here.

The approach to simulation which we use here is to estimate the stationary distribution by sampling a stationary process at a random moment in time – a *snapshot*. This method of simulation proceeds through exploration of the range of possible events affecting the present moment in time, from the more significant, to the less significant.

The reason this approach is enormously faster is that although the simulation takes into account events from the distant past, it does not simulate unnecessary fine details from the distant past. Very short flows (of which there are vastly more than long flows) are simulated in full detail only in the very recent past, short flows only in the recent past, and so on. Of events from the distant past, only the very longest flows are simulated in full detail, and there are only a very small number of these occurring.

In Section 2, the model of Internet traffic adopted in this paper is set out. In Section 3, the simulation algorithm is described in detail. In Section 4, results obtained using the simulation method of this paper are compared with analytical results for the same models, and concluding remarks are presented in Section 5.

2. SYSTEM MODEL

2.1 Internet Traffic Model

Internet traffic has been found to be *long-range dependent* and *self-similar* [11]. 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 [12, 13, 14].

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 [15] 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 τ is Pareto distributed:

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

and the corresponding density is

$$\frac{d\Pr(x < \tau \leq x + dx)}{dx} = \begin{cases} \frac{\gamma}{\delta}\left(\frac{x}{\delta}\right)^{-\gamma-1}, & x \geq \delta, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

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 $0.5^{-\frac{1}{1.1}} = 2^{\frac{1}{1.1}}$.

Different ways to measure the size of a flow, f, include: *total bytes* ($\tau(f)$), *duration* ($d(f)$), and *rate* ($r(f)$). Each of these is relevant in certain contexts. In the simple case where the rate is fixed, they are related by

$$d = \tau / r.$$

Generally we shall assume that the Pareto distribution applies to the *size* of flows. The distribution of the *rate* at which a flow is served cannot be specified as a descriptive system parameter because in normal operation it will depend upon the conditions which apply at the time.

However, it is more reasonable to suppose that with each flow is associated a certain *maximum rate*, $R(f)$, above which this flow cannot be forced to operate, even in an empty network. If we specify an a-priori distribution, e.g. an exponential distribution, for $R(f)$, the consequential value for $r(f)$, at any point in time, may be deduced from the conditions which apply at the time.

If we aggregate statistics of network observations in proportion to the affected bytes instead of according to the affected flows we

discover an implicit Pareto distribution with a much heavier tail. Let D denote the length of a flow containing a *byte* chosen at random. Then

$$\Pr(D>y) = \begin{cases} \int_y^\infty \left(\frac{x\gamma}{\delta}\right)\left(\frac{x}{\delta}\right)^{-\gamma-1}dx \Big/ \int_\delta^\infty \left(\frac{x\gamma}{\delta}\right)\left(\frac{x}{\delta}\right)^{-\gamma-1}dx, & y \geq \delta, \\ 1, \quad \text{otherwise}, \end{cases}$$

$$= \begin{cases} \left(\frac{y}{\delta}\right)^{1-\gamma}, & y \geq \delta, \\ 1, & \text{otherwise}. \end{cases} \quad (3)$$

which is another Pareto distribution but with shape parameter $\gamma-1$ now instead of γ. 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

$$\text{Prop}_\gamma = p^{\frac{\gamma}{\gamma-1}} \quad (4)$$

of the largest flows. The feature that a small number of large flows contain a remarkably large proportion of bytes is well established [16, 17].

2.2 Processing Model

Denote the size of the flow, f, by $\tau(f)$. In the FQ server, the capacity of the server is shared equally among all flows currently active at the server. In the SJF case, flows are served strictly in order of priority determined by $\tau(f)$, shorter flows ahead of longer ones, pre-empting any longer flows which are active when they arrive. In the SJF-*n* case, when there are fewer than n flows present, the discipline is identical to SJF. When more than n flows are present, the flows shorter than the n longest are given equal access to the server in the same manner as FQ, but the long flows receive no service.

