# Part II: Control Theory for Buffer Sizing

Gaurav Raina*
Judge Institute, Cambridge
gr224@cam.ac.uk

Don Towsley*
Computer Science, UMass
towsley@cs.umass.edu

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

## ABSTRACT

This article describes how control theory has been used to address the question of how to size the buffers in core Internet routers. Control theory aims to predict whether the network is stable, i.e. whether TCP flows are desynchronized. If flows are desynchronized then small buffers are sufficient [14]; the theory here shows that small buffers actually *promote* desynchronization—a virtuous circle.

## Categories and Subject Descriptors

C.2 [**Internetworking**]: Routers; C.4 [**Performance of systems**]: Modeling techniques

## General Terms

Design, Performance, Theory

## Keywords

Buffer size, TCP, congestion control, fluid model, control theory, synchronization

## 1. INTRODUCTION

The starting point of control-theoretic analysis is to write down a set of differential equations for all the rates of all the flows in a network, and for the drop probabilities at all of the routers, and then to determine whether this dynamical system is *stable*. Stability is simply the control-theoretic term for desynchronization between TCP flows. The theory we describe here aims to predict whether and to what extent there is synchronization.

A network will generally be stable for certain buffer sizes and unstable for others. We will explain how to choose buffer sizes to make it stable. Stability is also affected by Active Queue Management (AQM) parameters, round trip times, traffic mixes, and the TCP congestion avoidance algorithm itself. We will go on to describe how certain changes to TCP's rules for increasing and decreasing window size make the entire network less prone to synchronization.

## 2. FLUID MODEL

We now present differential equations for describing the network. There is one natural equation for TCP dynamics. For the queue we will describe two different dynamics: one suitable for queues with small buffers, the other for queues with large buffers and AQM.

The equations are supported by limit theorems (and simulations). The reason there are two possibilities for the queue is that there are two natural ways to scale the system[1]: in the large-buffer limit the number of flows increases and the line rates increase in proportion and the buffer *delay* is kept fixed; in the small-buffer limit the number of flows and line rates increase as before but the buffer *size* is either kept fixed or increases slowly, so that delay tends to zero.

### *Differential equation for TCP.*

Consider a collection of $N$ TCP flows with common round trip time $\mathit{RTT}$, and subject to a common packet loss probability. Let the average window size of all $N$ flows at time $t$ be $w(t)$, measured in packets, so that the average rate at which packets are sent is $x(t) = w(t)/\mathit{RTT}$. Let the packet loss experienced by packets sent at time $t$ be $p(t)$. The equation is [9]

$$\frac{dw(t)}{dt} = \frac{1}{\mathit{RTT}} - \frac{w(t)}{2}\Big[x(t-\mathit{RTT})p(t-\mathit{RTT})\Big]. \qquad (1)$$

The first term represents additive increase of window size, at rate 1 packet per round trip time for each of the $N$ flows. The second term represents multiplicative decrease of window size when packet losses are detected; the total transmission rate at time $t - \mathit{RTT}$ was roughly $Nx(t - \mathit{RTT})$, so the rate of packet losses is $Nx(t - \mathit{RTT})p(t - \mathit{RTT})$, and each loss results in the total window size $Nw(t)$ being reduced by $w(t)/2$ on average.

### *Large buffers, AQM.*

Let $q(t)$ be the queue size at time $t$, and let $y(t)$ be the total rate at which packets arrive at the queue. Suppose the AQM scheme drops packets with probability $L_{\mathrm{AQM}}(q)$ when the queue size is $q$. (If desired, it is easy to write down an extra equation to take account of queue-size averaging [9].) Let the total service rate be $C$. Then the extra equations are

$$\frac{dq(t)}{dt} = \begin{cases} y(t)\big(1-p(t)\big) - C & \text{if } q(t) > 0 \\ \max\big\{y(t)\big(1-p(t)\big) - C, 0\big\} & \text{if } q(t) = 0 \end{cases} \qquad (2)$$
$$p(t) = L_{\mathrm{AQM}}(q(t)).$$

*Research supported by DARPA, including Buffer Sizing Grant no. W911NF-05-1-0254. DJW is also supported by a Royal Society university research fellowship, and GR by the Communications Research Network, Cambridge.

