On engineering a stable and scalable TCP variant

CUED/F-INFENG/TR.435

Tom Kelly
Laboratory for Communication Engineering
Cambridge University Engineering Department
Trumpington Street
Cambridge, CB2 1PZ, United Kingdom
ctk21@cam.ac.uk

June 28, 2002

Abstract

This paper describes the engineering of a new congestion control for TCP motivated by theoretical results [21] which suggest an end-system based flow control protocol can exhibit scalable connection rates, stable operation and a decoupling between the congestion detection and response algorithms. The protocol is designed to be suitable for use in a low-loss and low-delay IP network but it could be adapted for use in current IP networks which have TCP connections that need to scale to high bandwidths on high latency links. It incorporates a packet pacing scheme, a variant of traditional slow-start, and parameter scaling to remove round trip time bandwidth allocation bias. Simulation results, in the context of a low-loss and low-delay IP network, show the strengths and weaknesses of the protocol interconnected with a simple static congestion detection algorithm at routers. It is concluded that the protocol can maintain a low-loss and low-delay packet service model and consistent throughput allocations but that more work is needed to improve link utilizations.

1 Introduction

The Internet is composed of many heterogenous networking technologies and efficiently supports the decentralized multiplexing of these resources; this ability has been instrumental in its success. Let a link in an IP network which experiences a packet arrival process with mean greater than its capacity be termed *congested*. If congestion is persistent then the link will start to drop packets when its buffers are full; this dropping is an implicit congestion signal to end-systems that is inferred by a TCP sender through timeouts, duplicate acknowledgements, and selective acknowledgements. TCP congestion control [8] is a closed loop control implemented at communicating end-systems which moderates a flow's sending rate using this implicit signaling. This control reduces the packet loss experienced at congested links and the possibility of congestion collapse that arose in the Internet when persistent congestion was left uncontrolled. An outcome of congestion control is the allocation of scarce link bandwidth between flows traversing a congested link. Congestion control has proved to be remarkably successful at providing robustness and sharing contended resources in a wide range of operational networks; so successful that it has remained largely unchanged since its introduction.

However several problems remain with TCP's current congestion control mechanisms and its operation in an increasing number of scenarios. The current TCP congestion control and packet drop signaling leads to non-negligible loss rates and high inter-packet jitter for packets traversing congested links; this can be detrimental to some applications, such as voice over IP, and hinder their deployment. Current TCP congestion control mechanisms also rely on some properties of the underlying network elements for good performance; these properties include significant buffering both in space and time, low link bit error rates, in order packet delivery, and sufficient bandwidth delay products for its self clocked window based sending scheme. Relaxing these constraints might aid designers of network elements; for example smaller buffering requirements could allow high speed router designs to use simpler buffering schemes with faster but less dense SRAM memory instead of more complex buffering architectures based on slower but denser DRAM memory; for similar reasons it might help the designers of optical packet switches. The current congestion control also introduces performance problems for connections with large round trip times and high bandwidths; a situation currently found in some scientific research networks. The problem arises because the current flow control algorithm increases the congestion window by only 1 packet per round trip time. Consider a connection with 100Mbps of available bandwidth, a round trip time of 100ms, and packets of size 1500 bytes; for such a connection it would take about 40 seconds to reach the equilibrium sending rate after a packet drop.

Explicit congestion notification (ECN), in which routers actively signal congestion by setting the ECN codepoint in the IP header[1] has been proposed in [6]. This proposal mandates that TCP primarily use ECN signaling to aid end-systems with the detection of congestion while responding in the same way to a marked packet as traditional TCP does to a dropped packet. This use of ECN provides performance advantages for short transfers where transaction rates are heavily affected by packet retransmissions and the delays imposed by packet loss detection; for a performance evaluation of the ECN-TCP proposed standard see [2]. It has been suggested [10] that protocols can make more use of ECN in order to provide a *low-loss and low-delay* packet service model with significantly reduced packet buffering requirements at network elements. A low-loss and low-delay packet service model will be taken to mean that all packets transmitted meet queues of order less than 20 packets at any queueing element and loss rates of order 0.1%; this is an ambitious performance target by which to judge an end-system based network control that aims to maintain good link utilization. Implicitly such a network would have a “better-effort” service model capable of supporting a wider range of applications without the need for complex signaling to and between heterogenous network elements or across administrative boundaries. The case for such a low-loss and low-delay congestion control framework is made from a systems perspective in [12].

This paper presents a new congestion control designed to be stable according to control theoretic modeling [21], scalable to both low and high bandwidth-delay products, and capable of operation in a low-loss and low-delay IP network framework.[2] The paper is organized as follows: Section 2 gives an overview of the mathematical models and related work upon which the new protocol is built; Section 3 discusses the implementation of the new TCP variant and a congestion detection algorithm for routers; Section 4 describes simulation results displaying the strengths and weaknesses of the protocol's design; finally a conclusion is given in Section 5.

2 Background

Mathematical modeling work [10] lead to the suggestion that a low-loss and low-delay network could be constructed if elastic traffic used appropriately designed flow control schemes and non-elastic traffic used appropriate end-system based connection acceptance control. For this to be possible it is necessary to build elastic rate control schemes such that the aggregate load arriving at each link is maintained close to, but less than, the link's capacity over sufficiently small timescales; call this *rate matching*. The problem of network rate matching on round trip timescales can be modeled by considering the properties of differential equations describing the individual source sending rates. Good rate matching would correspond to the differential equations converging quickly to an equilibrium point and displaying control theoretic stability about that equilibrium point. Theory [21] suggests that it would be possible to implement a new *scalable* congestion control in which flow rates are stable about the equilibrium point, independent of round trip time

[1]This signals congestion to the receiver and the sender learns of congestion through packets sent back to it by the receiver such as TCP acknowledgements.

[2]The design presented could be used in other datagram network architectures; for example an edge driven variant of ATM's available bit rate (ABR) service could be designed using similar methods.

and transmission rates, provided certain constraints are met independently by the routers and end-systems. The impact on flow level network performance due to the allocation of rates to different flows in a network has been studied [5] and here sharing mechanism is emphasized over sharing policy.