The system utilization due to flows shorter than τ is denoted by $\rho(\tau)$, so, using (2),

$$\rho(\tau) = \frac{\lambda}{C}\int_\delta^\tau \left(\frac{x\gamma}{\delta}\right)\left(\frac{x}{\delta}\right)^{-1-\gamma}dx$$

$$= \frac{\lambda\gamma\delta}{C(\gamma-1)}\left(1-\frac{\tau^{1-\gamma}}{\delta^{1-\gamma}}\right). \quad (5)$$

and the overall processor utilization is denoted simply by $\rho = \rho(\infty) = \frac{\lambda\gamma\delta}{C(\gamma-1)}$.

3. THE SIMULATION TECHNIQUE

The simulation proceeds by repeatedly simulating instantaneous snapshots (states) at a certain moment in time, which, since the origin of our time scale is arbitrary, we take to be time 0. Each such simulation is an independent, typical, view of the state of the system being simulated. Each snapshot is simulated, in turn, by using the observation that the flows arriving at a router in the Internet form a two dimensional Poisson process, where one dimension is the conventional dimension of time-of-arrival, and the other dimension is flow size. Whereas a conventional simulation exploits the Poisson property of the flow arrival instants, the simulation algorithm presented here is based, instead, on the independence of the arrivals of flows of different sizes.

This approach of generating flows in increasing order of their size, rather than increasing order of arrival is only used to *generate* the flows. (The flows can just as easily be generated in decreasing order of size – this makes no difference to subsequent processing when the discipline is FQ.) These flows are then stored in a list in order of arrival. Once all flows have been generated and stored, a conventional simulation, in normal time order, takes place. All arrivals and departures need to be simulated, and when departures occur, if the discipline is FQ, the remaining processing time associated with all the flows which are currently sharing the outgoing link is updated. In the case of the SJF discipline we can take advantage of the fact that flows are generated in increasing order of size to reduce the simulation stage to a triviality. By the time all flows have been generated, it is already clear how many flows are will be active at time 0. The SJF-n discpline, on the other hand, is a little more complicated than FQ, and it is necessary to maintain a list of currently active flows, in order of their size, during the simulation. This is obviously necessary since the selection of which flows are served depends on the relative size of the other flows present.

Any simulation procedure may be placed after the flow generation stage and there are no inherent limitations on the range of statistics which can be collected during this simulation.

3.1 Simulation in order of Size

The flows form a non-homogeneous point process in the two-dimensional space spanned by time, on one axis, and flow size, on the other. We can therefore generate flows in increasing order of size and decide on the arrival time after choosing the size. (Or, we could subdivide the space spanned by time of arrival and size of flow in a different way, and generate flows in an order different from time of arrival and also different from size.) Let us suppose, to be more specific, that our simulation is be confined to the period of time $(-T,0)$. With probability 1 no two flows will have exactly the same length. We will only consider flows which arrive in the interval $(-T,0)$. In this case, the expected number of flows with size in the interval (τ,∞) will be $\lambda T\left(\frac{\tau}{\delta}\right)^{-\gamma}$, in accordance with (1), to give the correct weight for this range of flow sizes.

Here is an inefficient and slightly inaccurate method to simulate these flows in order of size. Choose a small number, Δ , sufficiently small that the probability of two flows arriving both with lengths in the interval $(\tau,\tau+\Delta)$ is very small, for any $\tau \geq \delta$, and also choose a large integer, K, sufficiently large that the probability of a flow larger than $K\Delta$ arriving in the interval $(-T,0)$ is very small. Now, for each $k = 1,\ldots,K$, choose a Bernoulli random variable with probability of 1 given by $\lambda T\left(\left(\frac{k\Delta}{\delta}\right)^{-\gamma} - \left(\frac{(k+1)\Delta}{\delta}\right)^{-\gamma}\right)$ to determine whether a flow of length in the interval $(k\Delta,(k+1)\Delta)$ occurs or not. For each of these flows, assign its length to be a number in the interval $(k\Delta,(k+1)\Delta)$ and select a pseudo-random number uniformly distributed on the interval $(-T,0)$ as its arrival time. For sufficiently small Δ and sufficiently large K this method will simulate the flow sizes and arrival times accurately.