[1]These equations also been used to describe systems with few flows and large line rates [2].

The first equation comes from considering arrivals at the queue. If total arrival rate is $y(t)$ but the drop probability is $p(t)$, then the rate at which work actually enters the queue is $y(t)\big(1-p(t)\big)$, and so the rate at which the queue grows is $y(t)\big(1-p(t)\big)-C$. If $q(t)=0$, since the queue cannot go negative, we have to take the positive part of $y(t)\big(1-p(t)\big)-C$. We assume that when the queue is full $L(q(t))=1$.

### *Small buffers, droptail.*

Let $y(t)$ be the total rate at which packets arrive at the queue, and let $C$ be the service rate. Let $L_B(y)$ be the packet loss probability for an $M/D/1$ queue with buffer $B$, service rate $C$, and Poisson arrivals of rate $y$. As argued in [14], Poisson arrivals are a good approximation when the buffer is small. If the line rate is high then the typical length of a busy cycle will be very short, which indicates that the packet loss probability depends only on the current arrival rate. Thus

$$p(t) = L_B(y(t)). \tag{3}$$

If $B$ is very large then $L_B(y) \approx (1-C/y)^+$. The crucial point is that in this model the queue size fluctuates so quickly that TCP cannot control the queue *size*, only its *distribution*. That is why $q(t)$ does not appear in this equation.

### *Discussion.*

The descriptions of TCP we have given here are crude, but they have been validated and found to match well to `ns2` simulations [9]. The theory supporting them is described further in [11]. More refined differential equation models are possible [1, 8, 2]; these can incorporate timeout behaviour and flow duration.

The reason for the difference between the large-buffer and small-buffer equations is explored in [3, 11]. Those references also describe large buffers with droptail and small buffers with AQM.

### *Networks.*

These equations can be augmented in the natural way to describe networks with multiple links and heterogeneous round trip times.

## 3. ANALYSIS TECHNIQUE

We have specified a system of differential equations (either (1)&(2) or (1)&(3), depending on the buffer size). This system can be solved numerically, given initial conditions, though care is needed because of the time delays ($t-RTT$).

It can also be analysed theoretically. Just as ordinary differential equations can be analysed to determine whether they are stable, critically damped, unstable and oscillatory, or unstable and divergent[2], so too can these time-delayed differential equations. Complete results are known for single links with homogeneous delays and either large or small buffers. Sufficient conditions for stability are known for heterogeneous networks with small buffers. For recent surveys see [6, 12], and for details of the calculations here see [10].

We can then vary the buffer size, and see how stability is affected. But why is stability important?

[2]To be precise, the results described here concern local stability, not global stability. Another term for 'unstable and oscillatory' is 'locally stable limit cycles'.

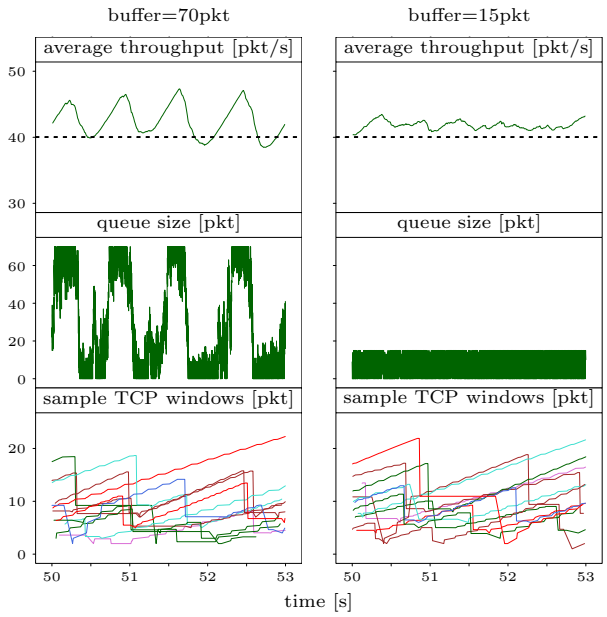


**Figure 1: Traces from a packet-level simulation of a single bottleneck link with 1000 flows, round trip times uniform in [120,280]ms, capacity 480Mb/s, and buffer of either 70 or 15 packets.**

