Snapshot Simulation of Internet Traffic: queueing of fixed-rate flows

Ron Addie
University of Southern Queensland
Toowoomba, 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 explore a representative range of starting states for a collection of simulations 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. This paper explores a technique previously applied to fair queueing and optimized queueing of bottle-necked flows, this time studying the buffering of an aggregation of flows with a common fixed rate (due to limitations of the source or elsewhere along the path of the flow). Direct and importance sampled simulations are validated and compared to an analytical model of the same system.

Categories and Subject Descriptors

I.6.5 [**SIMULATION AND MODELING**]: Model Development—*Modeling methodologies*; C.2.1 [**COMPUTER-COMMUNICATION NETWORKS**]: Network Architecture and Design—*Packet-switching networks* ; C.4 [**PERFORMANCE OF SYSTEMS**]: Modeling techniques

1. INTRODUCTION

An important challenge in modeling, analysis, and simulation of modern networks – wireless, optical, and the Internet – is the presence of processes and, in particular, flows, operating at widely divergent time-scales. Successful modeling of such systems has been a major challenge for physics, mathematics and engineering for many years. In particular, importance sampling, which is one of the most powerful approaches for speeding up simulations and improving accuracy, does not appear to be effective, by itself, in solving this problem.

An approach introduced in [1] successfully addresses this problem in the specific context of traffic flows in the Internet which share and compete for the resources of a link on the outgoing side of a router. In this paper we extend this technique methodologically, and to a different model. The model we consider in this paper is still traffic flows in the Internet which share and compete for the resources on a link, but in the case considered here, the flows all supply traffic with a constant rate. This model has been studied by many authors and the difficulties of simulation and analysis are well documented in this literature [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13].

Controversy concerning accurate modeling of this system continues to this day which effective simulations can help to resolve, but the fundamental difficulty of undertaking simulations of processes with widely divergent time scales has, up to this time, prevented such simulations from being undertaken. This problem appears quite directly in this paper because there are a large number of flows which have durations less than 0.1 second and also a small number of flows that have durations as long as an hour, and both types of flow are important, and need to be taken into account in the simulations. In a conventional simulation, in order to model the long flows the simulation will need to be very long, and in order to model the short flows it will need to be very detailed – this forces the simulation to be very slow to complete.

The importance sampling principle identified in [14] was applied to the same problem addressed in this paper, but with limited benefit, because it was not effective in improving simulation accuracy when applied to processes with divergent time scales. The problem of simultaneously simulating processes with different time scales is tackled in the present paper, and in [1], by simulating different time scales at different levels of detail.

The simulations carried out in this paper are significantly faster than traditional simulations. Conventional discrete event simulations (in which each event is simulated in order of its occurrence) of very similar systems reported in [2] required run times as long as 8 hours, and for particularly difficult parameter sets such simulations are unlikely to converge at all. The simulations for this paper were carried out on an early model Notebook PC with a 1.6 GHz Intel processor equipped with 2 Gigabytes of RAM. Most simulations, results of which are presented in this paper, required CPU

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5763
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5763

time less than ten minutes. There is no inherent restriction on the level of detail which *could* be included, although the simulations here include no more than basic details. Inclusion of more detail is a task for the future.

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. In the course of this exploration of potential events it will normally be necessary to consider events which occurred further and further *in the past* so, in a sense, the simulation proceeds predominately backwards in time, rather than forwards as we usually expect.

The reason this approach is much 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 this paper, as opposed to [1], the technique of multiple time-scale simulation is combined with the use of importance sampling. This has proved to be important because, in addition to the essential need to simulate multiple time scales – the majority of analytical solutions and approximations developed are asymptotically accurate for large buffers or other asymptotic regimes in which the probabilities under consideration are very small, so it will be a great advantage to be able to estimate probabilities of rare events.

In Section 2, the model of Internet traffic adopted in this paper is set out. In Section 3, the simulation algorithm is described. In Section 4, the methodology is described and techniques of validation are identified. In Section 5, Importance Sampling methods for this model are presented. In Section 6, results obtained using the simulation method of this paper are presented and some comparisons with analytical results for the same models are made. Concluding remarks are presented in Section 7.

2. SYSTEM MODEL

2.1 Internet Traffic Model

Internet traffic has been found to be *long-range dependent* and *self-similar* [15]. 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 [16, 17, 18].

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 [19] 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 X is Pareto distributed:

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

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

Different ways to measure the size of a flow, f, include: *total bytes* ($b(f)$), *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

$$d(f) = b(f)/r(f).$$

We assume that the Pareto distribution applies to the *size* of flows. In general, the *rate* at which a flow is served cannot be specified as a descriptive system parameter because it will depend upon the conditions which apply at the time. In an earlier paper [1] we considered the cases where the rate at which flows operate is controlled by a queueing discipline which mediates between all the flows which share a link. In this paper we focus on a different case: the case where all flows operate at the same, fixed, rate.