A faster and completely accurate simulation method can easily be identified. Instead of subdividing the flow axis into small intervals, we can simulate the random intervals between the successive lengths of the flows which occur, in the same way that we simulate a Poisson processes on the time axis by simulating inter-arrival times. The non-homogeneity of the Poisson process of flow sizes can be taken into account as follows. First define the monotonic function $\psi : \tau_1 \mapsto \psi_1 = \psi(\tau_1)$ by

$$\psi(\tau) = E\left(\#\{\text{flows of length} \leq \tau\}\right) = \lambda T\left(\frac{\tau}{\delta}\right)^{-\gamma}. \quad (6)$$

Suppose $\{\tau_1,\ldots,\tau_n\}$ is the set of flow sizes which occur in the interval $(-T,0)$. This is a non-homogeneous Poisson process on $(0,\infty)$. Now consider the set of *points* $\{\psi_1 = \psi(\tau_1),\ldots,\psi_K =$

$\psi(\tau_K)\}$. Let us show that this is a *homogeneous* Poisson process with intensity 1 on $(0,\lambda T)$. Denote the inverse of the function ψ by ϕ, i.e. it is the unique function such that $\psi(\phi(x)) = x$ for all $x \geq 0$.

To prove that $\{\psi_1,\ldots,\psi_K\}$ is a homogeneous Poisson process with intensity 1 it suffices to show that: (a) the number of ψs in one interval is independent from the number in any disjoint interval; (b) the expected number in an interval of length Δ is Δ.

Part (a) follows because the number of flows with sizes in one interval is independent from the number with sizes in a disjoint interval – and since ψ is monotonic two disjoint intervals of flow sizes are mapped to two disjoint intervals of ψ values ; to check (b) consider the expected number occurring in an infinitesimal interval of sizes, e.g. $(u,u+du)$. It is sufficient to confirm (b) on infinitesimal intervals. Let $\tau = \phi(u)$ and $d\tau = \frac{d\phi(u)}{du}du$ so $du = \frac{d\psi(\tau)}{d\tau}d\tau$. The expected number of flows in the interval $(\tau,\tau+d\tau)$ is (by the definition of $\psi(\tau)$ as the expected number of flows in the interval (δ,τ)) $\psi(\tau+d\tau)-\psi(\tau)) = u+\frac{d\psi(\tau)}{d\tau}d\tau - u = u+du-u = du$, i.e. the expected number of ψ's in the interval $(u,u+du)$ is du, as we wished to show. This shows that the process $\{\psi_1,\ldots,\psi_K\}$ is Poisson, with rate 1, on $(0,\lambda T)$.

Having established that the ψs are Poisson with intensity 1 on $(0,\lambda T)$, we now switch our point of view and simulate the ψs as a way to generate the flow lengths. Since the ψs and the flow lengths are related to each other by an invertible mapping, if one of them has the desired joint distribution, so does the other. When we simulate the ψs, we can generate the flow lengths by using the transformation ϕ, the inverse of the transformation which maps from flow lengths to ψs.

Since it is a *homogeneous* Poisson process, the ψ sequence can be generated by drawing a sequence of exponential random numbers, $\xi_1,\ldots,\xi_{K+1}$, with mean 1, such that

$$\psi_k = \sum_{j=1}^{k} \xi_j, \quad k = 1,\ldots,K+1,$$

and $\psi_k < \lambda T < \psi_{K+1}$.

We then calculate a corresponding sequence of flow lengths by the rule $\tau_k = \phi(\psi_k)$; these must then have the same joint distribution as the lengths of the flows which arrive in $(-T,0)$.

Next we will consider how to repeat this argument, in a slightly more complicated form, in order to derive a *faster* scheme for generating flow sizes with the correct joint distribution.

The majority of flows in this simulation will be very short and will occur at times well before 0. Since these flows will have minimal impact on the state of the simulation at time 0, a simple technique to speed up the simulation will be to skip the details of any flows which will terminate well before 0, taking them into account instead merely by their impact on the rate of service of the longer flows. The statistics we aim to collect are all based only on the presence or otherwise of flows at time 0, so it is not necessary to accurately simulate details which have minimal effect on the flow state at that time.

3.2 How to Generate Fewer Flows

The selection of the length of flows is achieved using a generalization of the procedure outlined in the previous subsection. Rather than justify this generalization, let us simply describe the procedure and prove that the generated process of flows has the correct statistics subsequently.