2.1 Related work

Several other authors have pursued work into end-system controlled networks using ECN. In [15] a marking variant and source response is designed which provides feedback encoded in the marking probability so that a network can be driven towards low-loss and low-delay operation. Such an approach applies similar design techniques to those used here, but requires a tighter coupling between the end-systems and the marking scheme through its encoded feedback scheme and has a slow response to transients at sources because of the need to average the feedback signal. Simulation results regarding the convergence and stability of an early ECN-TCP variant are presented in [13]. This paper can be seen as building on that work by considering a flow control that is scalable with different convergence properties. The adaptive virtual queue algorithm [14] achieves higher link utilizations through adaptation on timescales longer than a round trip time to observed utilizations on a link. Designing congestion detection algorithms is considered complementary to the flow control algorithm presented here and the design of such adaptive algorithms is seen as an interesting area. Parallel work [18] has lead to a flow control algorithm in the same class as that explored here with similar increase and decrease parameters arrived at through heuristic and stochastic process arguments. However that work does not tackle the problems arising in a low-loss and low-delay network and the resulting issues for a TCP implementation. In contrast to the related work, this paper aggressively pursues the goal of designing a robust protocol scalable to low and high bandwidth-delay products that is capable of delivering a low-loss and low-delay packet service in an IP network using ECN signaling.

2.2 Control theoretic modeling

A brief overview of the network modeling based around differential equation fluid limits is presented here; for a more complete overview see [10] and the results of [21]. Consider an ECN datagram network in which the probability that a packet is marked at a given link, j, is a static function, $p_j(y_j)$, of the instantaneous link load, y_j. Let each source and destination pair be identified with a route, r. Then the probability that a marked acknowledgment is received at the source of route r at time t is

$$P_r(t) = \sum_{j \in r} p_j \left(y_j \left(t - d_2(j, r) \right) \right) \tag{1}$$

where the sum[3] is over the marking probability at each link forming route r, the return delay from link j for the acknowledgment via the destination of route r is denoted by $d_2(j, r)$, and the load at link j is given by

$$y_j(t) = \sum_{s: j \in s} x_s \left(t - d_1(j, s) \right) \tag{2}$$

Here $d_1(j, r)$ is the forward delay from the source on route r to the link j. Notice that on each route, r, the round trip time, T_r, is given by[4]

$$T_r = d_1(j, r) + d_2(j, r) \qquad \forall j \in r \tag{3}$$

In datagram networks with window based flow control schemes (such as TCP) a window, $cwnd_r(t)$, is maintained by each source and is the maximum number of transmitted packets that

[3]In the treatment presented here this sum is an approximation of the marking probability on a route that requires marking probabilities at each link to be small or dominated by a single bottleneck. These results also hold for a more precise product form without this restriction but are omitted because the analysis is less concise.

[4]It is assumed that the rates arriving at links are matched to their capacity so that queueing delays are small and hence negligible. No assumptions are made about routing apart from that it remains static on flow timescales; asymmetry is permissible.

can be awaiting acknowledgment. Suppose that for each unmarked acknowledgment received the window, $cwnd_r(t)$, is incremented by $a_r cwnd_r^n(t)$, while for each marked acknowledgment received the window is decremented by $b_r cwnd_r^m(t)$, where $m > n$. Let $x_r(t) = \frac{cwnd_r(t)}{T_r}$ be a continuous approximation of the sending rate, then the sources satisfy the following set of differential equations

$$\frac{d}{dt} cwnd(t) = \frac{a_r cwnd^n \left(1 - P_r(t)\right) - b_r cwnd^m P_r(t)}{T_r / cwnd(t - T_r)} \tag{4}$$

or in terms of rates

$$\begin{aligned} T_r \dot{x_r}(t) = x_r\left(t - T_r\right) \big(a_r \left(x_r(t) T_r\right)^n \left(1 - P_r(t)\right) - \\ b_r \left(x_r(t) T_r\right)^m P_r(t)\big) \end{aligned} \tag{5}$$

For such a system the equilibrium point[5] would be

$$\hat{P}_r = \frac{a_r (\hat{x}_r T_r)^n}{a_r (\hat{x}_r T_r)^n + b_r (\hat{x}_r T_r)^m} \tag{6}$$

It has been shown [21] that for heterogeneous round trip times and arbitrary topologies this set of equations is stable about its equilibrium provided there exists γ such that

$$\begin{aligned} a_r \left(\hat{x}_r T_r\right)^n < \frac{1}{\gamma} \qquad & \forall r \in R \\ \frac{\hat{y}_j p_j'(\hat{y}_j)}{p_j(\hat{y}_j)} \leq \gamma \qquad & \forall j \in J \end{aligned} \tag{7}$$

where $\hat{x}_r$ and $\hat{y}_j$ are the equilibrium rates of each connection and at each link respectively, J is the set of all links, and R the set of all routes. If $n = 0$, $m = 1$, and γ is an agreed global constant then there is a natural decoupling of the congestion detection and response algorithms. It is argued that such decoupling is valuable as it offers a mechanism for providing differentiated services through an end-system's setting of a_r and b_r and allows the evolution of congestion detection schemes to be loosely coupled to end-system protocols.

Notice, as also observed in [21], that Equation 7 highlights a possible trade-off between convergence times and utilization rates of a stable system with a static congestion detection algorithm; for a high utilization it is necessary for $p_j(\hat{y}_j)$ to be low when $\hat{y}_j$ is close to capacity. This will lead to a high $p_j'(\hat{y}_j)$ at this point because $p_j(y_j)$ must reach 1 as y_j approaches capacity.[6] In turn a high value of $p_j'(\hat{y}_j)$ requires γ to be sufficiently large and then a small a_r is needed to give stability; this reduces convergence speeds. There is thus a trade-off in achieving the design goals of fast system convergence during transient periods and system stability at equilibrium. The stability condition in Equation 7 suggests choosing $n = 0$ would lead to a *scalable* control as local stability at equilibrium would not depend on the window size of individual connections. It is this control that is used as the basis of the variant designed.

2.3 Window process modeling

In [17] a class of ECN-TCP responses is considered in terms of the stochastic process formed by each flow's congestion window, $cwnd(t)$, in isolation. The arguments presented consider the average window size, window variance, average number of packets marked in each round trip time, and flow relaxation times; it concludes that the same parameters of $n = 0$ and $m = 1$ are the most promising from this class of controls. Furthermore it is shown that the coefficient of variance of the instantaneous sending rate for the scheme proposed would be

$$CoV\left(x_r\right) = CoV\left(\frac{cwnd_r}{T_r}\right) \sim \sqrt{\frac{b_r}{2}} \tag{8}$$

[5]Note that global convergence to equilibrium has not been proved but appears reasonable in most cases with the constraint $m > n$ and given the global convergence results for similar systems [11].

[6]This is so that the equilibrium point in Equation 6 is always admissible.

provided $P_r \downarrow 0$. This is of particular interest here because it suggests that for high sending rates the instantaneous sending rate's coefficient of variance is controlled primarily through the single parameter b_r.

