Inferring Traffic Burstiness by Sampling the Buffer Occupancy

Michel Mandjes[1] and Remco van de Meent[2]

[1] CWI, Amsterdam, Netherlands
michel@cwi.nl
[2] Univ. of Twente, Enschede, Netherlands
r.vandemeent@utwente.nl

Abstract. Common practice to determine the required bandwidth capacity for a network link is to measure the 5 minute average link load, and then add a safety margin to cater for the effect of burstiness on small time-scales. Because of the substantial measurement efforts required to determine the burstiness, network managers often rely on rules of thumb to find the safety margin, e.g. 'mean plus 50%'. In this paper we propose a novel method to accurately determine the burstiness of traffic on small time-scales, *without* requiring measurements on such small time-scales. Our method is based on coarse-grained polling of the occupancy of a buffer in front of the link, from which the burstiness on small time-scales is inferred. We provide the theoretical foundations of our approach, and a validation through both simulation using synthetic traffic as well as real network traffic taken from various operational networks. It turns out that using our approach, it is possible to accurately determine burstiness on small time-scales (for instance 10 ms), by sampling the buffer occupancy (for instance) every second.

Keywords: Network design and capacity planning, queueing network models.

1 Introduction

Provisioning of network resources addresses the interrelationship between: (i) offered traffic (in terms of both average load and burstiness), (ii) desired level of performance, and (iii) the required capacity. Generally, more capacity is needed when offered load and burstiness increase, or when the performance criterion becomes more stringent. To operate a network in a viable way, provisioning procedures balancing (i), (ii), and (iii) are required: scarce provisioning inevitably leads to performance degradation, whereas (too much) over-provisioning results in a waste of resources.

Provisioning procedures are commonly based on rules of thumb. Considering a time window in which network traffic can be assumed stationary, one often uses rules of the type 'the mean traffic rate, plus a margin of 50%'. Obviously, such a fixed margin is not universally applicable. This motivates the use of formulae of the type

$$C = M + \alpha\sqrt{V} \tag{1}$$

for mean load M, some factor α (reflecting the performance target, which is mostly determined by the applications involved), and a standard-error term $\sqrt{V}$ to account for

R. Boutaba et al. (Eds.): NETWORKING 2005, LNCS 3462, pp. 303–315, 2005.

traffic fluctuations, see for instance [1]. Hence, the size of the 'safety margin' is affected by the chosen performance target, and the burstiness of the offered traffic. From (1) we conclude that provisioning (for a given performance target) requires knowledge of both M and V.

The mean traffic rate M can be determined by standard coarse-grained traffic measurements. A common way is to poll Interfaces Group MIB counters via the Simple Network Management Protocol (SNMP) every 5 minutes; this yields the total amount of traffic sent through the network interface over this time-interval. Determining the burstiness V, on the other hand, is more involved, and is essentially the subject of this paper.

Let $A(t)$ denote the traffic offered over a window of length $t > 0$, and the function $V(t) := \mathrm{Var}A(t)$. Then the analysis in this paper indicates that there is some 'dominant time-scale' T, such that the (burstiness) V in formula (1) corresponds to $V(T)$; hence it is this $V(T)$ that needs to be estimated. Note that the time-scale T is usually relatively small (for instance in the order of 100 ms, and often even considerably less). Clearly, $V(T)$ can be estimated directly by doing measurements on time-scale T, and computing the sample variance. However, it is hard to do accurate measurements on these small time-scales; they are hardly feasible through SNMP. This motivates the need for alternative, 'cheap' measurement techniques for determining the traffic variance on small time-scales.

Contribution. As argued above, it is of crucial interest to develop methods for efficiently and accurately estimating $V(T)$ for small time-scales T. The main contribution of this paper is that we propose an alternative for 'direct estimation' (i.e., by doing measurements on the time-scale of interest). In our approach the (i) *buffer occupancy is polled* on a regular basis (for instance every 10 seconds), and (ii) subsequently we 'invert' the resulting (estimated) buffer content distribution to the variance function $V(\cdot)$. Importantly, this approach eliminates the need for traffic measurements on small time-scales. In this sense, we remark that our proposed procedure is rather counterintuitive: without doing measurements on time-scale T, we are still able to accurately estimate $V(T)$. In fact, one of the attractive features of our 'inversion procedure' is that it yields the *entire* 'variance curve' $V(\cdot)$ (of course up to some finite horizon), rather than just $V(T)$ for some pre-specified T.

Approach and organization. The variance estimation technique proposed in this paper relies on the assumption that traffic is (fairly) Gaussian: for any $t \geq 0$, the amount of traffic offered in a time window of length t is accurately described by a normal distribution, parameterized by a mean Mt and variance $V(t) := \mathrm{Var}A(t)$. This Gaussianity was observed in various measurement studies. Kilpi and Norros [2] have statistically verified that the use of Gaussian traffic models is justified as long as the aggregation is sufficiently large (both in time and number of flows), due to Central Limit type of arguments. Importantly, Gaussian models cover long-range dependent processes, such as fractional Brownian motion (fBm); traffic measurements in the 1990s showed that in various situations this fBm model accurately models network traffic, see, e.g., [3].