The procedure begins by selecting pseudo-random negative exponentially distributed random numbers, $\xi_1,\ldots,\xi_{K+1}$, with mean equal to 1, stopping as soon as the sum of these numbers exceeds E. The formula for E will be given shortly. We then set $\psi_k = \sum_{j=1}^{k}\xi_j$, $k = 1,\ldots,K+1$, and set $\tau_k = \phi(\psi_k)$, $k = 1,\ldots,K$ – these are the flow lengths – with $\phi = \psi^{-1}$, in which ψ is the function

$$\psi(\tau) = E\left(\#\{\text{flows of length} \leq \tau\}\right). \tag{7}$$

This procedure appears to be *identical* to the procedure introduced in Subsection 3.1. However in equation (7) no mention is made of *when* the flows arrive. Previously flows were constrained only to arrive in the interval $(-T,0)$. Now we change this assumption to something a little more interesting.

To speed up our simulations dramatically, instead of simulating *all* the flows which arrive in the interval $(-T,0)$, for some fixed T, we now simulate only flows arriving in the interval $(-d(\tau),0)$, in which $d(\tau)$ is a linear function of τ, the length of the flow. The function d will be chosen larger than the duration of the flow, but sufficiently close to it to significantly reduce the effort expended simulating short flows which complete well before time 0 and therefore have minimal impact on the state at time 0. With this in mind, we chose $d(\tau) = \frac{W\tau}{(1-\rho)C}$, in which W here is a parameter which allows us to control the error due to not simulating shorter flows. The minimum value we can sensibly use for W is 1 – with this value most flows which will still be alive at time 0 will be simulated; larger values of W should at least be tried. The largest value we have used in the simulations reported here is 1000.

Having chosen which flows will be simulated, let us now evaluate (7):

$$\begin{aligned}
\psi(\tau) &= \int_{\delta}^{\tau} \frac{\lambda\gamma}{\delta}\left(\frac{u}{\delta}\right)^{-\gamma-1} d(u)du \\
&= \int_{\delta}^{\tau} \frac{\lambda\gamma}{\delta}\left(\frac{u}{\delta}\right)^{-\gamma-1} \frac{Wu}{(1-\rho)C}du \\
&= \frac{\lambda\gamma W}{(1-\rho)C}\int_{\delta}^{\tau}\left(\frac{u}{\delta}\right)^{-\gamma}du \\
&= \frac{\lambda\delta\gamma W}{(1-\rho)C(\gamma-1)}\left(1-\left(\frac{\tau}{\delta}\right)^{1-\gamma}\right).
\end{aligned} \tag{8}$$

The expected number of flows is therefore

$$E = \psi(\infty) = \frac{\lambda\delta\gamma W}{(1-\rho)C(\gamma-1)} \tag{9}$$

and the function ϕ, the inverse of ψ, needed to generate flow lengths from the random numbers ψ_k, is

$$\phi(\psi) = \delta\left(1-\frac{(1-\rho)C(\gamma-1)\psi}{\lambda\delta\gamma W}\right)^{\frac{1}{1-\gamma}}. \tag{10}$$

3.3 The Simulation Algorithm

The simulation algorithm is shown in Algorithm 1. An implementation of this algorithm in the C programming language, with extensive testing and debugging facilities, has been written, tested and validated. The code may be viewed in [18].

The argument given above, in Subsection 3.1, to show that the procedure given there for generating flows does so according to the correct joint distribution could be repeated here, with minor adjustments, to show that the simulation algorithm generates flow sizes according to the correct distribution. However, here is a different argument which shows the same thing: the ψs generated in this algorithm form a Poisson point process on the interval $(0,E)$, where E is given in (9). By a well known result characterizing Poisson processes, the *number* of ψs is Poisson distributed, with mean E, and the individual ψs, when their order is ignored, have a uniform distribution on $(0,E)$. Let $\widetilde{\psi}_k = \psi_k/E$, $k = 1,\ldots,K$. These are uniformly distributed on $[0,1]$. Define $\widetilde{\phi}(u) = \phi(Eu)$ and