The convergence speed is of significance in an elastic rate protocol that must deal with network conditions that include sudden route changes, short transfers[7], or highly bursty non-elastic sources. Consider a source sending with window, $cwnd_r$. Suppose that at time t_0 a sudden overload shock occur then $P_r \uparrow 1$ and the source will reduce its sending rate upon receiving feedback by a factor of $\frac{1}{2}$ in less than[8]

$$\frac{log(\frac{1}{2})}{log(1-b_r).cwnd_r(t_0)} \text{ round trip times.} \tag{9}$$

By comparison traditional ECN-TCP congestion control would immediately reduce its window by a factor of $\frac{1}{2}$ on receiving feedback; but would also limit it's window reduction by at most a factor of $\frac{1}{2}$ per round trip time. In response to a sudden increase in the available capacity on a route, $P_r \downarrow 0$, and the time taken for the source to increase its sending rate by a factor of 2 is less than

$$\frac{1}{a_r} \text{ round trip times.} \tag{10}$$

By contrast traditional ECN-TCP congestion control would require $cwnd_r(t_0)$ round trip times to respond to the increase in available capacity. The scalable TCP scheme responds more effectively to changes in available capacity when window sizes are large and this enables it to operate in high bandwidth-delay product environments such as high-speed long haul networks. In small window scenarios ECN-TCP can adapt more rapidly but Equation 7 suggest this might be at the expense of system stability; while the ability of TCP to share resources well in small window scenarios is a known problem [16].

3 System design

The control theoretic modeling in Section 2 is based on a fluid flow model of connections and attempts to model the interaction between packet arrivals at a link and the congestion detection algorithm using a static feedback function $p_j(y_j)$. In an IPv4 network discrete packets of data are sent between source and sink, the ECN standard provides only a binary congestion signal per packet, and there is traffic present which is not subject to closed loop control; the implementation of the control must be robust to this.

3.1 A static ECN marking scheme

The following virtual queue based marking scheme will be used; it is a hybrid of the first virtual queueing schemes suggested [7] and the random early marking proposal [4]. It is not argued that it is the optimal congestion detection algorithm merely that it is simple and easily implemented. Each link in the network has a capacity C bps, a FIFO buffer of size B bits and a virtual queue which is drained continuously at rate θC. When a packet arrives to be forwarded on a link the packet is marked with probability

$$1 - e^{-\frac{\phi b}{s}} \tag{11}$$

where b is the current size of the virtual queue in bytes, s is the predominant packet size and ϕ sets the marking scheme's *effective buffer size*, $\frac{1}{\phi}$.

After the packet marking decision is made the packet's size is added to the virtual queue and the real packet is added to the real FIFO queue for forwarding at the link's speed C. Packets arriving at the real FIFO queue which can not fit into its buffer of size B are discarded; it is

[7]As predominantly found in many world wide web transfers, DNS lookups, instant messaging applications or cache validation protocols.

[8]This assumes that the overload causes no loss or delay effects which would result in further rate reductions for window based flow control schemes which require acknowledgements to release packets.

intended that packet arrivals at the link are controlled so that this event is rare and the real FIFO queue remains empty most of the time. Furthermore the virtual queue size, b, is constrained so that $b \in [0, 30 * s]$; this ensures that queue dynamics are kept as fast as possible in comparison to round trip times.

In order to get an intuition for how to set the parameters for this particular queueing scheme consider the following two different packet arrival models; a simple Markov Chain and a more sophisticated Brownian motion analysis.[9] Assuming that packet arrivals form a Poisson process with parameter y while service from the virtual queue forms a Poisson process of intensity θC. Then by the PASTA property for a Poisson process the probability that an arriving packet is marked is given by

$$\begin{aligned} p(y) &= \sum_{i=0}^{\infty} \left(\frac{y}{\theta C}\right)\left(1 - \frac{y}{\theta C}\right)\left(1 - e^{-\phi i}\right) \\ &= \frac{y\left(1 - e^{-\phi}\right)}{\theta C - y e^{-\phi}} \end{aligned} \qquad (12)$$

Providing $y < \theta C$, Equation 7 gives stability for

$$\frac{y p'(y)}{p(y)} = \frac{\theta C}{\theta C - y e^{-\phi}} < \frac{1}{1 - e^{-\phi}} \leq \gamma \qquad (13)$$

This suggests that setting ϕ higher will allow for a smaller γ and so a more aggressive gain at end-systems.

Consider now an alternative traffic model. Suppose that the work arriving at the link over a time period τ is Gaussian, with mean $y\tau$ and variance $y\tau\sigma^2$. With such a distribution of arriving traffic the probability that an arriving packet is marked for a load of $y < \theta C$ is

$$p(y) = \frac{\phi y \sigma^2}{\phi y \sigma^2 + 2(\theta C - y)} \qquad (14)$$

In the form of the stability condition of Equation 7 this is

$$\begin{aligned} \frac{y p'(y)}{p(y)} &= \frac{1}{1 - \frac{y}{\theta C}\left(1 - \frac{\phi \sigma^2}{2}\right)} \\ &< \begin{cases} \frac{2}{\phi \sigma^2} & \text{if } 1 - \frac{\phi\sigma^2}{2} \geq 0 \\ 1 & \text{if } 1 - \frac{\phi\sigma^2}{2} < 0 \end{cases} \end{aligned} \qquad (15)$$

Notice that the Brownian model gives an intuition as to how the burstiness of traffic might affect the stability of the system. If the traffic arriving at a link is more bursty σ^2 will be higher. This will make the bound in Equation 15 tighter allowing a lower γ in Equation 7 which allows stability across a larger range of a_r values; some bursts in the traffic aid stability but at the expense of lower utilization.

Throughout the analysis and design of this queueing algorithm it has not been argued that it is ideal but merely that it is simple, intuitive and achieves good performance. Designing active queue management schemes that perform well with a wide variety of end-system responses and traffic patterns remains an active area for research.

3.2 Bandwidth allocation and response curves

One difference between the scalable TCP and conventional ECN-TCP is their respective sending rates for a given level of congestion signaling. This relationship for the sending rate x_r in terms of

[9] To ease analysis the constraint $b < 30 * s$ is removed. It is assumed that this does not alter the qualitative insight of the analysis.

