Buffer sizing theory for bursty TCP flows

Damon Wischik*
Computer Science, UCL
Email: D.Wischik@cs.ucl.ac.uk

***Abstract*— In a router serving many TCP flows, queues will build up from time to time. The manner in which queues build up depends on the buffer space available and on the burstiness of the TCP traffic. Conversely, the traffic generated by a TCP flow depends on the congestion it sees at queues along its route. In order to decide how big buffers should be, we need to understand the interaction between these effects.**

This paper reviews the buffer-sizing theory in [1] and extends it to cope with bursty TCP traffic. This enables us to explain an observation about TCP pacing made in [2].

I. Introduction

The general purpose of buffers in Internet routers is to accomodate bursts in traffic.

'Bursty' is a word with no agreed meaning. Some authors stress the scale-free nature of traffic fluctuations: μs-long fluctuations on top of ms-long fluctuations on top of second-long fluctuations etc. [3]. Some authors propose to describe traffic variability simply in terms of the variance of the amount of traffic generated in a given time interval [4]. Some authors consider the effective bandwidth to be a useful descriptor of traffic variability [5].

In this paper, I will describe a different notion of burstiness, one tied specifically to the mechanics of the TCP protocol. I will explain how burstiness affects the build-up of queues, and how this in turn feeds back to affect the behaviour of TCP sources. This theory results a simple model which lets us calculate the impact of burstiness and buffer size on TCP throughput, link utilization, and drop probability.

Section II gives a brief overview of those parts of TCP that we will be concerned with, and explains the notion of burstiness. Section III describes the impact of bursty traffic on queues, expanding on the results for smooth traffic in [1]. Section IV and V give an integrated mathematical model for network performance in the presence of bursty TCP traffic.

A naive guess would be that greater burstiness leads to larger queues, which leads to more packet loss, which leads to reduced throughput. This guess suggests a mechanism for improving throughput: modify TCP so that it produces smoother traffic flows. Simulation studies [2], [6] show that this is not always the outcome: that smooth traffic flows can paradoxically cause burstiness in the network. In Section VI we use our mathematical model to explain this observation.

II. TCP

This paper will be concerned with long-lived TCP flows. The typical behaviour of such a flow is as follows. Packets are sent from a source to a receiver. The receiver sends back acknowledgements. Let RTT be the time it takes to send a packet from the source to the receiver, and to get back an acknowledgement. (RTT may vary, because of fluctuations in queue size, but we will not model this.) The source maintains a window parameter cwnd; this is the target number of packets that have been sent but not yet acknowledged. When an acknowledgement is received, cwnd increases by $1/\texttt{cwnd}$, and this may permit new packets to be sent. When a dropped packet is detected, cwnd decreases by $\texttt{cwnd}/2$, but no more than once every RTT. This results in cwnd evolving in a sawtooth, as illustrated in Figure 1. If cwnd stays constant then the source will transmit cwnd packets every RTT, i.e. it will send at rate $x = \texttt{cwnd}/RTT$ pkt/s.

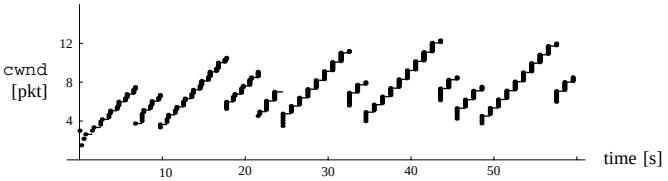

Fig. 1. cwnd as a function of time. When there are no drops cwnd increases steadily; when a drop is detected then cwnd is reduced. This results in a characteristic 'sawtooth' graph. Here $RTT = 1$s and the packet drop probability is $p = 5\%$.

(There is much more to TCP than this, including the mechanism by which TCP detects dropped packets, timeouts, and the behaviour of short-lived flows. There are some brief remarks about these aspects of TCP in the conclusion.)

This description of TCP permits some rather different behaviours. Figure 2 shows two simulation runs in which the round trip time is the same, the initial cwnd is the same, and there are no packet drops—the only difference is the spacing of the first three acknowledgements. In the first run, they arrive at times 0.2, 0.5 and 0.7; in the second they arrive at times 0.2, 0.22 and 0.24. The impact of clumped acknowledgements is to induce clumped packet transmissions. It can also be seen that TCP itself induces some clumpiness in packet transmissions, even when the acknowledgements start out paced.