Algorithm 1 The Snapshot Simulation Algorithm

$N = 1000$ {Number of snapshots in total}
for $i \leftarrow 0; i < N; i \leftarrow i+1$ **do**
  $\mathcal{E} \leftarrow E$(number of additional flows)
  $k \leftarrow 1$
  for $\psi_k \leftarrow$ exprand(1); $\psi_k < \mathcal{E}$; $\psi_k \leftarrow \psi_{k-1}+$exprand(1) **do**
    $\tau_k \leftarrow \phi(\psi_k)$
    $d \leftarrow W\tau_k/(C(1-\rho))$
    Choose arrival time, $\alpha \leftarrow$ uniformrand$(-d,0)$
    $k \leftarrow k+1$
  End of construction of a simulation configuration
  Calculate number of active flows at time 0
  Accumulate results (e.g. histogram of number of active flows)
End of simulation

$\widetilde{\psi}(x) = \psi(x)/E$, so $\widetilde{\phi}$ is the inverse of the function $\widetilde{\psi}$. Then, if $\widetilde{\psi}_0$ is a randomly chosen $\widetilde{\psi}_k$,

$$P(\widetilde{\psi}_0 \le u) = u \tag{11}$$

so

$$P(\widetilde{\phi}(\widetilde{\psi}_0) \le x) = \widetilde{\phi}^{-1}(x) = \widetilde{\psi}(x) = 1 - \left(\frac{x}{\delta}\right)^{1-\gamma}, \tag{12}$$

which is to say, the numbers generated by applying the mapping $\widetilde{\phi}$ to the $\widetilde{\psi}$s will have a Pareto distribution with shape parameter $\gamma-1$ and minimum size δ. This implies that the flow lengths generated by Algorithm 1 will also have this distribution, as desired.

The reason these flow lengths have a Pareto distribution with shape parameter $\gamma-1$ rather than with shape parameter γ is that we only generate the flows of size τ which arrive in the interval $(-d(\tau),0)$. The density of the generated flow lengths has to be weighted by $d(\tau)$ or, in effect, by length, leading to the change of shape parameter. A plot of the sample distribution of flow sizes generated by Algorithm 1, gathered from a run of the implemented algorithm, is shown in Figure 4.

3.4 Adjustment of Long Flows to Compensate for Missing Short Flows

According to the plan outlined above, at each point in the simulation, e.g. t, a range (δ, κ_t) of shorter flow sizes are *not simulated* (in detail), but instead are simulated merely changing the service rate of the longer flows. Since flows of length τ have arrival times confined to the interval $(-d(\tau),0)$,

$$\kappa_t = d^{-1}(-t) = \frac{(1-\rho)Ct}{W}. \tag{13}$$

In order to match the original system as closely as possible it is desirable that the effective load at any point in the simulation is unchanged. We can do this by adding *extra bytes* to flows to compensate for the missing short flows. The utilization due to these flows shorter than κ_t is $\rho(\kappa_t)$. The effect of these flows, when they are present, is to slow down processing of other flows by the factor $\frac{1}{1-\rho(\kappa_t)}$. Hence, the extra bytes we should add to a flow of length τ which arrives at time $-t$ should be

$$\eta(t)\tau \quad = \quad \frac{\rho(\kappa_t)\tau}{1-\rho(\kappa_t)}.$$

The effect of adding these extra bytes is that flows arriving at time $-t$ are lengthened by the factor $\frac{1}{1-\rho(\kappa_t)}$, which is the same as the effect which would have been caused by the now missing shorter flows.

3.5 Simulation of more complex disciplines

The work presented here confirms the accuracy and efficiency of the snapshot simulation technique for the two disciplines SJF and FQ. Other queueing disciplines can, however, readily be incorporated into the simulation technique. The primary difference between a snapshot simulation and a conventional simulation is that in a snapshot simulation the selection of events to be simulated is based on the arrival time *and size* of the flows in question, whereas in a conventional simulation all flows which arrive in a certain interval of time are simulated. There is nothing obvious in the snapshot approach which prevents any model at all from being successfully simulated. The only complication is that flows which are *absent* (by choice) must be simulated by their *aggregate* impact on the longer flows which exist at the time.

In the disciplines which have been simulated in this paper, SJF and FQ, we have simulated the missing flows by adding *extra bytes* to the longer flows. The reasoning for this approach is that it ensures that the load of the model is consistent with the real system. It also ensures that the load due to flows in a specified interval of sizes (assuming the interval of sizes is above the threshold where flows are included) is consistent with the original model.