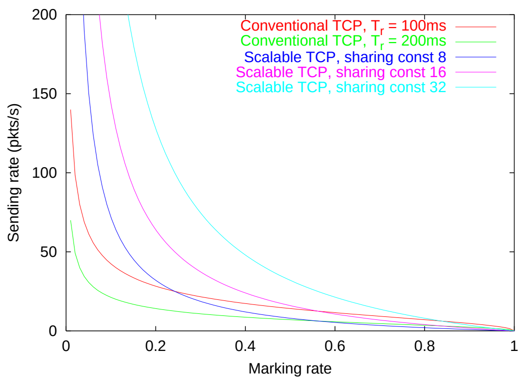

Figure 1: Response curves for conventional TCP and scalable TCP.

the signaling rate P_r is called the *response curve.* One model of the conventional TCP response curve is

$$x_r = \frac{s\sqrt{2}}{T_r}\sqrt{\frac{1-P_r}{P_r}} \tag{16}$$

where s is the segment size of the packets sent. In contrast the response curve for the scalable TCP derived from the equilibrium point in Equation 6 is

$$x_r = \frac{s.a_r}{T_r b_r}\frac{1-P_r}{P_r} \tag{17}$$

Notice that the curves have a different form for the multiplicative function of P_r. Also prevalent in both schemes is a factor of $\frac{1}{T_r}$ giving a bias towards connections with short round trip times. When queueing delays are significant with respect to propagation and serialization delay, round trip times of 200ms-400ms become common during congestion.[10] In this case the short round trip time connections would get a rate allocation twice that of their longer counterparts. In a regime where queueing delays are small, say a maximum of 10ms, the spread in round trip times becomes 10ms-210ms and the short round trip time connections will receive a share about 20 times greater than the longer round trip connections. Hence within a low-loss and low-delay network it becomes increasingly difficult to defend the bias which the conventional TCP response curve has towards connections with short round trip times. Setting $b_r = \frac{baseb_r}{T_r}$ removes the dependence on round trip time in the variant's response curve and makes explicit the resource allocation to a flow through the packet size and the fraction $\frac{a_r}{baseb_r}$, call this fraction the *sharing constant.* Comparisons of the response curve for various sharing constants and the TCP response curve at round trip times 200ms and 100ms are given in Figure 1. Notice that traditional TCP requires a very small marking rate in order to sustain a high sending rate with large round trip times and does not make significant use of higher marking rates.[11] Throughout the rest of this paper we will assume that all connections use the same packet size and sharing constant.[12] The use of a common sharing constant introduces a small technicality regarding b_r which in practice cannot exceed 1; indeed too violent a reduction in $cwnd_r$ in response to a mark could be detrimental to performance. Mathematically this is displayed in Equation 8 which links the point sample variance of the rate to b_r. Hence in the implementation b_r is capped to be less than 0.1 and a_r is scaled down to ensure that the sharing constant is correct for the current round trip time.[13] This

[10]Assumes buffers are provisioned to a bandwidth-delay product and that the longest connection has a propagation delay of 200ms

[11]This is a necessity if packet drop is the only signal available.

[12]Policing, interconnection complexities, and the economic implications of resource sharing policies are left untreated here; although the mechanisms provided are consistent within a congestion pricing and settlement framework which might provide sufficient incentives to address these issues.

[13]This is consistent with the condition needed for stability in equation 7.

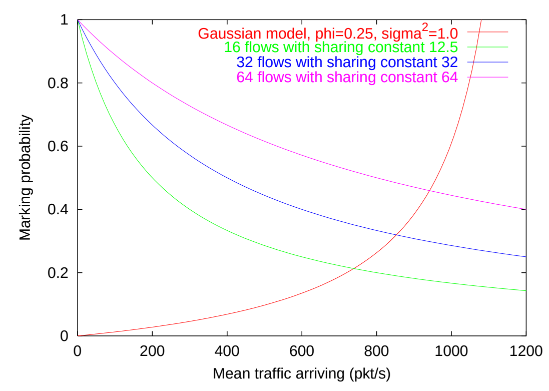

Figure 2: The effect of flow numbers on marking and utilization rates for a Gaussian traffic model.

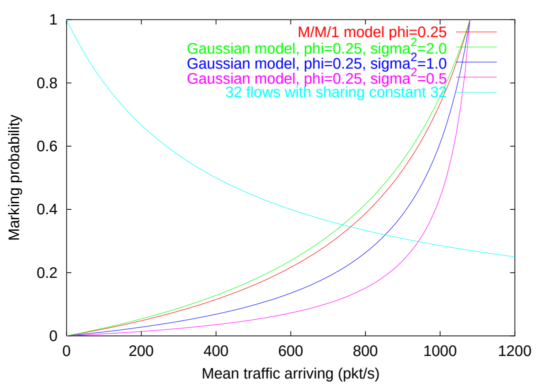

Figure 3: The effect of different traffic distributions on marking and utilization rates for a given number of flows.

mechanism of *b-capping* was introduced as a quick method to improve performance with small round trip times and the choice of optimal threshold value is an open question.

By plotting the response curves and marking probability functions with respect to the mean load arriving at a bottleneck link it is possible to get a feel for how the system's parameters will affect utilization. Suppose that there is a 10Mbps link, a packet size of 1000 bytes, and a sharing constant of 12.5 for all flows. If the arrivals at the link were Gaussian with $\sigma = 1.0$ then the utilizations are given as the point at which the aggregate response curve intersects the marking function's response to load as shown in Figure 2. Consider now the effect of the traffic characteristics on utilizations and marking rates; Figure 3 show the various utilizations and marking rates for different values of σ^2 in the Gaussian model along with those for a simple M/M/1 model. Notice how less burstiness results in higher utilizations but this is at the cost of slower system convergence since a_r must be set lower to ensure stability; a tradeoff between the system's ability to converge quickly in the presence of transients and the ability to maintain a high utilization.

Throughout the stability analysis it was assumed that the traffic pattern for a given load at a given link is invariant to the parameters of the flows, such as a_r, b_r and T_r, traversing that link. This assumption correspond to the use of a static function $p_j(y)$ to represent signaling response in Section 2. In reality there is a complex interaction between the source parameters, number of sources and marking schemes employed at routers which makes getting a grip on $p_j(y)$ more difficult. Ideally it would be desirable if the sending protocol could be designed such that the traffic it generated on superposition of any combination of flows with varying parameters generated a predictable packet arrival process (e.g. similar to the superposition of multiple Poisson processes). Such ideal *traffic normalization* may be impossible to achieve but some level of normalization may still be possible. Guided by the two different queueing models given earlier some rough estimates can be used to set a_r and ϕ coupled through γ. Suppose the system wished to use a convergence speed given by $a_r = \frac{1}{8}$ then setting $\phi = \frac{1}{4}$ would be consistent with an implied γ in the range $[4.5, 8.0]$ for a Poisson packet arrival at queues or a Gaussian arrival process with $\sigma^2 \geq 1.0$ and an implied γ of 8.