The reason why this case is important is that in the Internet and similar networks, now and in the future, the majority of links (or at least a large proportion) will not be sufficiently congested that the flows sharing the link are engaged in flow control procedures to a significant degree. Even though there are now, and will be for the forsee-able future, many links where congestion is occurring regularly and on which flows *are* actively engaged in congestion avoidance, nevertheless we need to understand the performance experienced by flows on links where congestion (leading to lost packets and congestion avoidance) is not a major contributing factor. The model in which flows all operate at the same fixed rate is a further simplification of this situation. The additional complexity of allowing flows to operate at individual distinct rates, and perhaps randomly varying rates, is also of interest, and is more realistic, but at the present stage of investigation of these models the simpler case is an appropriate model to consider since it allows us to tackle some of the essential difficulties of such systems.

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

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

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

$$\text{Prop}_\gamma = p^{\frac{\gamma}{\gamma-1}} \quad (3)$$

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5763
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5763

of the largest flows. The feature that a small number of large flows contain a remarkably large proportion of bytes is well established [20, 21].

In the following section we need to refer to the system utilization contributed by flows shorter than τ. This quantity, $\rho(\tau)$, can be evaluated, when the traffic model is as described above, as

$$\rho(\tau) = \frac{\lambda\gamma\delta}{C(\gamma-1)}\left(1 - \frac{\tau^{1-\gamma}}{\delta^{1-\gamma}}\right) \quad (4)$$

The proportion of time the server is occupied by flows *longer* than τ is denoted by

$$\widetilde{\rho}(\tau) = \frac{\lambda\gamma\tau^{1-\gamma}}{C(\gamma-1)\delta^{-\gamma}}, \quad (5)$$

and the overall processor utilization is denoted by $\rho = \rho(\infty) = \widetilde{\rho}(0) = \frac{\lambda\gamma\delta}{C(\gamma-1)}$.

2.2 Processing Model

The system under study includes a server, modeling a link, which operates at a fixed rate, C, together with an infinite buffer which is used to store bytes which have arrived but which cannot yet be served. Our focus is on the stationary distribution of the level in this buffer. The motivation is that we wish to understand whether, and if so, how, to choose the capacity of the link, C, and the size of the buffer, so that the probability of buffer overflow is quite small.

3. TECHNIQUE AND ALGORITHM

The flows arriving at a link 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*. A conventional simulation (in which events are simulated one-by-one in order of their occurrence, with future events generated by random or deterministic effects from past events) very often exploits the Poisson property of the flow arrival instants. The simulation algorithm here is based, instead, on the independence of the occurrence of flows of different sizes.

A conventional simulation develops a scenario by evolving in the same manner as the real system, through a sequence of epochs when significant events in the state of the link, and in particular its buffer, occur. This approach includes an inaccuracy, leading to bias, due to the fact that the system is initiated in a specific state (usually the idle state). In systems with multiple very divergent time scales this start-up bias can be virtually impossible to overcome by lengthening (warming up) the simulation.

A snapshot repeatedly simulates instantaneous *snapshots* (states) at a certain moment in time. Each such simulation is intended to provide an independent, typical, view of the state of the system being simulated.

Each snapshot is simulated by considering flows, successively, in order of length – shortest to longest – upwards along the y-axis of Figure 1 rather than left-to-right along the x-axis.

The method by which additional flows are added to the system is similar in spirit to the technique known as *unifomization* or *randomization* [22], whereby a continuous time Markov process is modelled by a Poisson point process together with a Markov chain. In our case, the flows which

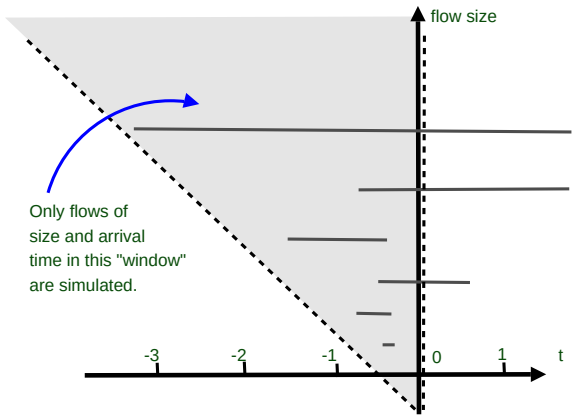

Figure 1: The window of simulated flows

arrive at the system form a non-homogeneous point process in the two-dimensional space spanned by time, on one axis, and flow size, on the other, as depicted in Figure 1. Normally, the configuration of arriving flows is explored by proceeding forwards in time, and generating the flow sizes *after* deciding on their arrival time. In our case, we will generate flows in increasing order of size and decide on the arrival time *after* choosing the *size*.