## 4. STABILITY AND DESYNCHRONIZATION

To illustrate the relationship between stability and desynchronization, we plot in Figure 1 two traces from a packet-level simulation of a single bottleneck link: one with parameters for which the theory in the next section predicts instability, one for which it predicts stability. There are 1000 flows sharing a 480Mb/s link (i.e. available bandwidth per flow is $C=40$pkt/s). Round trip times are chosen uniformly at random from $[120,280]$ms. Also, each flow has an ingress link of capacity $3C$, and the reverse path is loaded with 1000 TCP flows with similar parameters. The buffer is either 70pkt or 15pkt.

The top panel shows the mean throughput of the flows $x(t)$, estimated by dividing the average window size by the average round trip time. For a buffer of 70pkt, theory predicts oscillations in $x(t)$; for a buffer of 15pkt, theory predicts stability. The dotted line shows the available bandwidth per flow $C$. The middle panel shows the queue size. For a buffer of 70pkt, when $x(t)$ oscillates around $C$, the queue size fluctuates markedly: when $x(t)>C$ the queue size bounces around full, and Little's Law indicates the packet drop probability is $p(t)\approx 1-C/x(t)$; when $x(t)<C$ the queue size bounces around empty and $p(t)\approx 0$. It doesn't take much change in $x(t)$ to change the queue size dramatically. For a buffer of 15pkt, $x(t)$ has small oscillations, and they do not have such a big impact; instead the queue has small and very fast fluctuations.

The bottom panel shows a sample of TCP window sizes. For a buffer of 70pkt, in periods when $x(t)>C$ and the queue is full, several flows experience drops at nearly the same time and so they become synchronized. For a buffer of 15pkt this does not happen.

Synchronization has been reported again and again in simulations, so it's no surprise to see synchronization here. Our intention in showing these simulation results is rather to illustrate the link between control theory and the mechanism of synchronization.

In summary, relatively small fluctuations in average arrival rate $x(t)$ can lead to large fluctuations in $p(t)$. This is because the queue acts as an amplifier; if $x(t) = C - \varepsilon$ then the queue empties, but if $x(t) = C + \varepsilon$ then the queue fills, for even small $\varepsilon > 0$. (This means it is hard to detect synchronization just by looking at arrival rate.) Large fluctuations in $p(t)$ induce partial synchronization between flows.

# 5. SYSTEM DESIGN

### *Buffer size.*

Consider first the small-buffer regime. Figure 2 shows the oscillations in $p(t)$ as a function of buffer size, for a single link with homogeneous flows. The figure is based on algebraic calculations, as outlined in [11] and described further in [10]. The oscillations in $x(t)$ and $p(t)$ are about a fixed point $(x^*, p^*)$ which is the solution to $dx(t)/dt = dp(t)/dt = 0$.

Figure 2 shows both $p^*$ and the extent of oscillations about $p^*$. These depend on the notional 'ideal' window size $wnd = CRTT$. When $wnd$ is small[3], the system is stable only for small buffer sizes; buffer sizes larger than 50 packets or so cause severe oscillations. When $wnd$ is large, we have more flexibility in choosing buffer size. In order that the link should accomodate varying traffic conditions we recommend that buffers should be no larger than 50 packets.

### *AQM design.*

Consider now the large-buffer regime, with GentleRED AQM. Figure 3 shows the oscillations in $p(t)$ as a function of buffer delay, again for a single link with homogeneous flows. Now that the queueing delay is non-negligible, the round trip time $RTT$ comprises propagation delay $PT$ plus queueing delay. Since queueing delay depends on the AQM scheme and the traffic mix, we have parameterized the plots by $flt = CPT$ rather than $wnd = CRTT$. When $flt$ is small we can get away with a buffer which is a quarter of the propagation delay; when $flt$ is large the buffer needs to be much bigger.

References [5, 12] describe alternatives to GentleRED, inspired by control theory, but do not specifically study the impact of buffer size. We do not know if these alternatives would be stable in the presence of large-$flt$ flows.