Section 2 presents preliminaries on Gaussian queues. Particular attention is paid to provisioning formulae (for both buffer and bandwidth), in line with (1). The provisioning formula motivate the need for methods to estimate the variance function $V(\cdot)$, i.e., $V(t)$ as a function of the interval length t. The formulae for Gaussian queues lead to our

efficient method for estimating the $V(\cdot)$, that is explained in Sect. 3; importantly, we derive an 'inversion formula' that yields $V(\cdot)$ from the empirically determined buffer content distribution. In Sect. 4 we describe the inversion procedure, and demonstrate it by using synthetic traffic (in this case fBm traffic).

In the inversion procedure, we have identified three sources of possible errors, viz.: the (large-deviations) asymptotics give an *approximation* of the overflow probability (rather than an exact formula), the buffer content distribution is *estimated* through the buffer polling procedure, and we *assume* that traffic is (accurately approximated by) Gaussian. In Sect. 5 we present a detailed, quantitative study of the impact of each of these errors on the resulting estimation of the variance.

To investigate the applicability of the procedure in practice, we have performed extensive numerical experiments with real data from various real-life settings. These networks have different access technologies and link speeds, and different applications and user populations. We present and discuss the results of this experimental validation in Sect. 6.

Section 7 presents a number of reflections of the feasibility of implementing our estimation procedure, and concludes that there are no conceptual impediments. Section 8 concludes, and and lists a number of possible directions for future work.

2 Gaussian Queues – Motivation

In this section we review some basic principles of Gaussian traffic, and recapitulate the main fundamental (large-deviations) theory for queues with Gaussian input. Then we derive, for this Gaussian setting, a number of dimensioning rules. These formulae motivate the need for estimating specific traffic characteristics, viz., the mean rate M and the variance function $V(\cdot)$, cf. provisioning rule (1).

Preliminaries on Gaussian queues – many-sources asymptotics. Consider n independent, statistically identical *Gaussian* sources. It is assumed that the traffic pattern generated by an individual source corresponds to a Gaussian process with stationary increments. This type of sources is characterized by their mean traffic rate μ, and their variance function $v(t)$, for $t \geq 0$. With $A_i(t)$ denoting the amount of traffic generated by the ith source in an interval of length $t \geq 0$, then $\mathbb{E}A_i(t) = \mu t$, and $\mathrm{Var}A_i(t) = v(t)$.

Now suppose that the n sources feed into a queue with capacity C, and apply the scaling $C \equiv nc$. It is well-known that the stationary queue length, say Q_n has the same distribution as the maximum of the corresponding 'free-buffer process': $Q_n \stackrel{\mathrm{d}}{=} \sup_{t>0} \left(\sum_{i=1}^{n} A_i(-t,0) - nct\right)$. The following fundamental result is found in, e.g., [4]:

Lemma 1. *Suppose that there is an $\alpha < 2$ such that $v(t)/t^{\alpha} \to 0$ for $t \to \infty$. Then, for any $b > 0$, and $c > \mu$:*

$$I(b) := -\lim_{n\to\infty} (1/n) \log \Pr(Q_n \geq nb) = \inf_{t>0} (b + (c-\mu)t)^2/(2v(t)) \ .$$

The above result holds for the system 'scaled by n', but gives rise to an approximation for the 'unscaled' situation, see also for instance [4], [5–Eq. 3]. With $B \equiv nb$, consider the probability that the buffer content Q exceeds B. Denote $M \equiv n\mu$ as the aggregate mean, and $V(t) \equiv nv(t)$ as the aggregate variance.

Approximation 2. *For any $B > 0$, and $C > M$,*

$$\Pr(Q > B) \approx \exp\left(- \inf_{t>0} \; (B + (C-M)t)^2/(2V(t)) \right) \quad . \tag{2}$$

Provisioning formulae. One of the major tasks in network management is the provisioning of resources: choose the link rate and/or buffer size such that some pre-specified performance criterion is met. The above approximation formula (2) provides us with a tool for developing such provisioning rules. For several performance criteria, we derive the corresponding rules.

Link provisioning of an unbuffered resource. Suppose the goal is to provision the link such that the probability of exceeding the capacity C for a period of length T is smaller than ϵ. Hence we have to find the smallest $C = C(T, \epsilon)$ such that:
$\exp\left(-((C-M)T)^2/(2V(T))\right) \leq \epsilon$, cf. (2). It is readily checked that this yields, with $\delta := \sqrt{-2\log\epsilon}$,

$$C(T,\epsilon) = M + \delta/T \cdot \sqrt{V(T)} \quad . \tag{3}$$

Notice that $C(T, \epsilon)$ decreases in ϵ, as expected: the less stringent the performance target, the less bandwidth is needed.