Since no two flows are ever *exactly* the same in length, generating them in increasing order of size is feasible. In addition, we will only consider flows which arrive sufficiently close, in time, to a particular time on which we are focussing our attention – the *observation time*, t_0. In the case of short flows, this arrival time must be quite close to t_0 for the flow to be *relevant*, whereas longer flows which arrive at more distant times, before t_0, may still be relevant. Since the system is stationary, the choice of snapshot time is not important and is assumed to be zero in the simulations, as shown in Figure 1, however we refer to this time also as t_0 in the text in order to bring attention to its role as the moment when the system is observed.

3.1 How to generate flows

Let $\epsilon > 0$ denote a small number (signifying a target level of accuracy). The selection of the length of the next longest flow is achieved by selecting a pseudo-random negative exponentially distributed random number, $\mathcal{P}$, with mean equal to 1. This number is used to select the *size*, ℓ, of the next flow to be added to the system, by using the mapping $\ell = \phi(\mathcal{P}, \sigma)$, where σ denotes the current *configuration* of flows in this snapshot.

A configuration of flows is nothing more or less than the number of flows, and the list of their lengths. In the present instance, the function ϕ actually depends only on the length of the longest flow in the configuration, but in a more general setting the dependence of ϕ on σ could be more complex.

The time of arrival of this next flow, f, relative to the snapshot time, is then selected immediately by choosing a random number from a uniform distribution over an interval $(t_0 - \omega(\sigma), t_0)$, where $\omega(\sigma)$ is a function chosen so that the

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5763
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5763

if the flow arrives at time $t_0 - \omega(\sigma)$, the probability that it is still active at time t_0 will be less than ϵ. It is also possible, when $\mathcal{P}$ exceeds a certain level, that no further flows will be added – simulation of the snapshot is complete. In the following defining property for ϕ, f denotes a flow in the snapshot, $\ell(f)$ its size, and $\tau(f)$ its arrival time. The mapping, ϕ, from probabilities, $\mathcal{P}$, to flow sizes is chosen so that for any interval $(p_1, p_2) \subseteq [0, \infty)$,

$$P\left(\mathcal{P} \in (p_1, p_2)\right) = E\big(\#\{f : \ell(f) \in (\phi(p_1), \phi(p_2)) \text{ and } \tau(f) \in (t_0 - \omega(\ell), t_0)\}\big). \quad (6)$$

In order to fully define the simulation, it remains to define the functions , $\omega(\ell)$, $\phi(\mathcal{P}, \sigma)$, and $\mathcal{E}(\sigma)$, in which σ is the current configuration of simulated flows, at a certain stage in the simulation, $\mathcal{P}$ denotes the pseudo-random number determining the characteristics of the next flow, and $\mathcal{E}(\sigma)$ is the expected additional number of flows to be simulated, when the current configuration is σ. In all the cases considered in this paper as σ changes through a succession of configuration states, σ_0, σ_1, ..., $\mathcal{E}(\sigma)$ changes according to the formula $\mathcal{E}(\sigma_{n+1}) = \mathcal{E}(\sigma_n) - \mathcal{P}$, however it is conceivable that we may wish to develop flow configurations in a more complex manner which deviates from this simple principle. In this case the number of flows generated will probably no longer be Poisson, however the Markov property, that the probability distribution of the configuration of flows added is well-defined given the current configuration, will remain. This is all that we need.

The definition of $\mathcal{E}(\sigma)$, $\omega(\ell)$, and $\phi(\mathcal{P}, \sigma)$ depends upon the queue discipline and the traffic model. In the previous paper [1] on this simulation technique the SJF queueing discipline and FQ queueing disciplines were considered, whereas in this paper we consider the case where all flows deliver bytes at the same fixed rate, r, and when more bytes are delivered than the link can cope with, the excess are stored in a buffer.

3.2 The Simulation Algorithm

The broad details of the simulation algorithm do not depend on the queue discipline and are as follows:

Algorithm 1 A Snapshot Simulation Algorithm

```
N = 1000 {Number of snapshots in total}
for i ← 0; i < N; i ← i + 1 do
    E ← E(number of flows in the window)
    k = 0 {Next flow is number k}
    for P ← exprandom(1); P < E; P ← exprandom(1) do
        ℓ_k = φ(P, σ); τ_k = uniformrandom(−ω(ℓ_k), 0);
        σ = σ ∪ {ℓ_k}; k = k + 1;
    End of construction of a simulation configuration, σ
    Calculate level in buffer
    Accumulate results (histogram of buffer levels)
End of simulation
```

Flows have length which is Pareto distributed, as at (1), so the utilization of this system must be $\frac{\lambda\delta\gamma}{(\gamma-1)C}$. Note that r does not appear since the Pareto distribution is for the flow *size*, not its duration, as was the case in [23].