### *TCP re-design.*

Reference [13] considers alternatives to TCP with different window increase and decrease rules, and finds a sufficient condition for stability in a network with many links and heterogeneous round trip times, in the small-buffer regime. Suppose that the congestion window $w$ increases by $i(w)$ per ACK and decreases by $d(w)$ per drop. We replace the differential equation for TCP (1) by

$$\frac{dw(t)}{dt} = \Big(i\big(w(t)\big) - d\big(w(t)\big)p(t - RTT)\Big)\frac{w(t - RTT)}{RTT}. \quad (4)$$

[3]For even smaller $wnd$, the fluid model predicts the oscillations in $p(t)$ will be smaller—but timeouts become significant at small window sizes and the fluid model breaks down.

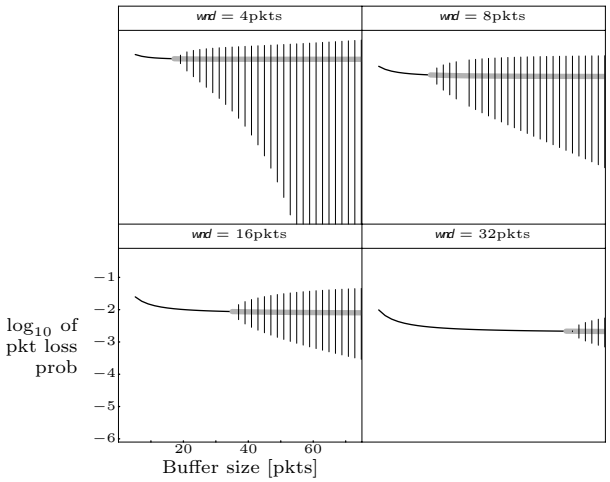


**Figure 2: Oscillations in packet drop probability $p$, as a function of buffer size. The vertical bars indicate the minimum and maximum of the oscillations, for a given buffer size; the curve indicates the fixed point $p^*$. The results depend on $wnd = CRTT$ where $C$ is the bandwidth per flow and $RTT$ is the round trip time.**

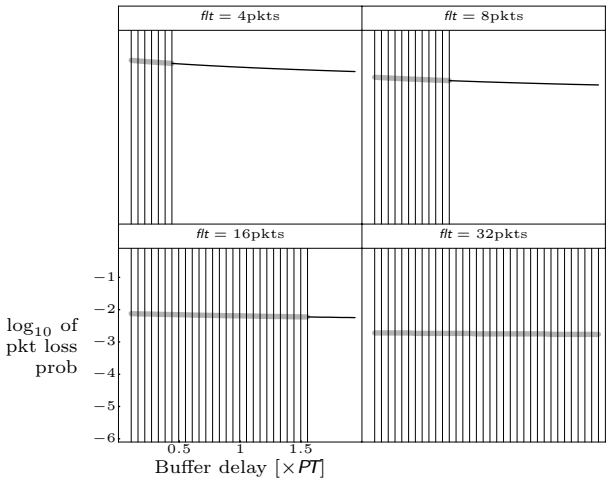


**Figure 3: Oscillations in packet drop probability $p$, as a function of buffer delay, for a queue using GentleRED. The delay is measured in multiples of the propagation delay $PT$. The results depend on $flt = CPT$, where $C$ is the bandwidth per flow.**

To find out if the network is stable, first calculate the fixed point. Suppose that the fixed point gives window size $w_r^*$ to flow $r$, and total flow $y_l^*$ on link $l$. A sufficient condition for stability is that

$$i(w_r^*)\frac{y_l^* L'_{B_l}(y_l^*)}{L_{B_l}(y_l^*)} < \frac{\pi}{2}$$

whenever flow $r$ uses link $l$. Here $B_l$ is the buffer size at link $l$. An easy way to ensure that this condition is always met is to require

$$i(w) < \Big(\sup_{y\geq 0} \frac{yL'_B(y)}{L_B(y)}\Big)^{-1}\frac{\pi}{2}.$$

A sufficient condition for this is simply $i(w) < \pi/2B$.