A. Traffic/burstiness model

We will work with two different models for TCP, a smooth traffic model and a bursty traffic model.

Smooth TCP traffic: In the first model, the traffic generated by a single TCP flow is a point process. Each point represents a single data packet (and all packets are taken to have the

*Research supported by a Royal Society university research fellowship, and DARPA Buffer Sizing Grant no. W911NF-05-1-0254.

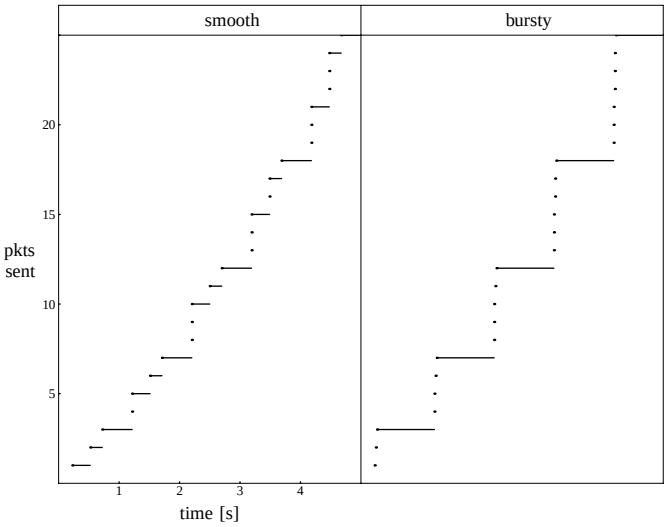

Fig. 2. Number of packets sent as a function of time. If the initial packets are spaced out then subsequent packets will inherit this spacing and the flow will be smooth; if the initial packets are clumped together then the entire flow will be bursty.

same size). The points are assumed to be separated, e.g. there exists some $\delta > 0$ such that no two points are any closer than δ in time. This would be guaranteed if, for example, the TCP source is throttled by a slow access link. The precise statistics of the point process do not matter in what follows.

Bursty TCP traffic: In the second model, the traffic generated by a single TCP flow is a point process. Each point represents a *clump* of data packets. For a TCP flow with average window size w, the number of packets in a clump is assumed to be uniformly distributed between $2w/3$ and $4w/3$; this reflects the shape of the TCP sawtooth. The clumps are assumed to be separated by one round trip time. As above, packets are all the same size.

These two cases are extremes. Figure 2 suggests that the true behaviour of TCP is likely to be somewhere between these two extremes. But it's easiest to work with the extremes, so we will stick with them in the analysis that follows.

B. Throughput formula

When a TCP flow experiences drop probability p, then (according to e.g. [7]) its throughput is roughly

$$x = \frac{0.87}{RTT\sqrt{p}} \quad \text{pkt/s.} \tag{1}$$

When packet drops are independent, p is simply the per-packet drop probability. However in the bursty TCP model it's likely that if one packet in a clump is dropped then all subsequent packets in that clump are dropped also. In this case it's easiest to define p operationally: take two successive clumps with drops in them, and let $1/p - 1$ be the expected number of not-dropped packets between the drops. (For clumps of size 1, this reduces to the former case of independent packet drops.) A crude estimate is $p = q/w$ where q is the per-clump drop probability and w is the average clump size; this should be a decent approximation when q is small.

III. Queue model

Consider a single queue, with constant service rate, fed by many TCP flows. Raina and Wischik [1] have described the queueing behaviour that results, for smooth TCP traffic. In this section we will review the conclusions of that paper, and also see how the behaviour changes when the flows are bursty. The aim of this section is to produce an expression for p, the loss probability term in (1).

In fact, [1] gives two different queueing models, one for small buffers and one for large buffers. The models are justified heuristically, in the limit as the number of flows N increases. The small-buffer model is suitable when buffer size does not depend on N; the large-buffer model is suitable when buffer size increases with N. To be concrete, the small-buffer model seems good for buffers up to a several hundred packets, and the large-buffer model seems good when maximum possible queueing delay is a non-negligble fraction of the round trip time.

A. Smooth traffic, small buffers

Let there be N smooth TCP flows with average window size w and common round trip time RTT (so that the average transmit rate is $x = w/RTT$), sharing a queue with service rate NC and buffer size B. Then the packet drop probability p is approximately the same as that for a queue with constant service rate NC and buffer size B, fed by a Poisson flow of rate Nx (i.e. a classic $M/D/1/B$ queue). Indeed, the queue length distributions in the real system and the approximate Poisson system should be roughly equal.