Suppose the flows simulated up to now are k in number, arrived at times τ_0, ..., τ_{k-1}, and were of lengths $\ell_0 > \ell_1 > \cdots > \ell_{k-1}$ (in bytes). The maximum length of these flows is $\ell_{\max} = \ell_{k-1}$. Let $\tau_{\min}$ denote the earliest of the arrival times of these flows (not necessarily τ_{k-1}).

We now determine the functions $\omega(\ell)$, $\phi(\mathcal{P}, \sigma)$ and $\mathcal{E}(\sigma)$ for this case. The duration of a flow is determined as soon as we know its size; a flow of size ℓ will have duration ℓ/r.

We now introduce a parameter $W \in [1, \infty)$ which is used to trade off between accuracy and speed. When W is very large no details are ommitted in the simulation. When $W = 1$, only the flows which are certain to be still present at time t_0 are simulated.

So, with $W \gg 1$ and at least $W \geq 2$, we define the window of simulated flows, as those of length ℓ, which arrive in the interval $(t_0 - W\ell/r, t_0)$. This implies, in particular, that $\omega(\ell) = -W\ell/r$, i.e. if the last generated flow was of length ℓ, it's arrival time will be uniformly generated over the interval $t_0 - W\ell/r$ to t_0. As discussed earlier, W shall be varied to determine if the simulations converge to a well-defined behaviour as $W \longrightarrow \infty$. The expected number of flows of length greater than ℓ_{k-1} which need to be taken into account is

$$\begin{aligned} E_k(\sigma) &= \int_{\ell_{k-1}}^{\infty} \frac{\lambda\gamma}{\delta}\left(\frac{\ell}{\delta}\right)^{-\gamma-1}\left(\frac{W\ell}{r}\right) d\ell \\ &= \left(\frac{W\lambda\gamma\delta}{r(\gamma-1)}\right)\left(\frac{\ell_{k-1}}{\delta}\right)^{1-\gamma}. \end{aligned} \quad (7)$$

In particular, adopting the convention that $\ell_{-1} = \delta$, which makes sense here, this equation gives us the formula $E_0 = \frac{W\lambda\gamma\delta}{r(\gamma-1)}$, for the expected number of flows in a snapshot.

Set $P_k = \sum_{j=0}^{k-1} \mathcal{P}_j$. The next longest flow should have length, ℓ, defined by the equation

$$\begin{aligned} \mathcal{E}_0 - P_k &= \int_{\ell}^{\infty} \frac{\lambda\gamma}{\delta}\left(\frac{x}{\delta}\right)^{-\gamma-1}\left(\frac{Wx}{r}\right) dx \\ &= \left(\frac{W\lambda\gamma\delta}{r(\gamma-1)}\right)\left(\frac{\ell}{\delta}\right)^{1-\gamma}, \end{aligned} \quad (8)$$

hence

$$\begin{aligned} \mathcal{E}(\sigma) &= \left(\frac{W\lambda\gamma\delta}{r(\gamma-1)}\right)\left(\frac{\ell_{k-1}}{\delta}\right)^{1-\gamma}, \\ \phi(\mathcal{P}_{k-1}, \sigma) &= \delta\left(\frac{r(\gamma-1)(\mathcal{E}_0 - P_k)}{W\lambda\gamma\delta}\right)^{\frac{1}{1-\gamma}}, \end{aligned}$$

and

$$\omega(\ell) = W\ell.$$

3.3 Discrete Event Simulation of a Snapshot

Once a snapshot has been assembled, by selecting which flow lengths will occur in the vicinity of this moment, and when these flows will arrive, we need to deduce the details which are of interest about this situation about which we wish to collect statistics. In the case of the simulation of flows with fixed rates, the parameter of primary interest is the level in the buffer of the server.

A completely general technique which we can make use of is to sort the flow events – arrivals and departures – into the order of their occurrence in time, and *simulate* their impact on the system in the usual manner of a discrete event simulation. It might be expected that for most systems there is a shortcut which enables us to deduce the important details of the system at time t_0 without going to all the trouble of a discrete event simulation. In the case of the simulations considered in [1] this was the case. In the present instance

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5763
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5763

discrete event simulation of the events identified by means of Algorithm 1 would be a reasonably straightforward approach. However, the technique we have used is to sort the flow arrival and departure events in reverse-time order and compute the implied buffer level, at time t_0, due to the flows between each of these moments and t_0, one-by-one, as shown in Algorithm 2, and then take the maximum of these as the actual buffer level. In effect, we have used Reich's equation instead of discrete event simulation to deduce the buffer level implied by the flow configuration.