Link provisioning of a buffered resource. In the setting of provisioning rule (3) we considered an unbuffered resource. In practice, however, network elements are often equipped with a queue, to absorb traffic rate fluctuations. If the router has a queue of size B, and suppose we wish to provision the capacity, we have to find the minimal $C = C(\epsilon)$ such that (2) is below ϵ. Hence we are searching for:

$$\min\left\{C \mid \forall t > 0 : \exp\left(-(B + (C-M)t)^2/(2V(t))\right) \leq \epsilon\right\} \quad .$$

After rearranging terms, we find:

$$C(\epsilon) = M + \inf_{t>0} \left(\delta/t \cdot \sqrt{V(t)} - B/t \right) \quad . \tag{4}$$

Again, the bandwidth required decreases in ϵ. Moreover, it also decreases in B: the larger the queue, the better traffic fluctuations can be absorbed by the buffer, and hence less link capacity is needed.

Buffer provisioning. Similarly, we can determine the minimum required buffer $B = B(\epsilon)$:

$$B(\epsilon) = \inf_{t>0} \left(\delta\sqrt{V(t)} - (C-M)t \right) \quad . \tag{5}$$

Example 3. *fBm. Motivated by several measurements studies [3], we focus here on one of the key models in current traffic theory, namely fractional Brownian motion input, i.e., Gaussian traffic with $V(t) = \sigma^2 t^{2H}$ – take for simplicity $\sigma = 1$. $H \in (0,1)$ is the so-called Hurst parameter; for network traffic a typical value is 0.7-0.8. Straightforward computations give for (3):*

$$C(T,\epsilon) = M + \delta/T^{1-H} \quad .$$

When computing $C(\epsilon)$ in (4), the optimizing t is given by $B^{1/H}(1-H)^{-1/H}\delta^{-1/H}$, yielding:

$$C(\epsilon) = M + \delta^{1/H}\left((1-H)/B\right)^{1/H-1} H \ .$$

In buffer provisioning rule (5) the optimizing t is given by $(\delta H)^{1/(1-H)}(C-M)^{-1/(1-H)}$, such that:

$$B(\epsilon) = \left(\delta H/(C-M)^H\right)^{1/(1-H)} \cdot (1-H)/H \ .$$

The main conclusion from this section is that the above provisioning formulae (3), (4), and (5) indicate that it is of crucial importance to have accurate estimates of the average traffic rate M, as well as the variance curve $V(\cdot)$ (i.e., $V(t)$ as a function of $t \geq 0$); having these at our disposal, we can find the required bandwidth capacity or buffer size. As estimating M is straightforward, the next sections concentrate on efficient methods for estimating the variance curve $V(\cdot)$.

3 Derivation of the Inversion Formula

As mentioned in the introduction, the mean traffic rate M can be determined by standard coarse-grained traffic measurements. It is clear that determining the variance curve $V(\cdot)$ is more involved. The standard way to estimate $V(T)$ (for some interval length T) is what we refer to as the 'direct approach'. This method is based on traffic measurements for disjoint intervals of length T, and just computes their sample variance. It is noted that the convergence of this estimator could be rather slow when traffic is long-range dependent [6–Ch. I], but the approach has two other significant drawbacks:

1. When measuring traffic using windows of size T, it is clearly possible to estimate $V(T)$, $V(2T)$, $V(3T)$, etc. However, these measurements obviously do not give any information on $V(\cdot)$ on time-scales *smaller* than T. Hence, to estimate $V(T)$ measurements should be done at granularity T or less; T is typically rather small. This evidently leads to a substantial measurement effort.
2. The provisioning formulae (4) and (5) require knowledge of the *entire* variance function $V(\cdot)$, whereas the direct approach described above just yields an estimate of $V(T)$ on a pre-specified time-scale T. Therefore, a method that estimates the entire curve $V(\cdot)$ is preferred.

This section presents a powerful alternative to the direct approach; we refer to it as the *inversion approach*, as it 'inverts' the buffer content distribution to the variance curve. This inversion approach overcomes the problems identified above. We rely on the many-sources framework of Sect. 2.

Define the 'most likely epoch of overflow' for a given buffer value $b > 0$: $t_b := \arg\inf_{t>0}(b+(c-\mu)t)^2/(2v(t))$; note that t_b is not necessarily unique. Define the set $\mathcal{T}$ as follows: $\mathcal{T} := \{t > 0 \mid \exists b > 0 : t = t_b\}$. The following theorem gives, for any $t > 0$, an upper bound on the variance $v(t)$, for given $I(b)$, and presents conditions under which this upper bound is tight.

Theorem 4. *(i) For any $t > 0$, $v(t) \leq \inf_{b>0}(b+(c-\mu)t)^2/(2I(b))$;*

(ii) There is equality *for all $t \in \mathcal{T}$;*
(iii) If $2v(t)/v'(t) - t$ grows from 0 to ∞ when t grows from 0 to ∞, then $\mathcal{T} = (0,\infty)$.

Proof. Clearly, due to Lemma 1, for all $b > 0$ and $t > 0$, we have that $I(b) \leq (b + (c-\mu)t)^2/(2v(t))$, which implies claim (i) immediately. Now consider a $t \in \mathcal{T}$. Then there is a $b = b_t > 0$ such that $I(b) = (b + (c-\mu)t)^2/(2v(t))$. We thus obtain claim (ii). Now consider claim (iii). We have to prove that for all $t > 0$ there is a $b > 0$ such that $t = t_b$. Evidently, t_b solves $2v(t)(c-\mu) = (b + (c-\mu)t)v'(t)$, or, equivalently, $b = b_t := (2v(t)/v'(t) - t)\,(c-\mu)$. Hence, it is sufficient if b_t grows from 0 to ∞ when t grows from 0 to ∞.