The justification for approximating the traffic by Poisson is that the typical duration of a busy period is $O(1/N)$, and over an interval of duration $1/N$ the aggregate of N independent simple point processes is approximately Poisson.

B. Bursty traffic, small buffers

Let there be N bursty TCP flows with average window size w and common round trip time RTT (so that the average transmit rate is $x = w/RTT$). Packets arrive in clumps, and the clump size is uniform between $2w/3$ and $4w/3$, so the average clump size is w. Clumps arrive as a point process, say with mean rate λ. The average transmit rate is then $x = \lambda w$ pkt/s, so $\lambda = x/w$.

Over an interval of duration $1/N$, the aggregate *point* process is approximately Poisson, as above, and so the aggregate *packet* arrival process can be modelled as a batch Poisson arrival process, where the batch size is uniform between $2w/3$ and $4w/3$. For large N, the duration of the interval $1/N$ is so short that there can be at most one clump from any single flow, and so we will assume that batch sizes are independent. Hence the packet drop probability is the same as for a queue with constant service rate NC, fed by packets which arrive in clumps according to a Poisson process of rate $N\lambda$.

For numerical purposes one should set up a Markov chain and compute the packet drop probability p from it. For simple

algebraic argument, here is a further simplification. First some notation. Write $L_C(B, \lambda, \mathcal{D})$ for the drop probability in a queue with constant service rate C, buffer size B, and a Poisson arrival process of rate λ where packet sizes are independent and have distribution $\mathcal{D}$. (This is the first simplification. In the real system, if a clump of packets arrives and some of the packets can fit in the buffer then those packets are admitted; in this $L_C(B, \lambda, \mathcal{D})$ system the entire clump is dropped.) We want to calculate $q = L_C(B, \lambda, \mathcal{U}_w)$, where $\mathcal{U}_w$ denotes the uniform distribution between $2w/3$ and $4w/4$. As a further simplification, approximate this by $L_C(B, \lambda, \mathcal{F}_w)$, where $\mathcal{F}_w$ indicates that all batches have fixed size w. But this is exactly equal to $L_{C/w}(B/W, \lambda, \mathcal{F}_1)$ (just rescale packet size), which is exactly equal to $L_C(B/W, \lambda w, \mathcal{F}_1)$ (just rescale time). In summary, while the packet drop probability for smooth traffic is $L_C(B, x, \mathcal{F}_1)$, the clump drop probability for bursty traffic is $q \approx L_C(B/w, x, \mathcal{F}_1)$. An even cruder approximation is

$$q \approx \frac{(1-\rho)\rho^{B/w}}{1-\rho^{B/w+1}}, \quad \text{where } \rho = x/C; \tag{2}$$

this comes from taking the service to be Markov rather than constant-rate.

What we have calculated here is the clump drop probability, i.e. the probability that there is not enough space in the buffer to accomodate all the packets in an incoming clump. The p in (1) is then $p \approx q/w$.

C. Smooth traffic, large buffers

Let there be N smooth TCP flows with average window size w and common round trip time RTT (so that the average transmit rate is $x = w/RTT$), sharing a queue with service rate NC. According to [1], if $x < C$ then the queue length is $O(1)$ as $N \to \infty$, and the packet drop probability is approximately zero. If $x > C$ then the packet drop probability is $(x-C)/x$, or $1 - 1/\rho$ when expressed in terms of $\rho = x/C$, from Little's Law. The amount of free space in the queue can be estimated by considering a queue with constant arrival rate NC and with exponential service times of rate Nx; the queue size distribution in this $D/M/1$ queue is the same as the distribution of free buffer space in the real queue. Furthermore, the typical time between two drops if $O(1/N)$.