This approach works with other disciplines as well. One could create an artificial discipline which could not be satisfactorily simulated by this technique – for example, a discipline which artificially selected flows of certain sizes for special preferential, or prejudicial, treatment. However, until such disciplines arise for reasons other than merely to challenge our modeling assumptions, it seems safe to adopt the *extra bytes* philosophy for modeling the missing short flows.

For example, if we wished to take a snapshot approach to conducting NS2 simulations, we could select the flows to simulate using the snapshot approach, then add extra bytes to flows in accordance with their arrival times, to compensate for the missing shorter flows, and then submit the configuration to NS2 for simulation.

4. VALIDATION AND COMPARISON WITH ANALYSIS

The simulation software includes extensive internal validation tests, which can be turned off to speed up simulations once testing is complete. These tests mainly take the form of checks that bytes in arriving flows are fully represented in all stages of the simulation. The range of tests of this form is sufficient to give confidence that the simulation code is free from significant errors.

In addition, several approaches to validating the simulations by comparing the results generated to predictions generated independently have been undertaken: (a) we show that the generated flow lengths have the desired distribution; (b) we show that as $W \longrightarrow \infty$, the simulations converge to a limit; (c) we compare the results of our simulations with conventional simulations; (d) we compare with analytical results for models which have them; and, finally, (e) we can check that the simulation results converge to a limit as $\gamma \longrightarrow 1$. These different validation methods have been applied to both the FQ and the SJF queueing disciplines. All of these approaches have been taken, and in some cases combinations of these approaches. Space is not available for showing all of these results. For more details, see [19].

Conventional simulations of FQ and SJF systems of lengths 1,000 seconds, 10,000 seconds and 100,000 seconds are shown in Figure 1. Since $\lambda = 66$ flows per second were generated in all these simulations, the longest of these simulations comprised approximately 6.6 million flows. It took approximately 1 1/2 hours of computer time to complete. The convergence to theoretical results – a geo-

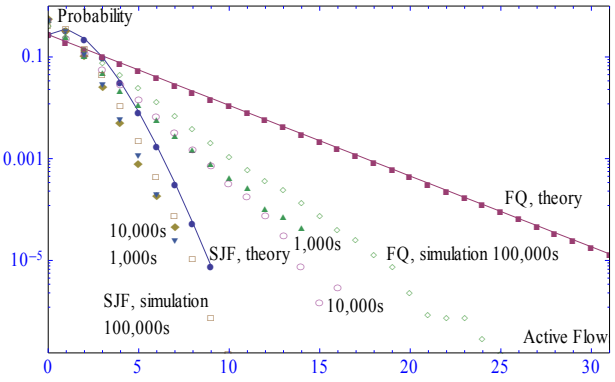

Figure 1: Convergence of conventional simulations to theoretical results. Parameters: $\gamma = 1.1$, $\lambda = 66$, $\delta = 1$, $C = 1000$.

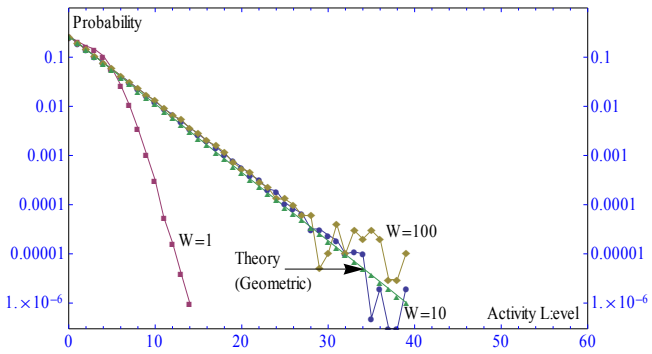

Figure 2: Activity Levels under FQ, for $W = 1$, 10, 100, 1000, **and comparison with theory. Parameters:** $\gamma = 1.1$, $\lambda = 66$, $\delta = 1$, $C = 1000$.