Algorithm 2 Calculation of Buffer Level Implied by a Flow Configuration

```
1: {Sort flow arrival and completion events in reverse time order}
2: buff ← 0; maxbuff ← 0; τ ← 0
3: for f ← each flow event before 0 do
4:    buff ← buff+ arriving bytes between τ_f and τ - servable bytes between τ_f and τ
5:    if buff > maxbuff then
6:       maxbuff ← buff
7:    τ ← τ_f;
   End of loop through flow events
```

4. METHODOLOGY AND VALIDATION

The key idea for allowing short time-scale and long time-scale processes to be simulated simultaneously is to find a way to summarize (rather than simulate in full detail) the short time-scale processes which occur during the more remote decades and centuries of the evolution of the long time-scale processes. In order to define a clear, rigorously justified methodology we need to ensure that unbiased (or boundedly biased) estimates of all quantities of interest in the real system may be deduced from the simulation with reduced detail.

4.1 Asymptotic Bias

The term *bias* must, of necessity, refer to some specific numerical feature, or perhaps *list* of such features, the mean value of which we wish to ensure is unchanged. In the present instance, the mean values we are concerned about are the proportion of instances when buffer levels are above x, for $x \geq 0$. The detail we intend to omit is the arrivals and departures of the *short* flows whose entire existence is over before the moment, t_0, when the measurement of the buffer level takes place.

The approach adopted to avoid introducing bias by omitting details of these short flows is as follows:

- a constant load is used in place of the aggregation of all omitted short flows – in other words, these flows are replaced by a single flow with the same *mean* impact on other flows; also, in *some* simulations, we ensure that all short flows which arrive during the busy period which includes t_0 are simulated in detail;
- the proportion of short flows of which details are omitted is controlled by a parameter to allow the bias of this distortion to be reduced arbitrarily.

Since replacement of short flows by their mean load will consistently have the effect of reducing variation, we cannot expect all observations to be unbiased by this technique, however we can reasonably assume that the amount of bias introduced reduces as we increase the proportion of short flows which are fully simulated, and hence we can estimate and minimize the bias.

4.2 Confidence Intervals

Each snapshot provides an independent estimate of all of $P(Q > x)$, for each value of x. These estimates are combined together simply by taking their mean, and hence these estimates have a (scaled) Binomial distribution, with standard deviation $\sqrt{P(Q > x)P(Q \leq x)/n}$, where n is the number of simulations. So, if the estimate of $P(Q > x)$, from the simulations, is p_x, an estimate of the standard deviation of this estimate is $\sqrt{p_x(1 - p_x)/n}$. Confidence intervals of twice this estimate of the standard deviation above and below the estimated probability distribution value are included in some of the results in the next section.

4.3 Validation

In all simulations which make use of importance sampling the difficulty of establishing that the simulation is faithful to the original system is much greater. The importance sampling (and the snapshot simulation technique) increase the complexity of the simulation by means which are neither direct nor natural. There is a high likelihood that some of these additional complexities which are being added to the system under study will distort its operation in an unintended and unknown manner, thereby nullifying any potential insight it might provide into the original system.

It is essential, therefore, to incorporate simple, reliable systems for validating the operation of the simulation. For example, it would be highly advantageous if we could observe, in detail, a small subset of all flows in a manner which enables us to confirm that all aspects of their treatment are consistent with the original system.

Three important and basic techniques employed to ensure that simulations are valid are:

- detailed observation of particularly simple special cases (in which the number of flows expected is very low, for example;
- monitoring easily observed parameters of system behaviour with known expected values – in particular, mean number of flows, the shape parameter of flows (estimated from the sample mean of the *logs* of flow lengths divided by δ, and system utilization; and
- undertaking simulations with the characteristic snapshot features turned *off*.

The key technique used in this paper, to ensure that simulations are valid despite the characteristically important feature of omission of detail, is to undertake a sequence of simulations with successively more detail included. The progression of these simulations will then demonstrate if our objective of asymptotically unbiased simulation has been achieved. Details are progressively added to the simulations by increasing the size of the *window* of detail which is included in the simulations, as depicted in Figure 1, and discussed further in the next Section. The *size* of the window of flows which are simulated in full detail is proportional to a parameter W which, broadly speaking, determines the slope

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5763
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5763

of the dashed line on the left of the window, as it appears in Figure 1, and hence determines the *area* of this window.

4.4 Validation Experiments

1. A facility is available in the simulations to keep a complete log of all flows. This can be used to manually check that simulations are consistent with expectations. This is only feasible when the model under study generates relatively simple patterns of flows, however this is not difficult to engineer. A suitable model is obtained by setting $\lambda = 1$, $\delta = 1$, $\gamma = 2$, $r = 1$, $C = 4$, and $W = 1$. The average number of flows in each snapshot should then be $\frac{W\lambda\gamma\delta}{(\gamma-1)r} = 2$ and the system utilization should be 50%.