Interestingly, stability does not depend on $d(w)$. The decrease rule is important for determining the equilibrium window size—from (4) the window size $w^*$ for a flow experiencing drop probability $p^*$ satisfies $i(w^*)/d(w^*) = p^*$—but it is not important for determining stability.

TCP has $i(w) = 1/w$ and $d(w) = w/2$. This window increase rule is too aggressive to ensure stability when $w$ is small, and more timid than necessary when $w$ is large. This timidity has prompted work on making TCP more aggressive at large window sizes, to enable it to achieve high throughput [4, 7].

# 6. CONCLUSION

We have described a control-theoretic approach to analysing the stability of a network carrying TCP traffic. The approach consists in writing down a system of differential equations, then analysing the system to see if it's stable. A stable solution corresponds to desynchronization between TCP flows. As argued in [14], if flows are desynchronized then we can get away with very small buffers in routers.

Stability depends on buffer size in two ways. First, the order of magnitude of buffer size determines which differential equations to write down. Second, the actual value of buffer size affects the stability of the equations.

The plots in Section 5 give mixed messages. It is not possible to stabilize TCP over the full range of window sizes, either with small buffers & droptail or with large buffers & GentleRED. In the small buffer regime, buffers larger than 50 packets cause some instability for small window sizes (this is because the TCP window increase rule is very aggressive when the window is small). In the large buffer regime, flows with larger window sizes require ever larger buffer sizes. For best overall scalability, we recommend buffers be no larger than 50 packets.

The considerations described here can guide the evolution of TCP. If the window increase and decrease rules were different then the network could have small buffers and there would be no instability at all. ScalableTCP [7] was designed along these lines.

It is a topic for further research, to discover how stability is affected when there is a mix of TCP and non-TCP flows, or when some links have small buffers and other links have large buffers. It is also a topic for further research, to discover whether other AQM schemes might give better performance.


### *Acknowledgements.*

We are grateful for helpful discussions with F.P.Kelly.


# 7. REFERENCES


[1] F. Baccelli, D. R. McDonald, and J. Reynier. A mean-field model for multiple TCP connections through a buffer implementing RED. *Performance Evaluation*, 2002. Available as INRIA research report RR-4449.

[2] A. Bain. *Fluid limits for congestion control in networks*. PhD thesis, University of Cambridge, 2003.

[3] S. Deb and R. Srikant. Rate-based versus queue-based models of congestion control. In *ACM Sigmetrics*, 2004.

[4] S. Floyd. HighSpeed TCP for large congestion windows, 2003. RFC 3649, Experimental.

[5] C. V. Hollot, V. Misra, D. Towsley, and W.-B. Gong. On designing improved controllers for AQM routers supporting TCP flows. In *IEEE Infocom*, 2001.

[6] F. Kelly. Fairness and stability of end-to-end congestion control. *European Journal of Control*, 2003.

[7] T. Kelly. On engineering a stable and scalable TCP variant. Technical Report CUED/F-INFENG/TR.435, Cambridge University Engineering Department, 2002.

[8] M. A. Marsan, M. Gatetto, P. Giaccone, E. Leonardi, E. Schiattarella, and A. Tarello. Using partial differential equations to model TCP mice and elephants in large IP networks. In *IEEE Infocom*, 2004.

[9] V. Misra, W.-B. Gong, and D. Towsley. Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED. *ACM/SIGCOMM CCR*, 2000.

[10] G. Raina. Control theory and instability analysis of TCP. Technical report, Statistical Laboratory, Cambridge, 2005. To appear.

[11] G. Raina and D. Wischik. Buffer sizes for large multiplexers: TCP queueing theory and instability analysis. In *EuroNGI*, 2005. Extended version to appear in Queueing Systems.

[12] R. Srikant. *The Mathematics of Internet Congestion Control*. Birkhäuser, 2004.

[13] G. Vinnicombe. On the stability of networks operating TCP-like congestion control. In *Proceedings of IFAC World Congress on Automatic Control*, 2002.

[14] D. Wischik and N. McKeown. Part I: Buffer sizes for core routers. *ACM/SIGCOMM CCR*, 2005.