Note that even if condition (iii) does not apply, as was found in some recent traces, see [7], $v(t)$ will be bounded from above by the infimum over b, due to (i). Remarkably, Theorem 4 gives, loosely speaking, that for Gaussian sources the buffer content distribution uniquely determines the variance function. This property is exploited in the following heuristic.

Approximation 5. *The following estimate of the function $V(t)$ (for $t > 0$) can be made using the buffer content distribution:*

$$V(t) \approx \inf_{B>0} \frac{(B + (C-M)t)^2}{-2\log \Pr(Q > B)}. \tag{6}$$

Hence, if we can estimate $\Pr(Q > B)$, then 'inversion formula' (6) can be used to retrieve the variance; notice that the infimum can be computed for any t, and consequently we get an approximation for the entire variance curve $V(\cdot)$ (of course up to some finite horizon). These ideas are exploited in the procedure described in the next section.

4 Demonstration of the Inversion Procedure

In this section we show how the theoretical results of the previous section can be used to estimate $V(\cdot)$. First, we propose an algorithm for estimating the (complementary) buffer content distribution (in the sequel abbreviated to BCD), such that, by applying Approximation 5, the variance curve $V(\cdot)$ can be estimated. In our demonstration, second, we specialize to the case of synthetic input, i.e., traffic generated according to some stochastic process; we choose fBm input, but we emphasize that the procedure could be followed for any other process. Finally, we compare, for fBm, our estimation for $V(\cdot)$ with the actual variance curve, yielding a first impression of the accuracy of our approach (a more detailed numerical evaluation follows in Sect. 5 and 6).

The inversion procedure consists of two steps: (1) determining the BCD, and (2) 'inverting' the BCD to the variance curve $V(\cdot)$ by applying Approximation 5. We propose the following algorithm:

Algorithm 6. *Inversion approach.*

1. *Collect 'snapshots' of the buffer contents: $q_1, \ldots, q_N$; here q_i denotes the buffer content as measured at time $\tau_0 + i\tau$, for some $\tau > 0$. Estimate the* BCD *by the empirical distribution function of the q_i, i.e., estimate* $\Pr(Q > B)$ *by* $\phi(B) = \#\{i : q_i > B\}/N$.
2. *Estimate $V(t)$, for any $t \geq 0$, by* $\inf_{B>0}(B + (C-M)t)^2/(-2\log\phi(B))$.

In the above algorithm, snapshots of the buffer content are taken at a constant frequency. To get an accurate estimate of the BCD, both τ and N should be chosen sufficiently

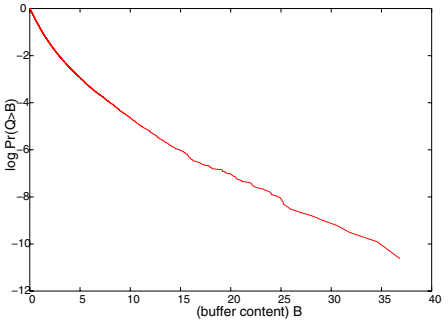

Fig. 1. Sample BCD

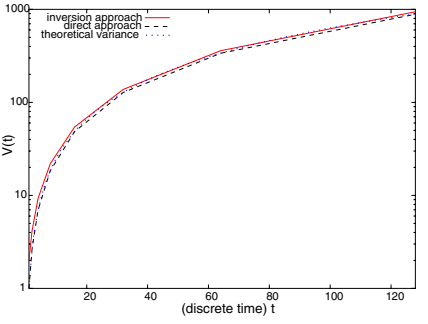

Fig. 2. Sample variance curves

large. We come back to this issue in Sect. 5. Notice that we chose a fixed polling frequency (i.e., τ^{-1}) in our algorithm, but this is not strictly necessary; the BCD-estimation procedure obviously still works when the polling epochs are not equally spaced.

In the remainder of this section we demonstrate the inversion approach of Algorithm 6 through a simulation with synthetic (fBm) input. The simulation of the queue fed by fBm yields an estimate for the BCD; this estimated BCD is inverted to obtain the estimated variance curve, which is compared with the actual variance curve.

Simulation procedure. Concentrating on slotted time, we generate traffic according to some stochastic process. In this example we focus on the case of fBm input, but it is stressed that the procedure could be followed for any other stochastic process. The traffic stream is fed into a simulated queue with link rate C. The buffering dynamics are simulated as follows: (1) using a fBm simulator [10], fBm is generated with a specific Hurst parameter $H \in (0, 1)$, yielding a list of 'offered traffic per time slot'; (2) for every slot, the amount of offered traffic is added to, and an amount equal to C is drained from the queue (while assuring the queue's content is non-negative); (3) every τ slots, the queue's content is observed, yielding N snapshots that are then used to estimate $\Pr(Q > B)$ (cf. Algorithm 6). In this demonstration of the inversion procedure, we generate an fBm traffic trace with Hurst parameter $H = 0.7$ and length 2^{24} slots. The link capacity C is set to 0.8, and we take snapshots of the buffer content every $\tau = 128$ slots.