   Here follow ten flow configurations generated by the snapshot simulation software for this model. Each flow is described, in this list, by two numbers – time of arrival, and length in bytes:

   ```
   flows = {{{-4.50631,20.6293},{-0.496365,1.82168},
   {-0.197854,1.10582}}, {{-2.42612,3.59136},
   {-0.64676,1.0741},
   {-0.297366,1.03101}}, {{}}, {{}},
   {{-0.623736,2.6715},{-0.575498,1.20028},
   {-0.0488581,1.69582}},
   {{-0.481397,1.45677},{-0.067689,2.17242}},
   {{}}, {{}},
   {{-2.95061,11.1781},{-0.687318,6.24069}},
   {{-0.658381,1.32597},{-0.0846115,1.27688},
   {-0.0828626,1.73089}}, };
   ```

   Collecting and inspecting this log of the flows in a simulation provides a basic sanity check on the simulation and, more critically, allows a check to be made on the feature which minimizes bias due to omission of detail regarding shorter flows – the replacement of these omitted flows by their aggregate mean.

2. The mean number of flows in a snapshot, the mean number of *active* flows in a snapshot, the average system utilization, and an estimate of the shape parameter, γ, are reported by the simulation software at the end of each run. These have been checked, for various values of the parameters, and used to validate the implementation. Note: the mean flow length in a snapshot is often (whenever $\gamma < 2$) infinite, so checking this parameter can only be undertaken in a subset of simulations.

3. Because the snapshot simulation technique relies on *omission of detail*, and compensation for the omitted details is provided by adjusting the *means* of the quantities in the system which are not omitted, a particularly important feature to check is the *mean load* as a function of time, in each simulation, or collection of simulations of a particular class. The mean load should be approximately constant over time (i.e. the times preceding the observation time) and equal to the target load value, for all choices of W.

   Arguably, this is the most critical test of an implementation of the snapshot simulation method.

   In order to carry out this test, optionally, during simulations, a record is kept of the mean load arriving during the simulation *tabulated by its time of arrival*. In this way, as well as checking that the simulated load is precisely as intended overall, it is possible to check that no errors in implementation of the snapshot simulation method are introducing distortion of the model.

   A plot of the mean load from time t to t_0, as a function of t, for collections of simulations with a range of values for W is shown in Figure 4. Convergence to the appropriate value of load, which is -2.5 in this case (negative because the server is included as a negative load), is in evidence in this diagram. The amount of variation in the sample mean load curves in is quite high, but it needs to be kept in mind that each curve is a result of a collection of simulations and so residual randomness is expected. It should be possible to determine how much randomness *should* remain in these curves, and thereby undertake a rigorous test, however this has not been undertake at this stage.

4. Comparison with conventional simulation. By undertaking simulations with $\gamma \gg 2$ it should be possible to confirm that the snapshot method produces similar results for parameter settings where conventional simulation is effective. In addition, if simulations with $\gamma \approx 1$ are undertaken, it should be possible to illustrate that the snapshot method is more accurate for models where the multiple time scale problem is severe. A conventional simulation has been prepared, however no comparative simulations have been generated and compared at this stage. This remains as work for the future.

5. IMPORTANCE SAMPLING

Importance sampling *by itself* is only marginally effective in improving the accuracy and speed of simulations of a Poisson Pareto Burst Process queueing system, as evidenced in [14]. However, if the problem of multiple time scales (the problem that both very long and very short flows have to be correctly modeled) can be tamed, for example by the snapshot technique discussed in this paper, importance sampling has the potential to have a very significant beneficial impact on speed and accuracy of simulations.

The basic idea is to distort the simulation so that it visits the region of the state space of most interest much more often, and to correct for the distortion by an adjustment to the weight attributed to observations. There are two obvious parameters which can be altered in order to introduce desirable distortion of the simulated model: λ and γ.

5.1 Distortion of λ

The effect of increasing λ, to Λ for example, is to increase the rate at which flows arrive, and thereby to increase congestion. Suppose an observed configuration has flows $f_0, \ldots, f_{k-1}$ of lengths $\ell_0, \ldots, \ell_{k-1}$ and arrival times $\tau_0, \ldots, \tau_{k-1}$. The correction to the weight attributed to this event is the ratio of the density of the probability measure at this event in the true model to the density of the distorted probability measure at this event, which is just

$$\frac{\left(\frac{W\lambda\gamma\delta}{(\gamma-1)r}\right)^k e^{-\frac{W\lambda\gamma\delta}{(\gamma-1)r}}}{\left(\frac{W\Lambda\gamma\delta}{(\gamma-1)r}\right)^k e^{-\frac{W\Lambda\gamma\delta}{(\gamma-1)r}}} = \left(\frac{\lambda}{\Lambda}\right)^k e^{-\frac{W(\lambda-\Lambda)\gamma\delta}{(\gamma-1)r}} \qquad (9)$$