D. Bursty traffic, large buffers

Let there be N bursty TCP flows, with parameters as in Section III-B. If $x < C$ then, as with smooth traffic, the queue size is $O(1)$ and the drop probability is approximately zero. If $x > C$ then the packet drop probability is still $(x-C)/x = 1-1/\rho$, from Little's Law. To find the probability q that an arriving clump of packets experiences one or more packet drops, here is an approximation:

For smooth traffic, the amount of free buffer space has the same distribution as the queue size in a $D/M/1$ queue. Approximate it by the queue size of an $M/M/1$ queue with the same mean arrival and service rates; this gives

$$\mathbb{P}(\text{free space} = r) = (1 - 1/\rho)/\rho^r.$$

(Observe that the probability that there is no free space, i.e. that an incoming packet is dropped, is $1 - 1/\rho$, so this approximation is consistent with the smooth-TCP case.) Now consider an incoming clump of w packets. This clump experiences a drop if the amount of free space is less than w, which has probability $1 - 1/\rho^w$. We therefore propose the approximation $q \approx 1 - 1/\rho^w$. The p in (1) is then $p \approx q/w$.

IV. Fixed point calculation

When deciding on buffer size, the first questions to ask are: what throughput do I expect to get? what is the link utilization? and what sort of quality of service? We now find out how these quantities depend on TCP burstiness.

Consider N long-lived TCP flows sharing a single bottleneck link with service rate NC, with drop probability p, and where the common round trip time is RTT. By (1), the offered traffic intensity is

$$\rho = \frac{Nx}{NC} = \frac{0.87}{CRTT\sqrt{p}}. \tag{3}$$

From Section III, we have two different formulae for what $q = pw$ might be:

$$q \approx \begin{cases} (1-\rho)\rho^{B/w}/(1-\rho^{B/w+1}) & \text{for small buffers} \\ (1-1/\rho^w)^+ & \text{for large buffers} \end{cases} \tag{4}$$

where $\rho = x/C$ is the offered traffic intensity, B is the buffer size and w is the average clump size: $w = 1$ for smooth TCP traffic and $w = xRTT$ for bursty TCP traffic. (One can come up with more accurate values for p and q from a Markov chain calculation, but these simple formulae are more readily interpretable.)

We have two equations with two unknowns, ρ and p. To find ρ and p, solve the two equations simultaneously. This is referred to as a *fixed point* solution. Figure 3 illustrates. There are two confounding effects: burstier traffic leads to higher per-packet drop probability, which indicates lower throughput; but at the same time burstier traffic means that packet drops come in bursts so TCP doesn't respond to all of the packet drops, which indicates higher throughput. For the parameters in Figure 3 these two effects cancel out. For smaller buffers or for larger burst sizes, the net effect is to reduce throughput. For very large buffers, the difference between bursty and smooth is negligible.

For more complicated network topologies, there will be more equations: one for each route of TCP flows, and one for each queue. They should all be solved simultaneously, as in [8], [9].

V. Stability calculation

When deciding on buffer size, a second question to ask is: do I expect stable good performance, or will there be jitter and instability? Fluid models have emerged as a powerful tool for answering this question. A crude first-cut answer is as follows:

Consider as above a single bottleneck shared by N long-lived flows, with mean rate x, and where the service rate is NC. Let $\rho = x/C$ be the traffic intensity. Let $p(\rho)$ be the

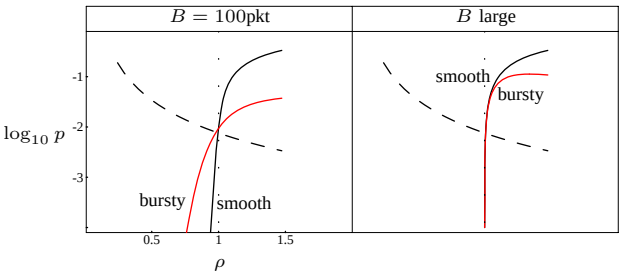

Fig. 3. A plot of drop probability p against traffic intensity ρ. The dashed line shows TCP throughput (3), with $C\mathit{RTT} = 6$pkt. The solid lines show (4), one line for smooth traffic ($w = 1$) and one for bursty traffic ($w = x\mathit{RTT}$). Buffer size is either 100pkt (left) or very large (right). Where the dashed line intersects with the relevant solid line, there is the simultaneous solution of (3) & (4).