Estimating the variance curve. We now discuss the output of the inversion procedure for our simulated example with fBm traffic. First we estimate the BCD; a plot is given in Fig. 1. For presentation purposes, we plot the (natural) logarithm of the BCD, i.e., $\log \Pr(Q > B)$. The BCD in Fig. 1 is 'less smooth' for larger values of B. This is due to the fact that large buffer levels are rarely exceeded, leading to less accurate estimates. Second, we estimate the variance $V(t)$ for t equal to the powers of 2 ranging from 2^0 to 2^7, using the BCD, i.e., by using Algorithm 6. The resulting variance curve is shown in Fig. 2 ('inversion approach'). The minimization (over B) was done by straightforward numerical techniques. To get an impression of the accuracy of the inversion approach, we have also plotted in Fig. 2 the variance curve as can be estimated directly from the synthetic traffic trace (i.e., the 'direct approach' introduced in Sect. 3), as well as the real variance function for fBm traffic, i.e., $V(t) = t^{2H}$. Figure 2 shows that the three

variance curves are remarkably close to each other. This confirms that the inversion approach is an accurate way to estimate the burstiness.

5 Error Analysis of the Inversion Procedure

In the previous section the inversion approach was demonstrated. It was shown to perform well for fBm with $H = 0.7$, under a specific choice of N and τ. Evidently, the key question is whether the procedure still works under other circumstances. To this end, we first identify the three possible sources of errors:

- The inversion approach is based on the *approximation* (2).
- $\Pr(Q > B)$ is *estimated*; there could still be an estimation error involved. In particular, we wonder what the impact of the choice of N and τ is.
- The procedure *assumes* perfectly Gaussian traffic, although real network traffic may not be (accurately described by) Gaussian.

We will now quantitatively investigate the impact of each of these errors on our 'indirect approach'. These investigations are performed through simulation as outlined in Sect. 4.

Approximation of the buffer content distribution. In Equation (2) an approximation of the BCD is given. As the inversion approach is based on this approximation, evidently, errors in (2) might induce errors in the inversion.

We first determine the infimum in the right-hand side of (2), which we consider as a function of B. In line with the previous section, we choose fBm input: $M = 0$ and $V(t) = t^{2H}$. Straightforward calculations now reveal that we can rewrite (2), viz.:

$$\log \Pr(Q > B) \approx -1/2 \cdot (B/(1-H))^{2-2H} \, (C/H)^{2H} \quad .$$

We verify how accurate the approximation is, for two values of H: the pure Brownian case $H = 0.5$, and a case with long-range dependence $H = 0.7$ (in line with earlier measurement studies of network traffic). Several runs of fBm traffic are generated (with different random seeds), 2^{24} slots of traffic per run. We then simulate the buffer dynamics. For $H = 0.5$ we choose link rate $C = 0.2$, for $H = 0.7$ we choose $C = 0.8$; these choices C are such that the queue is non-empty sufficiently often (in order to obtain a reliable estimate of the BCD). Figures 3 and 4 show for the various runs the approximation of the BCD, as well as their theoretical counterpart. It can be seen that, in particular for small B the empirically determined BCD almost perfectly fits the theoretical approximation.

Estimation of the buffer content distribution. As we estimate the BCD on the basis of snapshots of the buffer content, there will be some error involved. The impact of this error is the subject of this subsection. It could be expected that the larger N (more observations) and τ (less correlation between the observations), the better the estimate.

First, we investigate the impact of N. The simulator is run as in previous cases (with $H = 0.7$), with the difference that we only use the first $x\%$ of the snapshots samples to determine $\Pr(Q > B)$. Figure 5 shows the estimation of the buffer content distribution, for various x ranging from 0.1 to 100. The figure shows that, in particular for relatively small B, a relatively small number of observations suffices to get an accurate estimate of the BCD.

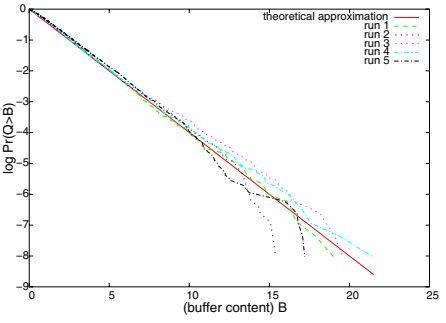

Fig. 3. $\Pr(Q > B)$ and theoretical approximation ($H = 0.5$)

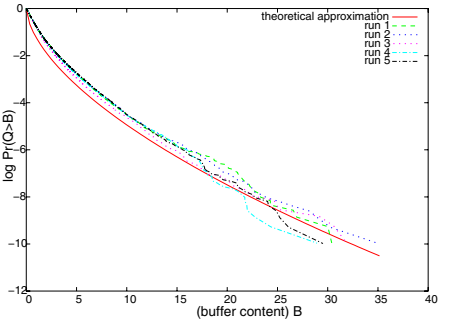

Fig. 4. $\Pr(Q > B)$ and theoretical approximation ($H = 0.7$)