3.3 Changes to TCP

The most relevant points of TCP congestion control implementation are outlined here to provide a baseline for changes; for a more complete treatment see [8]. The sender can have a window of sent but not yet acknowledged packets in flight with the size of this window being the minimum of the congestion window, $cwnd_r$, and the receiver's available receive buffer. The receiver acknowledges the receipt of data back to the sender indicating successful transmission and advertising its receive buffer.[14] Upon receiving an acknowledgement, the sender immediately sends as much data as available and permitted by the window. Since the data will arrive at the receiver according to the route's available bandwidth, the acknowledgements received at the sender will be spaced accordingly. The use of the acknowledgements as the trigger for sending more data implicitly makes the scheme's sending rate *self-clocking* and avoids the need for fine grained timers in the sender. The scheme is made elastic by altering $cwnd_r$ according to whether packets are dropped/marked on their way to the receiver; in steady state the increase is $\frac{1}{cwnd_r}$ and the decrease is $\frac{cwnd_r}{2}$. Various enhancements to this basic framework have been made including methods for signaling multiple packet drops in a single window of data, various methods for determining the initial value of $cwnd_r$ at connection setup, and alternative algorithms for sending packets while recovering from a packet drop.

Several elements of this basic algorithm are unsuited to low-loss and low-delay operation. Primarily the use of self-clocking and its interaction with small window sizes. For example in a low-loss and low-delay network round trip times will be of the order of 50ms within a continent such as Europe, while regional connections will be of the order of 10ms. Such low round trip times will result in very small window sizes of the order of less than three 1KB packets for connections of up to 500Kbps with a round trip time of up to 50ms. At window sizes of around three packets the fast retransmit algorithm used to detect packet loss quickly in TCP performs badly. This leads to course grained timeouts and poor multiplexing of resources as some connections stabilize with large windows while the majority experience very poor throughput and many timeouts [16]. The use of ECN and ECN-TCP help to mitigate some of these small window effects and aid fairness but do not go far enough in considering the effects of fractional windows and instead use course grained backoffs. There are reasons to believe that such small window problems will affect an increasing number of TCP bytes transfered and total connections; lower round trip times can be expected to result from increasing link speeds reducing serialization times, improved end-system processing speeds leading to shorter delays in protocol stacks, and content distribution or caching networks pushing content closer to the end-system. With this premise it is argued that future flow control protocols will inevitably have to introduce fine grained timers to pace their transmission at accurate fractions of round trip times. In the past such timers were expensive to implement but now several implementation of fine grained timers for use in network protocol stacks exist. Soft-timers [3] are an operating system based approach to building microsecond based timers; process cross talk and CPU caching effects in such an approach suggest that there might be benefits to running timers and scheduling tasks on the network interface card when possible [19].

The use of rate pacing timers is not without its disadvantages. Work on TCP pacing [1] concluded that pacing often has significantly worse throughput than regular TCP because it is susceptible to synchronized losses and it delays congestion signals. This is consistent with Equation 15 which suggests that stabilizing a system where the elastic flows have smooth sending rates requires the closed loop gains to be lower. In this paper the performance of rate pacing timers is improved by scaling the inter-packet pacing by a clamped exponentially distributed random variable. The addition of randomness adds some burstiness to large connections making them appear more like the super-position of many small connections; in the Gaussian model presented earlier this has the effect of increasing σ^2 allowing for faster system convergence by increasing a_r.

The implementation of the rate pacing timers in the variant is as follows. Each endpoint maintains a congestion window, $cwnd_r$, as a real number which implicitly determines the current sending rate as $\frac{cwnd_r}{\overline{RTT}_r}$ pkts/s. Each packet acknowledgement contains congestion feedback of the segment it acknowledges; if no congestion was signaled then $cwnd_r \mapsto cwnd_r + a_r$ otherwise if congestion was observed then $cwnd_r \mapsto (1 - b_r) * cwnd_r$. Let $\overline{RTT}_r$ be the mean round trip time seen in the last $\max\{10, 4\lceil cwnd_r \rceil\}$ segments acknowledged. Timers are set to expire at $X\frac{\overline{RTT}_r}{cwnd_r}$ since the last packet transmission where X is taken from a clamped exponential distribution with

[14]Many receiver's delay acknowledgements in order to send batched acknowledgements to reduce reverse path traffic. Delayed acknowledgements could be added to the protocol described although such an extension would need to provide feedback of each segment's congestion experience and timestamp; care would be needed to ensure that the added noise and time lag did not introduce oscillations to the system.

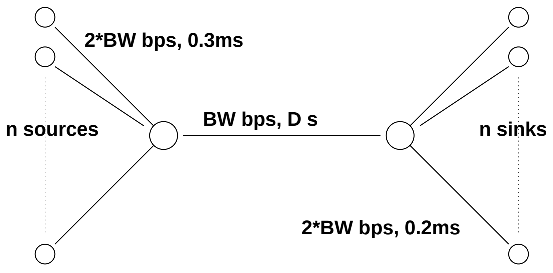

Figure 4: Topology for single round trip time simulations.

lower bound $\frac{1}{cwnd_r}$ and upper bound 1. Changes to $cwnd_r$ due to acknowledgements reschedule the timer but X is not resampled. When the timer fires a single packet is released if permitted by the minimum of $\lceil cwnd_r \rceil$ and the receiver's advertised window. The rounding up of the window provides for fractional windows and smooth rate speedups. Maintaining the window for final packet sending decisions maintains robustness in the case of a sudden change in network conditions or with legacy networks which only support packet drop. In the case of a packet drop the implementation falls back to traditional TCP detection mechanisms and congestion responses. When the round trip time estimator, $\overline{RTT}_r$, is recalculated the parameters a_r and b_r are adjusted to ensure that $b_r < 0.1$ while maintaining the same sharing constant; as discussed earlier this b-capping is to bound the variance in sending rate while maintaining consistent bandwidth allocations.

Most of the discussion so far has dealt with the stability of an elastic rate connection that is close to its equilibrium sending rate. During connection start up a TCP connection must find its operating point for the route it is using. Traditionally slow-start achieved this goal by increasing a connections $cwnd_r$ by 1 for each acknowledgement received until the first loss is encountered. This has the effect of approximately doubling the sending rate each round trip time.[15] Instead an experimental scheme termed *soft-start* is used. Here the TCP variant increases $cwnd_r$ by $a_r 2^{ssgrace}$ on each unmarked acknowledgement and decreases $cwnd_r$ by $b_r cwnd_r$ and $ssgrace$ by 1 on each marked acknowledgement. $ssgrace$ is initialized to 3 and the connection leaves soft-start when $ssgrace$ reaches 0. This provides a grace of three marked packets for the connection to experience inflated window growth before the connection changes over to steady state congestion control. This scheme aims to allow connections to reach their operating point faster given the higher signalling rates permitted by ECN marking. A more in depth study of soft-start compared to slow-start in a variety of operating scenarios is an area for further work; this is of importance in high bandwidth long haul networks where the convergence times could be lengthy. The use of soft-start does not preclude other methods to initialize $cwnd_r$ such as those based on historical route caches or control block sharing [20].