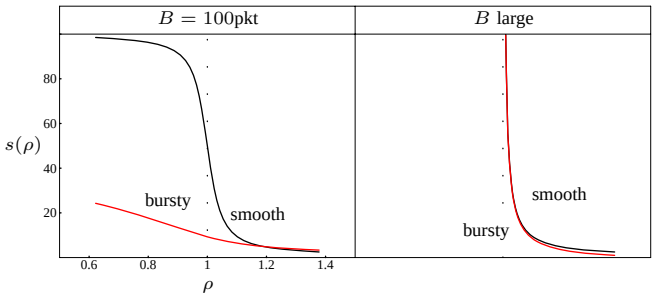

Fig. 4. Instability index $s(\rho)$ as a function of traffic intensity ρ. The higher the instability index, the more prone the system is to synchronization. For a given small buffer size, smooth TCP has a higher instability index than bursty TCP. For large buffer size there is negligible difference.

drop probability resulting from traffic intensity ρ, from one of the calculations in Section III. Let ρ^* be the fixed-point traffic intensity, calculated as in Section IV. Define the *instability index* to be

$$s(\rho^*) = \frac{\rho^* p'(\rho^*)}{p(\rho)}.$$

The larger this is, the more prone the system is to instability (see (19) and (20) in [10]). Figure 4 plots the instability index as a function as a function of ρ^*. With small buffers, smooth traffic is much more prone to instability than bursty traffic; with large buffers this effect is still there but it's barely noticeable.

Reference [1] gives much more detail about the nature and consequences of instability. The typical way in which instability shows itself is in oscillations in queue size, which leads to synchronization of TCP flows (i.e. many of them cutting their windows at the same time), which can lead to unfairness [6]. Additionally it leads to jitter, and oscillations in traffic rate. In [1] it is explained how to calculate the amplitude of all these oscillations.

VI. TCP Pacing

TCP pacing is a mechanism which enforces smooth traffic. It does this by spacing out packets: if the window is `cwnd` and the round trip time is *RTT* then it attempts to send one packet every *RTT*/`cwnd`. (See [2] for references.)

According to the results in the previous two sections, this will have several different effects. First, the traffic is made smoother, so packet drop probability is lower, which indicates increased throughput. Second, packets are spaced out, so packet drops are made independent, so TCP responds to more of them, which indicates decreased throughput. (The balance of these two effects will depend on the specific parameters.) Third, the instability index is made higher, which indicates instability and synchronization. Paradoxically, by making the traffic smoother at the source, we have induced "bursty" behaviour in the network! This theoretical prediction validates simulation results in [2], [6].

The problem of synchronization can be solved by making the buffer smaller. This makes $p'(\cdot)$ smaller, so the instability index is lower.

It can be seen that TCP burstiness has a range of different effects, sometimes conflicting, and that the relationship between burstiness and buffer size merits careful analysis.

A topic for further analysis is the impact of short-lived flows, timeouts, etc. In slow start, TCP tends to emit packets in clumps, which may make the aggregate traffic burstier. On the other hand, many short flows (of one or two packets) may make the aggregate traffic smoother, so that occasional bursts do not do too much damage.

References

[1] Gaurav Raina and Damon Wischik, "Buffer sizes for large multiplexers: TCP queueing theory and instability analysis," in *EuroNGI*, 2005.

[2] Amit Aggarwal, Stefan Savage, and Tom Anderson, "Understanding the performance of TCP pacing," in *IEEE Infocom*, 2000.

[3] 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*, vol. 2, pp. 1–15, 1994.

[4] Ronald G. Addie, Moshe Zukerman, and Timothy D. Neame, "Application of the central limit theorem to communication networks," Tech. Rep. SC-MC-9819, University of Southern Queensland, 1998.

[5] Frank Kelly, "Notes on effective bandwidths," in *Stochastic Networks: Theory and Applications*, F. P. Kelly, S. Zachary, and I. Ziedins, Eds., Royal Statistical Society Lecture Note Series, chapter 8, pp. 141—168. Oxford University Press, Oxford, 1996.

[6] David X. Wei, Pei Cao, and Steven H. Low, "TCP pacing revisited," Unpublished, 2006.

[7] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: a simple model and its empirical validation," in *Proceedings of ACM SIGCOMM*, 1998.

[8] Tian Bu and Don Towsley, "Fixed point approximations for TCP behavior in an AQM network," in *ACM SIGMETRICS*, 2001.

[9] R. J. Gibbens, S. K. Sargood, C. Van Eijl, F. P. Kelly, H. Azmoodeh, R. N. Macfadyen, and N. W. Macfadyen, "Fixed-point models for the end-to-end performance analysis of IP networks," in *13th ITC specialist seminar*, 2000.

[10] Frank Kelly, "Fairness and stability of end-to-end congestion control," *European Journal of Control*, 2003.