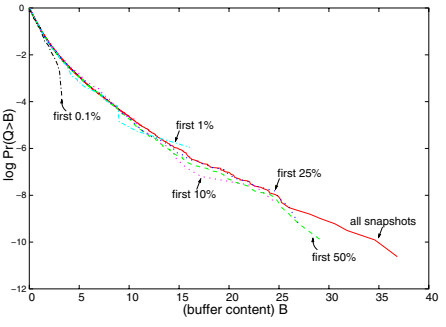

Fig. 5. Comparing $\Pr(Q > B)$ for various trace lengths

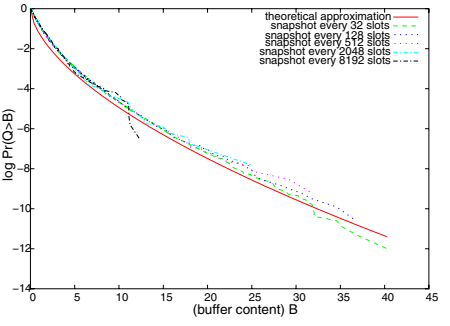

Fig. 6. Comparing $\Pr(Q > B)$ for various polling intervals

Second, we investigate the impact of the interval length between two consecutive snapshots τ. Figure 6 shows the determined BCD for τ ranging from observing every 32 to every 8192 slots. It can be seen that, particularly for small B the fit is quite good, even when the buffer content is polled only relatively rarely.

The impact of the Gaussianity assumption. Approximation (2) explicitly assumes that the traffic process involved is Gaussian. Various measurement studies find that real network traffic on the Internet is (accurately described by) Gaussian, see, e.g., [3]; others claim, however, that particularly on small time-scales, traffic may not be Gaussian [2]. Therefore we now investigate how sensitive our 'inversion approach' is with respect to Gaussianity of the input traffic.

We study the impact of non-Gaussianity by mixing, for every slot, a fraction α of the generated fBm traffic with a fraction $1 - \alpha$ traffic from an alternative (non-Gaussian) stream, before the mixture is fed to the (virtual) queue. Note that the variance of the mixture is $V(t) = \alpha^2 V_{[\mathrm{fBm}]}(t) + (1-\alpha)^2 V_{[\mathrm{alt}]}(t)$. We vary α from 1 to 0, to assess the impact of the non-Gaussianity.

The alternative input model that we choose here is an M/G/∞ input model, inspired by, e.g., [4, 8]. In the M/G/∞ input model, jobs arrive according to a Poisson(λ) process. The job durations are i.i.d., and during their duration each job generates traffic at a

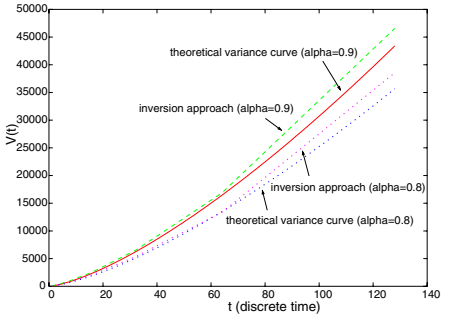

Fig. 7. Variance curves for Gaussian/non-Gaussian traffic mixtures, $\alpha = \{0.8, 0.9\}$

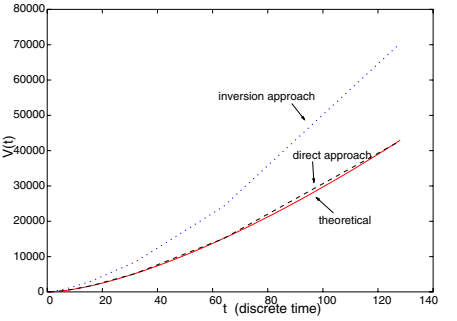

Fig. 8. Variance curves for Gaussian/non-Gaussian traffic mixture, $\alpha = 0$

constant rate r. In line with measurements studies, we choose Pareto(β) jobs. As the objective is to assess the impact of varying the parameter α, we have chosen to select the parameters of the M/G/∞ model such that the fBm and M/G/∞ traffic streams are 'compatible', in that their mean and variance $V(\cdot)$ are similar, which is achieved as follows.

The means of both traffic stream are made compatible by adding a drift to the fBm inputs equal to the mean of the M/G/∞ traffic stream, i.e., $\lambda r/(\beta - 1)$. Here λ and r can be chosen arbitrarily. The Gaussianity of the fBm input is not affected by the addition of such a drift.

To make the variances of both traffic streams compatible, we make use of a derivation in earlier work of the exact variance function $V(t)_{[\mathrm{alt}]}$, see [11]. It is not possible to achieve the desired 'compatibility' of the variance on all time scales. As long-range dependence is mainly a property of long time-scales, we choose to focus on these. For larger time scales, the variance of the M/G/∞ traffic stream roughly looks like $V_{[\mathrm{alt}]}(t) \approx 2r^2\lambda t^{3-\beta}/((3-\beta)(2-\beta)(\beta-1))$, assuming $\beta \in (1,2)$. We can now estimate the remaining parameters and compute the variance of the traffic mixture.