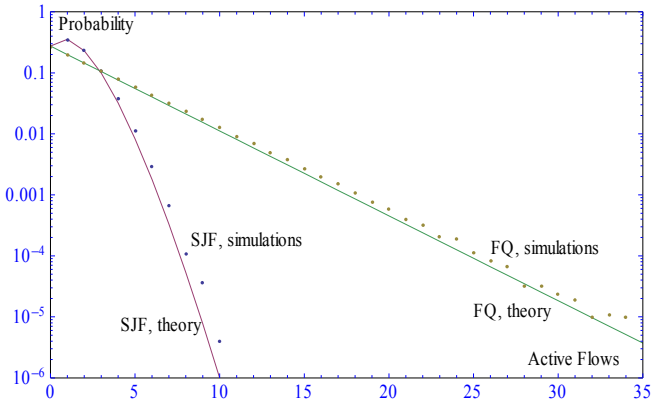

Figure 3: Activity Levels under SJF and FQ, as calculated by snapshot simulations and by analysis. Parameters: $\gamma = 1.1$, $\lambda = 66$, $\delta = 1$, $C = 1000$, $W = 100$.

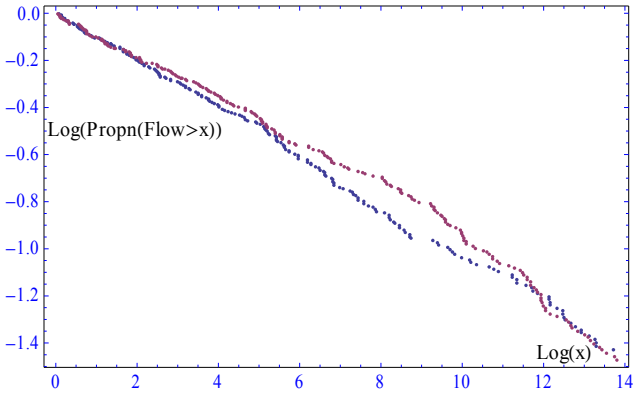

Figure 4: Sample Complementary Flow Distributions from short simulations. Parameters: $\gamma = 1.1$, $\lambda = 66$, $\delta = 1$, $C = 1000$.

metric distribution for the FQ discipline [8, 9] and a Poisson distribution for the SJF discipline [20] – is visible, but so slow that satisfactory results will not be obtainable for the model under consideration. This confirms a major contention of this paper, that conventional simulations cannot be used to estimate the performance of these models.

Convergence of snapshot simulation results to the theoretical result expected, as $W \longrightarrow \infty$, is shown in Figure 2. Only results for the FQ discipline are shown. It appears that satisfactory convergence has occurred by the time $W = 10$. The same observation applied to the SJF case. The fact that the results when $W = 100$ appear to be less accurate than the $W = 10$ runs is probably due to the fact that the *lengths* of these simulations were shorter. The lengths of the simulations shown in Figure 2 were as follows: $W = 1$: 1,000,000 runs; $W = 10$: 1,000,000 runs; and $W = 100$: 100,000 runs.

The SJF and FQ simulation results have been compared to analytical results for these systems as presented in [20] and the results are shown in Figure 3. The simulations required 6 minutes of processing time (for both at once). Conventional simulations of the same systems requiring more than 1 1/2 hours each, but still not providing satisfactory accuracy, are presented in Figure 1.

The flow lengths generated in some relatively short simulation runs (approximately 100 flows in total), were collected and their sample complementary distribution functions are shown in Figure 4. These are consistent with the expected Pareto distribution with parameter 0.1.

As a demonstration of the application of the snapshot simulation technique to queueing disciplines other than FQ and SJF, simulations of SJF-3, SJF-5, and SJF-10 were implemented and run. The SJF-n discipline serves the n largest flows in SJF order and the remaining flows are given priority over these flows and served in FQ order amongst themselves. The results are shown in Figure 5. These results show that in this particular case most of the benefits of the impractical SJF discipline can be achieved by discriminating against only the 3 longest flows!

5. CONCLUDING REMARKS

A new approach to simulating Internet traffic has been outlined and some experiments described. This approach is very fast, and converges to the stationary behavior of the system in a fraction of the time a conventional simulation would take. The reason for this is that the real system contains processes which operate at extremely diverse time-scales; the technique we use focuses on the impact of flows at a specific point in time; during periods of time far from the main focus, full detail of how shorter flows affect longer flows is not simulated. Each simulation experiment is very fast and can therefore easily be repeated many times. By this method, very accurate results may be obtained.