4 Simulation results

The marking algorithm and protocol changes to TCP in Section 3 were implemented in the discrete event network simulator *ns-2b8*.[16] Various validation tests were conducted to ensure protocol correctness; although some experience of the robustness of the system to common implementation errors is important as well!

Many of the results here are qualitative in nature and aim to capture the advantages of a network using stable flow control in terms of a number of performance metrics such as utilization, queueing delay size and variation, and packet drop rates. Queueing delay and variation has two effects; it determines whether certain applications can meet latency requirements and implicitly

[15]End-systems implementing delayed acknowledgements during slow-start will increase their sending rate by a factor of 1.5 instead.

[16]Available from `http://www.isi.edu/nsnam/ns/`

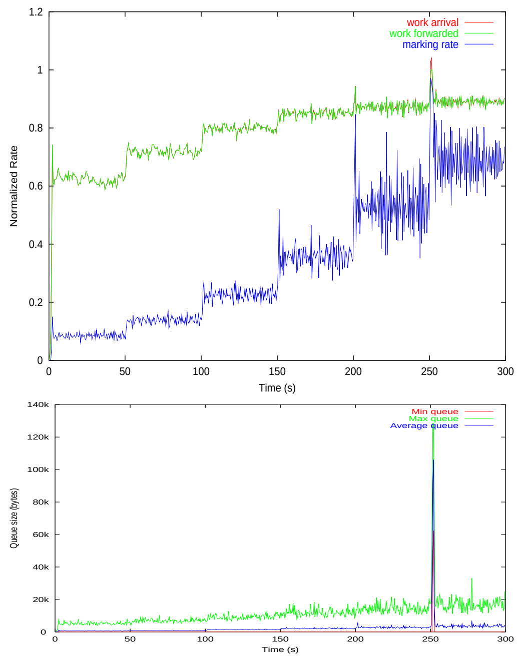

Figure 5: Round trip time 256ms: Work arrival, work forwarded, and marking rate (top); queue (bottom)

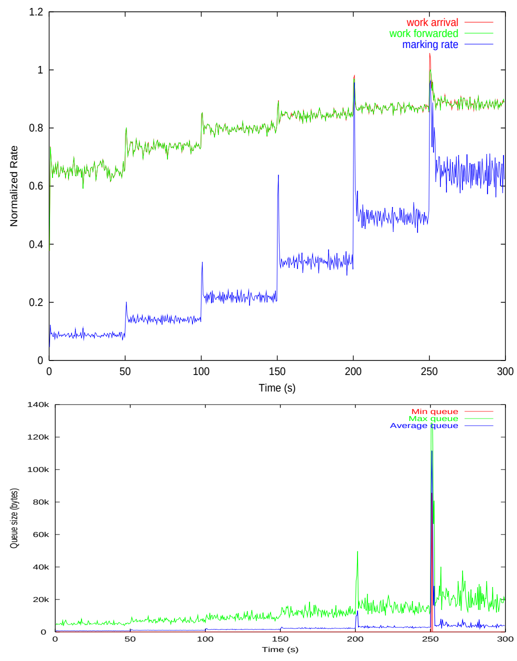

Figure 6: Round trip time 16ms: Work arrival, work forwarded, and marking rate (top); queue (bottom)

acts as a measure of queue occupancy with implications for buffer provisioning. Here the mean queueing delay and variance are the metrics used to provide insight on performance suitable for these two requirements. Packet drop rates measure raw loss performance which is harmful to many real-time applications and can lead to congestion collapse.

4.1 Basic operation

The simulations results shown in Figures 5 and 6 attempt to show the basic operation of the flow control protocol interconnected with the static congestion detection scheme described in Section 3. These simulations used the topology shown in Figure 4 with round trip times of 16ms and 256ms and bottleneck bandwidth of 30Mbps. Initially there were 16 connections in each direction started uniformly over the first second. Each 50 seconds the number of connections doubled in both directions with start times picked uniformly over a second; thus the number of connections went from 16 to 512 in each direction exercising connection bandwidths from 1.8Mbps through 50Kbps. The queue state is shown as the minimum, maximum, and average queue size over 1 second intervals. The work arrival, work forwarded, and marking rates are also averages taken over 1 second intervals.

The virtual queue marking scheme described in Section 3 was used running at 90% of real capacity, a ϕ of 0.125, and a real packet buffer of size 128KB. The sources all used a $baseb_r$ of 0.01 and an a_r of 0.125; the theory suggests this system would be just stable.

Comparing the results for both the 256ms and 16ms shows how the flow control scales over different round trip times with the marking rates, utilizations, and queuing delays being very similar. The 256ms results display more noise on the marking rate due to the smaller number of round trip times over which these measures are averaged. Notice how the queue size is kept small and below 20 packets even when under heavy load excluding the transients of synchronized connection arrivals. Interestingly the queue performance under heavy load is slightly worse for the 16ms connections compared to the 256ms connections due to the small fractional windows

Number of flows	Marking rate	Utilization	Queue delay μ (s)	Queue delay std dev.	Flow goodput μ (bps)	Goodput fairness
1	0.0181	0.418	4.78e-05	9.98e-05	1.141e+07	1
4	0.0491	0.561	8.4e-05	0.000146	3.825e+06	0.9989
8	0.0807	0.657	0.000126	0.000197	2.241e+06	0.9989
16	0.134	0.749	0.00019	0.00027	1.277e+06	0.9986
32	0.222	0.819	0.000287	0.000381	6.978e+05	0.9989
64	0.358	0.862	0.000401	0.000513	3.672e+05	0.9987
128	0.531	0.874	0.000512	0.000669	1.862e+05	0.9972
256	0.678	0.889	0.000665	0.000892	9.472e+04	0.9925
512	0.786	0.906	0.000845	0.00117	4.825e+04	0.982

Table 1: Network and flow performance metrics for different numbers of connections.

used by the 16ms connections; for the last 50 seconds the average window for the 256ms and 16ms connections is of the order of 0.1 packets and 1 packet respectively.[17]

4.2 Operating range

Here we present simulation statistics for some steady state scenarios. In these simulations the round trip propagation delay was set to 100ms and the simple bottleneck topology in Figure 4 was used. All simulations used a virtual queue operating at 90% capacity (i.e. $\theta = 0.9$) and a real buffer of size 128KB. In each simulation the access link had an extra propagation delay displacement chosen uniformly from [0ms, 0.25ms] added to avoid simulation artifacts due to exactly homogeneous round trip times. Connections were started uniformly distributed over the first 10 seconds of simulation with equal numbers of forward and backward connections. The statistics were collected for 60 seconds after a further 25 seconds of simulation so that the system was in steady state.