The next step is to run, for different values of α, the simulation. We then determine the (theoretical) variance curve of the traffic mixture, and compare it to the variance curve found through our 'indirect approach'. In Fig. 7 we focus on the 'nearly-Gaussian' cases $\alpha = 0.8$ and $\alpha = 0.9$, which are plotted together with their theoretical counterparts. The figure shows that the presence of non-Gaussian traffic has some, but no crucial impact on our inversion procedure. We also consider the (extreme) case of $\alpha = 0$, i.e., no Gaussian traffic at all, to see if our inversion procedure still works. In Fig. 8 the various variance curves are shown: the theoretical curve, the curve based on the 'direct approach', as well as the curve based on the inversion approach. Although not a perfect fit, the curves look similar and still relatively close to each other (but, of course, the fit is worse than for $\alpha = 0.8$ and 0.9). Note that the non-Gaussian traffic may 'have some Gaussian characteristics' if there is a large degree of aggregation, by virtue of central-limit type of arguments, which may explain that the fit is still reasonable.

We conclude that our simulation experiments show the 'robustness' of the inversion procedure. Despite the approximations involved, with a relatively low measurement effort, the variance curve is estimated accurately, even for traffic that is not perfectly Gaussian. Given the evident advantages of the inversion approach over the 'direct ap-

proach' (minimal measurement effort required, retrieval of the entire variance curve $V(\cdot)$, etc. – see the discussion in Sect. 3), the former method is to be preferred. In the next section we verify whether this conclusion also holds for real (i.e., not artificially generated) network traffic.

6 Empirical Validation of the Inversion Procedure

Whereas the previous sections studied the performance of our inversion approach by executing simulation experiments with synthetic traffic, in this section we use traces of real network traffic. We evaluate the inversion approach by comparing with the 'direct approach'.

Measurement and simulation setup. The traces used here are collected at the so-called (Ethernet) uplinks of five distinct networks (i.e., links that connect these networks to their Internet service providers). These networks resemble various scenarios, such as different types of users (e.g. students, 'normal consumers', and web-servers), and different access techniques (ranging from 512 kbps ADSL to 100 Mbps Ethernet), in order to test our 'indirect approach' under different circumstances. For each network, we have hooked up an off-the-shelf PC to a router/switch that copies all traffic from/to the uplink to the measurement PC. Using the standard tcpdump software, all packet headers are captured, time-stamped [9], and subsequently made anonymous through the tcpdpriv tool to protect the users' privacy. In this way we have obtained over 400 traces in total, each of them containing 15 minutes of traffic.

The collected packet traces are used in two ways: (1) to estimate the variance curve through the 'direct approach'; and (2) to 'replay' the traffic through a virtual queue and link, analogous to the simulation procedure described in Sect. 4, and then applying the 'indirect approach' to estimate the variance, cf. Algorithm 6.

Empirical validation. Because of paper length restrictions, we cannot discuss here the empirical validation for all traces, nor all networks that we have measured. Therefore we resort to presenting only the validation of two of the hundreds of traces; empirical validation of the other traces has shown similar results. The first example is based on a trace captured from the 1 Gbps uplink of an ADSL access network that has about 800 subscribers. The second example is about a web-server farm, with approximately 150 servers; access is 100 Mbps Ethernet, and the uplink is 30 Mbps.

We have set the 'sampling interval' τ (used to estimate the BCD) to 1 second, to ensure that a considerable number of snapshots (900) can be taken. A substantial fraction of these snapshots, also depending on the value of the output link's capacity, do not provide any information as the buffer turns out to be empty at the time the snapshot is taken. We choose the smallest interval length for which we compare the variance estimated through our inversion approach with the actual variance found through the 'direct approach', to be 10 ms (which is, in other words, 100 times as small as the interval we poll the buffer occupancy).

Figures 9 and 10 show the variance curves for the first and second example, respectively. Clearly, the graphs demonstrate that the inversion approach is capable of adequately estimating $V(\cdot)$. Hence even for real network traces, of which for instance the Gaussian character is far from evident, our coarse-grained buffer polling approach suffices to estimate $V(\cdot)$. We recall that this implies that this eliminates the need for doing

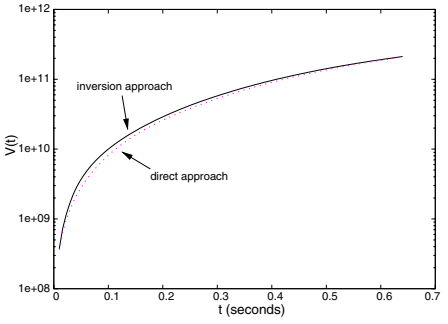

Fig. 9. Variance curves, ADSL access net

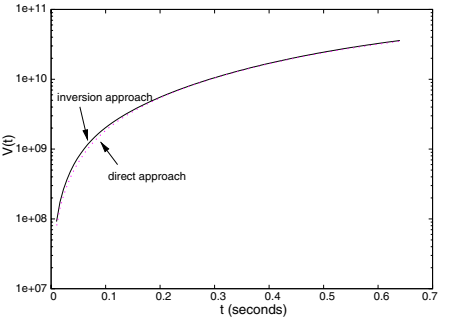

Fig. 10. Variance curves, web-server farm

detailed, small-time-scale, traffic measurements. In particular, our results indicate that we can estimate the variance on the time-scale 10 ms, just by polling the buffer content at a low frequency, without ever doing a traffic measurement on the 10 ms time scale.