5.2 Optimal Distortion

If a distorted value of λ greater than the true value of λ is used in a simulation, which is usually the case, but the number of flows generated happens to be rather low, it can occur

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5763
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5763

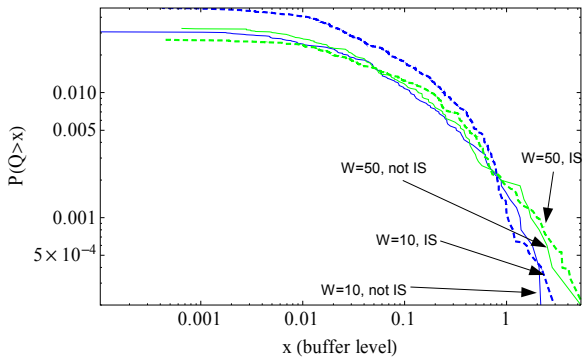

Figure 2: Importance sampling simulations compared to undistorted simulations. Parameters: $\gamma = 1.5$, $\lambda = 1$, $\delta = 1$, $C = 4$, $W = 10, 50$.

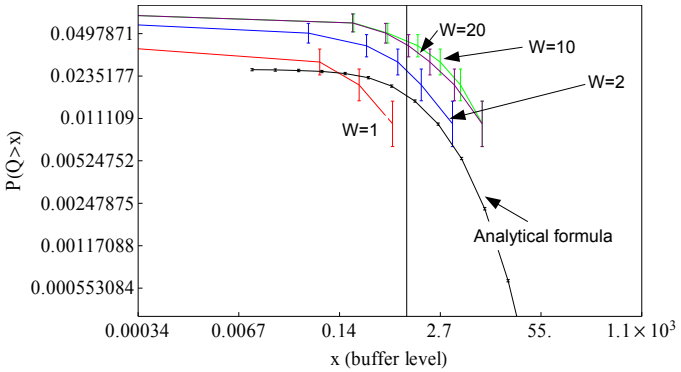

Figure 3: Snapshot simulations of systems with increasing W**. Parameters:** $\gamma = 1.5$, $\lambda = 10$, $\delta = 1$, $r = 1$, $C = 40$.

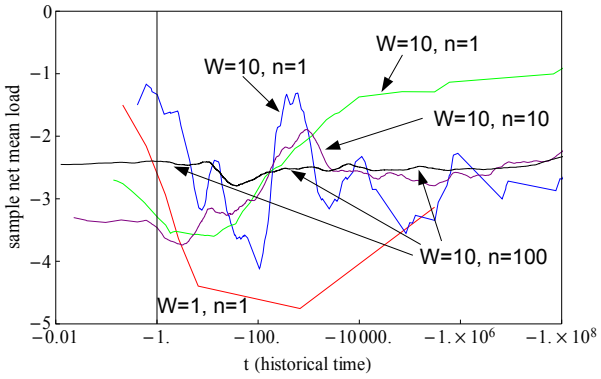

Figure 4: Sample mean load from time t **to** 0**, as a function of** t **. Parameters:** $\gamma = 1.2$, $\lambda = 1$, $\delta = 1$, $C = 8.5$, $W = 1$, 10, n **(number of simulations)** $= 1$, 10, 100.

that the probability associated with this event, computed using the Radon-Nikodym derivative above, will be greater than 1. This is due to this particular simulation generating information which is unbiased, but highly inaccurate, for the probability it is seeking to quantify. Ideally we should estimate the accuracy of each simulation and discard those that contribute a tiny amount to overall accuracy. Fortunately, the examples where this occurs are fairly obvious, due to the probabilities lying above 1, for example.

Each snapshot simulation may be regarded as an individual, separate estimate of the probability of the buffer in the system under study exceeding a certain level. Simulations with highly distorted parameters provide estimates for remote events and simulations with more normal parameters for more likely events. As a consequence, it is best to combine a collection of such simulations together to produce an estimate of a complete stationary probability distribution of buffer level. The importance-sampled simulations shown in Figure 2 have been conducted by randomly generating distorted values of λ from a normal distribution centered somewhat above the true λ.

6. RESULTS

6.1 Convergence as $W \longrightarrow \infty$

As discussed in Section 4.4, an important test of the simulation technique is testing convergence of simulations as $W \longrightarrow \infty$. Tests of a system in which $\lambda = 10$, $\delta = r = 1$, $\gamma = 1.5$, $C = 40$, and $W = 1$, 2, 10, and 20 are shown in Figure 3. Each of these simulations comprised 10000 snapshots. The confidence intervals shown span two standard deviations (as explained in Subsection 4.2, using the estimated value of the distribution) above and below the estimated distribution value. Convergence as $W \longrightarrow \infty$ appears to be present in this succession of examples; it is pleasing that convergence appears to occur for quite modest values of W, which confirms that the snapshot technique can be used to provide good accuracy with greatly reduced computation time. The longest of the simulations reported in Figure 3, in which $W = 20$ and $n = 10,000$ (this is the number of snapshots) took 19 minutes of CPU time (estimated using the Unix `time` command) on a somewhat dated Toshiba notebook computer with a single core Pentium processor running at 1.4 GHz. The shortest (with $W = 1$, $n = 10000$) took 11 seconds.