These simulations had a bottleneck link speed of 30Mbps and the queue ϕ was set to 0.125. The number of connections was varied but all had a set to 0.125, a $baseb$ of 0.005 and segment size of 1000 bytes.

These simulations attempt to display the effectiveness of the system to a range of offered loads by changing the number of connections. Table 1 displays network and flow level performance statistics for the 60 seconds during which data was collected. As can be seen the flow fairness, measured by Jain's fairness index [9], is good across a range of flow rates. The rate achieved is in line with that expected from the theoretical response curve; for a marking rate of 0.018 and 0.79 the response curve suggests a goodput of $1.1e7$bps and $5.3e4$bps respectively. The mean queueing delay and jitter is also kept below 5 packet serialization times. Throughout the simulations no packet drops were observed. The utilizations can be poor when the aggregate demand is low and there is room for improvement here.

4.3 Effect of packet size

Simulations were conducted to see what effect packet size had on performance. Decreasing packet size would be expected to improve stability and multiplexing at the expense of increased packetization overhead. In these simulations the bottleneck link speed was 30Mbps, the number of connections was 40, a was set to 0.125, while segment sizes[18], s, were chosen from $250, 500, 1000, 1500$. The parameter $baseb$ was chosen equal to $0.05 * \frac{s+50}{100*1050}$ so that each connection had the same response curve across all packet sizes. The marking schemes at queues used a ϕ set to 0.125. These choices of ϕ and a were guided by the same arguments given for M/M/1 and Gaussian models in

[17] A congestion window of $cwnd < 1$ corresponds to a packet being sent once every $\frac{1}{cwnd}$ round trip times.

[18] Excluding TCP and IP headers; headers added 50 bytes of overhead with the TCP timestamp option was used to measure the round trip time.

Segment size (bytes)	Marking rate	Utilization	Queue delay μ (s)	Queue delay std dev.	Flow goodput μ (bps)	Goodput fairness
250.0	0.284	0.85	9.46e-05	0.000123	4.554e+05	0.9997
500.0	0.269	0.842	0.00017	0.000222	5.261e+05	0.9992
1000.0	0.26	0.838	0.000321	0.000419	5.714e+05	0.9986
1500.0	0.258	0.837	0.000475	0.000614	5.881e+05	0.9979

Table 2: Network and flow performance metrics for different packet sizes.

Section 3. All other simulation parameters were unchanged from those for the operating range simulations.

Table 2 shows aggregate statistics taken over the whole of the 60 seconds during which data was collected; throughout these simulations no packet drops were observed. It can be seen that the noise of the system reduces as the segment size decreases from 1500 bytes to 250 bytes. However with 50 bytes of additional TCP and IP header overhead it appears that these variability gains must be offset against increasing overhead inefficiency as seen in the reduction of flow goodput. It could be argued that a packet size of 500 bytes offers a good tradeoff between connection goodput efficiency and variability. Reduced segment size improves absolute jitter performance as the number of packets in the queue is what affects stability not their size.[19]

4.4 Round trip time fairness

Figure 7 displays the dynamic operation of the system with heterogenous round trip times. These simulations used the topology shown in Figure 4 with the delays of the right hand set of links changed so that connections had round trip times from {4ms, 8ms, 16ms, 32ms, 64ms, 128ms, 256ms, 512ms} and a bottleneck link speed of 30Mbps. Initially there was a total of 16 connections in each direction started uniformly over the first second and over all round trip times. Each 50 seconds the number of connections doubled in both directions with start times picked uniformly over a second and with round trip times uniform across those available; thus the number of connections went from 16 to 512 in each direction exercising connection bandwidths from 1.8Mbps through 50Kbps. The queue state is shown as the minimum, maximum, and average queue size over 1 second intervals. The work arrival, work forwarded, and marking rates are also averages taken over 1 second intervals. The parameter settings were the same as those used in the basic operation simulations.

Notice that the marking rates, utilizations, and queue dynamics are roughly the same as those seen for the single round trip time case shown in Figures 5 and 6; this invariance is desirable because of the predictability it introduces. The throughput achieved by the connections in each round trip time class is roughly equal. However the dynamics while the system stabilizes is dependent on the round trip time since those with short round trip times can react faster than those with longer round trip times.

Figures 8 and 9 show the normalized throughput of twenty five connections coming from five different groups. These results are for a bottleneck link of 30Mbps with equal reverse path traffic. Connections were started uniformly distributed over the first 10 seconds and goodput was measured for 60 seconds from the 30th simulation second. The ECN-TCP simulations did not use delayed acknowledgements and the bottleneck used adaptive RED with a target queueing delay of 20 packets and ECN marking. The scalable TCP variant simulations used the same parameters as the simulations used to display the scheme's basic operation. Traditional ECN-TCP displays significant variation in contended bandwidth allocation; the connections with a round trip time of 16ms get more than 10 times the goodput of the connections with a round trip time of 256ms. The TCP variant is designed to remove this bias toward small round trip times and the simulation

[19] These results suggest that the approach would perform extremely well in an ATM environment where cell sizes are only 53 bytes.

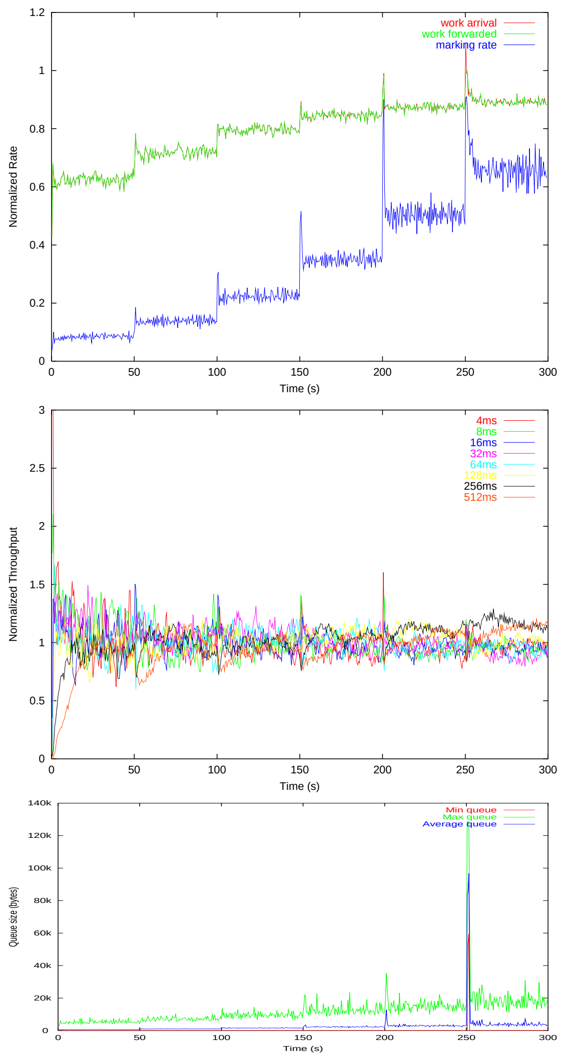

Figure 7: Heterogenous round trip times; Work arrival, work forwarded, and marking rate (top); normalized throughput by round trip time (middle); queue samples (bottom).