7 Discussion

After the numerical tests of the previous sections, the next issue concerns the feasibility of implementing the inversion approach in operational environments.

The above experiments (both with artificial traffic and real traces) have been done off-line, in that we have used scripts that 'parse' the synthetic and real traffic, mimicking the buffer content dynamics. An interesting question is whether this approach is feasible in run-time, in operational environments. From the proposed procedure, we can derive three functional requirements for an implementation of the inversion procedure.

First, a notion of the amount of data in a buffer. This has already been addressed in real Internet routers: Random Early Detection (RED) [12] queuing algorithms, which are widely implemented in modern routers, also keep track of the amount of queued data. Second, a way to poll the buffer occupancy; SNMP may be used for this (in case the entire procedure is not run on the router itself). Third, software/hardware to determine the BCD, and then determine the resulting estimate of $V(\cdot)$. This has also already been addressed – see our results in this paper.

The above aspects lead us to believe that there are no fundamental or conceptual problems preventing the actual application of our approach in practice.

8 Concluding Remarks

We have presented a novel method to determine the burstiness of network traffic; here burstiness is in terms of the variance $V(T)$ of the traffic generated in an arbitrary window of length T. Our approach estimates the entire variance curve $V(\cdot)$ (of course up to some horizon), also on small time-scales, *without performing detailed traffic measurements*. Instead, the buffer content is polled (at some coarse-grained frequency) to obtain an estimate of the buffer content distribution. Then this distribution is 'inverted' to find the variance curve of the traffic rates, which gives the 'burstiness' $V(T)$ of the

network traffic *at any time scale* T (up to some horizon). Knowledge of the variance curve can immediately be used in provisioning formulae. We have presented the mathematical foundations under this inversion method, and have investigated its accuracy by performing a thorough analysis of the possible sources of error. Although there exists an assumption on Gaussianity of the traffic, the error analysis has shown that even when the traffic is not perfectly Gaussian, our inversion method gives good results.

Furthermore, we have validated our approach in various real-life settings. This has shown that our inversion method provides remarkably accurate estimates of the traffic's burstiness. In particular, we have observed that our approach yields reliable estimates of the variance for very small time-scales. A seemingly counterintuitive but representative example: by sampling from the buffer occupancy every second we have found an accurate estimate of the variance on the time-scale of 10 ms.

In future work we intend to investigate the impact of the precise choices of the queue's link rate C on our inversion approach. In particular it would be interesting to optimize C such that the fit is optimal (with respect to some optimality criterion). We also intend to test our approach in even more real-life settings, and extend the concept to a network setting.

References

1. van de Meent, R., Pras, A., Mandjes, M., van den Berg, H., Roijers, F., Venemans, P., Nieuwenhuis, L.: Burstiness predictions based on rough network traffic measurements. In: Proc. 19th World Telecommunications Congress (2004)
2. Kilpi, J., Norros, I.: Testing the gaussian approximation of aggregate traffic. In: Proc. Internet Measurement Workshop (2002)
3. Leland, W., Taqqu, M., Willinger, W., Wilson, D.: On the self-similar nature of Ethernet traffic (extended version). IEEE/ACM Trans. on Networking **1** (1994) 1–15
4. Addie, R., Mannersalo, P., Norros, I.: Most probable paths and performance formulae for buffers with gaussian input traffic. European Trans. Telecommunications **13** (2002) 183–196
5. Fraleigh, C., Tobagi, F., Diot, C.: Provisioning IP Backbone Networks to Support Latency Sensitive Traffic. In: Proc. IEEE Infocom, (2003)
6. Beran, J.: Statistics for Long-Memory Processes. Chapman & Hall/CRC (1994)
7. Zhang, Z., Ribeiro, V.J., Moon, S., Diot, C.: Small-Time Scaling Behaviors of Internet Backbone Traffic: An Empirical Study. In: Proc. IEEE Infocom (2003)
8. Fredj, S.B., Bonald, T., Proutiere, A., Régnié, G., Roberts, J.W.: Statistical Bandwidth Sharing: A Study of Congestion at Flow Level. In: ACM SIGCOMM CCR **4** (2001) 111-122
9. van de Meent, R., Pras, A., Mandjes, M., van den Berg, H., Nieuwenhuis, L.: Traffic Measurements for Link Dimensioning: A Case Study. Proc. of the 14th IFIP/IEEE Workshop on Distributed Systems: Operations and Management (DSOM2003) (2003) 106–117
10. Dieker, T.: (Fractional Brownian motion simulator) `http://homepages.cwi.nl/~ton/fbm/index.html`.
11. Mandjes, M., Saniee, I., Stolyar, A.: Load characterization, overload prediction and load anomaly detection for voice over IP traffic. In: Proc. 38th Allerton Conference (2000)
12. Floyd, S., Jacobson, V.: Random Early Detection (RED) gateways for Congestion Avoidance. IEEE/ACM Trans. on Networking **1** (1993) 397–413