In the experiments described here only minimal details of Internet operations have been incorporated. However, there are no limitations on how much detail can be added to the simulation without fundamentally changing the algorithm. This will allow us to further investigate the significance of these details. In this paper, the simulations have been used to validated a mathematical model of

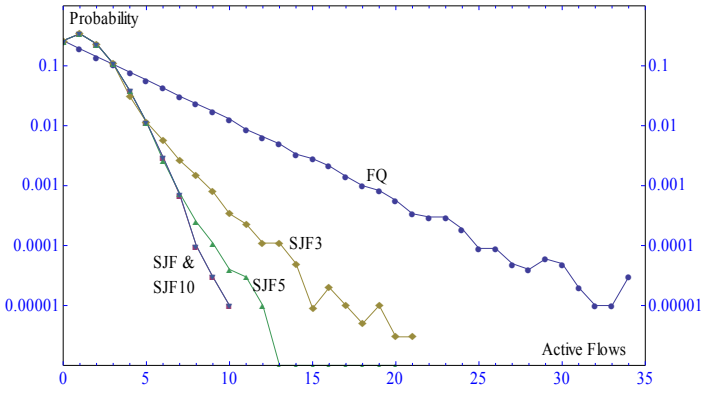

Figure 5: Simulations of SJF-n compared to SJF and FQ. Parameters: $\gamma = 1.1$, $\lambda = 66$, $\delta = 1$, $C = 1000$.

the activity level in Internet systems which implement the SJF or FQ queueing disciplines. The simulation technique used here can be used to investigate other queueing disciplines, e.g. disciplines which combine features of SJF and FQ to gain most of the efficiency of SJF without having to introduce much additional complexity, such as SJF-*n*. The results obtained in this paper already show that this particular queueing discipline is capable of delivering good performance despite its low complexity.

The snapshot simulation method is not able, in itself, to produce accurate estimates of probabilities of events which occur with probabilities lower than approximately 10^{-5}. This is because the snapshot simulations are natural simulations, so any such event would occur only in 1 in 100,000 simulations. However, there is no reason not to apply importance sampling techniques *in addition* to the snapshot approach. In this way it should be possible to produce better estimates of the probabilities of unlikely events.

Acknowledgements

I would like to thank Don McNickle for bringing my attention to important related work, in particular for noticing the relevance of queueing network results with the processor sharing discipline.

6. REFERENCES

[1] Nandita 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.

[2] Yang Guo, Weibo Gong, and Don Towsley. Time-stepped hybrid simulation (TSHS) for large scale networks. *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2:441–450 vol.2, 2000.

[3] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *IEEE/ACM Transactions on Networking*, 1(3), 1993.

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

[5] L. Schrage. A proof of the optimality of the shortest remaining processing time discipline. *Operations Research*, 16:678–690, 1968.

[6] R. G. Addie, S. Braithwaite, Jishu 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.

[7] 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.

[8] M. Sakata, S. Noguchi, and J. Oizumi. Analysis of a processor shared model for time sharing systems. In *2nd Hawaii Int. Conf. on System Sci.*, 1969.

[9] Frank Kelly. *Reversibility and stochastic networks*. Wiley series in probability and mathematical statistics. John Wiley, Chichester ; New York, 1979.

[10] ns users@isi.edu. The network simulator - ns-2. Web site, 2005. `http://www.isi.edu/nsnam/ns/`.

[11] 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.

[12] 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.

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

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

[15] Nicolas Hohn, Darryl 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.

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

[17] Liang Guo and Ibrahim 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.

[18] R. G. Addie. Snapshot simulation c code. Technical Report SC-MC-0804, University of Southern Queensland, January 2008. http://www.sci.usq.edu.au/research/workingpapers.php.

[19] R. G. Addie. Snapshot simulation validation and testing mathematica code. Technical Report SC-MC-0805, University of Southern Queensland, January 2008. http://www.sci.usq.edu.au/research/workingpapers.php.

[20] O. Yevdokimov R. G. Addie. Asymptotically accurate flow completion time distributions under fair queueing. In *Proceedings, the Australian Telecommunication Networks and Applications Conference 2007*, December 2007.