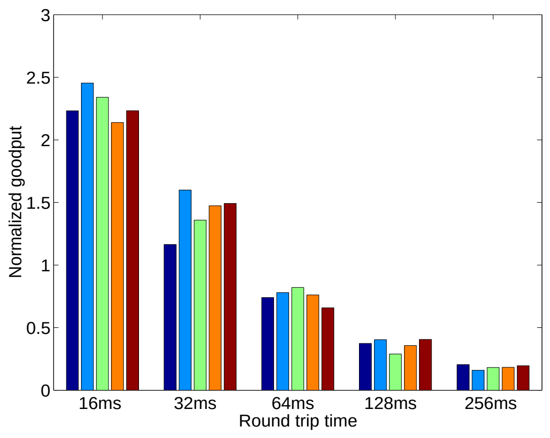

Figure 8: Normalized throughput of 25 ECN-TCP connections with heterogenous round trip times and adaptive RED.

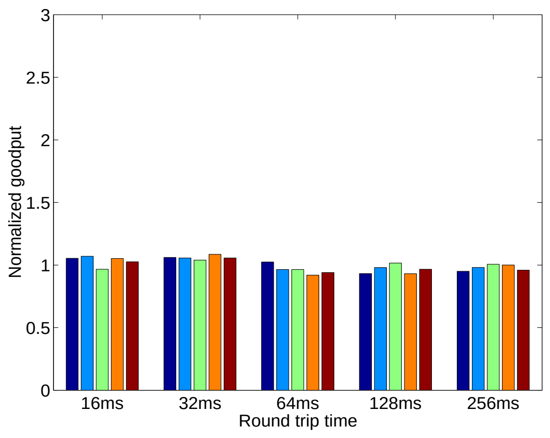

Figure 9: Normalized throughput of 25 TCP variant connections with heterogenous round trip times and static congestion detection algorithm.

results display the invariance of the bandwidth allocation to round trip time.

5 Conclusion

This work attempts to show the viability of a low-loss and low-delay network using the IP ECN signalling standard, relatively small changes to TCP congestion control and a simple static congestion detection algorithm. The protocol changes to TCP involve implementing a packet pacing scheme, altering slow-start to deal with higher signalling rates, changing the congestion window update algorithm, and using parameter scaling to ensure fairness in throughput between connections with different round trip times.

The design was studied through simulation; the results suggest that this end-point based flow control can indeed maintain low-loss and low-delay with fairness between round trip times. However more work remains to improve utilizations when the demand is low with adaptive marking schemes appearing the most promising direction for further research.

6 Acknowledgments

This work was done while a Royal Commission for the Exhibition of 1851 Industrial Fellow at AT&T Research Labs, Cambridge. Thanks go to Frank Kelly, Andy Hopper, and Sally Floyd for many useful discussions. Thanks also to Glenn Vinnicombe and all those in the Laboratory for Communication Engineering for their support.

References

[1] A. Aggarwal, S. Savage, and T. Anderson. Understanding the performance of TCP pacing. In *IEEE INFOCOM 2000*, Tel-Aviv, Israel, March 2000.

[2] U. Ahmed and J. Hadi Salim. Performance evaluation of explicit congestion notification (ECN) in IP networks. *Internet RFC 2884*, July 2000.

[3] M. Aron and P. Druschel. Soft timers: Efficient microsecond software timer support for network processing. *ACM Transactions on Computer Systems*, 18(3):197–228, August 2000.

[4] S. Athuraliya, V. H. Li, S. H. Low, and Q. Yin. REM: active queue management. In *Proceedings of the 17th International Teletraffic Congress*, September 2001.

[5] T. Bonald and L. Massoulie. The impact of fairness on network stability and performance. In *ACM SIGMETRICS 2001*, Cambridge, Massachusetts, June 2001.

[6] S. Floyd, K. Ramakrishnan, and D. Black. The addition of explicit congestion notification (ECN) to IP. *Internet RFC 3168*, September 2001.

[7] R. J. Gibbens and F. P. Kelly. Resource pricing and the evolution of congestion control. *Automatica*, 35:1969–1985, 1999.

[8] V. Jacobson. Congestion avoidance and control. *SIGCOMM Symposium on Communication Architectures and Protocols*, pages 314–329, 1988. An updated version is available via `ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z`.

[9] R. Jain, D. Chiu, and W. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Technical report, DEC Research Report, September 1984.

[10] F. P. Kelly. Models for a self-managed internet. In *Philosophical Transactions of The Royal Society*, volume A358, pages 2335–2348. The Royal Society, August 2000.

[11] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research Society*, 49:237–252, 1998.

[12] T. Kelly. The case for a new IP congestion control framework. Technical Report CUED/F-INFENG/TR.434, Laboratory for Communication Engineering, Cambridge University, June 2002.

[13] P. Key, K. Lavens, and D. McAuley. An ECN-based end-to-end congestion-control framework: experiments and evaluation. Technical Report MSR-TR-2000-104, Microsoft Research, October 2000.

[14] S. Kunniyur and R. Srikant. Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management. In *ACM SIGCOMM 2001*, San Diego, California, August 2001.

[15] S. H. Low, F. Paganini, J. Wang, S. Adlakha, and J. C. Doyle. Dynamics of TCP/RED and a scalable control. In *IEEE INFOCOM 2002*, June 2002.

[16] R. Morris. TCP behavior with many flows. In *IEEE International Conference on Network Protocols*, Atlanta, Georgia, 1997.

[17] T. Ott. ECN protocols and the TCP paradigm. In *Workshop on Modeling of Flow and Congestion Control Mechanisms*. Ecole Normale Supérieure, September 2000.

[18] T. J. Ott and A. Misra. ECN and 'meekly aggressive' flows. Under submission, 2001.

[19] I. Pratt and K. Fraser. Arsenic: A user-accessible gigabit ethernet interface. In *IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001.

[20] J. Touch. TCP control block interdependence. *Internet RFC 2140*, April 1997.

[21] G. Vinnicombe. On the stability of networks operating TCP-like congestion control. In *15th IFAC World Congress on Automatic Control*, Barcelona, Spain, July 2002.