6.2 Comparison with Analysis

The analytical result from [23] for a system in which flows are delivered at a fixed rate are also included in Figure 3. The degree of similarity between simulation and analytical results is satisfactory, at this stage of the development of the analytical model and of the snapshot simulation technique, although there is clearly a discrepancy. A simple explanation fopr this is that in the analytical result the short flows are approximated by a Gaussian process, however this Gaussian model will be inaccurate when the flow intensity is low, as in this example (10 arrivals per second).

7. CONCLUDING REMARKS

A new approach to simulating Internet traffic has been outlined and some experiments described. The new approach appears to converge to the stationary behaviour of the system quickly. The real system contains processes which operate at extremely diverse time-scales (hours and milliseconds); the technique we use focuses on the impact of flows

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5763
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5763

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, accurate results may be obtained, as shown above.

A powerful and intriguing technique for extending the range of systems that can be investigated by means of simulation has been developed and explored.

8. REFERENCES

[1] R. G. Addie. Snapshot simulation of internet traffic: fast and accurate for heavy-tailed flows. In *Proceedings of the 1st International Workshop on the Evaluation of Quality of Service through Simulation in the Future Internet*. IEEE, March 2008. http://www.sci.usq.edu.au/research/workingpapers.php.

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

[3] N. G. Duffield. Queueing at large resources driven by long-tailed M/G/∞-modulated processes. *Queueing Systems*, 28(1–3):245–266, May 1998.

[4] M. M. Krunz and A. M. Makowski. Modeling video traffic using M/G/∞ input processes: A compromise between Markovian and LRD models. *IEEE Journal on Selected Areas in Communications*, 16(5):733–748, June 1998.

[5] Nikolay Likhanov and Ravi R. Mazumdar. Loss asymptotics in large buffers fed by heterogeneous long-tailed sources. *Advances in Applied Probability*, 32(4):1168–1189, 2000.

[6] N. Likhanov, B. Tsybakov, and N. D. Georganas. Analysis of an ATM buffer with self-similar ("fractal") input traffic. In *Proceedings of IEEE INFOCOM '95*, pages 1–15, April 1995.

[7] M. Mandjes. A note on queues with M/G/∞ input. *Operations Research Letters*, 28(5):233–242, 2001.

[8] M. Mandjes and S. Borst. Overflow behavior in queues with many long-tailed inputs. *Advances in Applied Probability*, 32(4):1150–1167, 2000.

[9] Minothi Parulekar and Armand M. Makowski. Tail probabilities for M/G/∞ input processes (i): preliminary asymptotics. *Queueing Systems - Theory and Applications*, 27(3-4):271–296, 1997.

[10] B. Tsybakov and N. D. Georganas. Self-similar traffic and upper bounds to buffer-overflow probability in an ATM queue. *Performance Evaluation*, 32(1):57–80, February 1998.

[11] Boris Tsybakov and Nicolas D. Georganas. Overflow and losses in a network queue with a self-similar input. *Queueing Systems*, 35(1-4):201–235, 2000.

[12] B. Zwart, S. Borst, and M. Mandjes. Exact asymptotics for fluid queues fed by multiple heavy-tailed on-off flows. *Annals of Applied Probability*, 14:903–957, 2004.

[13] R. G. Addie, T. D. Neame, and M. Zukerman. Performance analysis of a poisson-pareto queue over the full range of system parameters. November 2008. Accepted for publication in Computer Networks.

[14] R. G. Addie. Quantum simulation - rare event simulation by means of cloning, thinning and distortion. *European Transactions on Telecommunications*, 13(4):387–397, 2002.

[15] 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.

[16] 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.

[17] R. G. Addie, M. Zukerman, and T. D. Neame. Fractal traffic: Measurements, modelling and performance evaluation. In *Proc. IEEE INFOCOM '95*, volume 3, pages 977–984, April 1995.

[18] 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.

[19] 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.

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

[21] 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.

[22] D. Gross and D. R. Miller. The randomization technique as a modeling tool and solution procedure for transient markov processes. *Operations Research*, 32(2):343–361, Mar.-Apr. 1984.

[23] R. G. Addie, T. D. Neame, and M. Zukerman. Performance evaluation of a queue fed by a Poisson Pareto burst process. *Computer Networks*, 40(3):377–397, October 2002.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5763
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5763