Resource pricing and the evolution of congestion control *

R. J. Gibbens and F. P. Kelly
Statistical Laboratory
University of Cambridge

Draft version

Abstract

We describe ways in which the transmission control protocol of the Internet may evolve to support heterogeneous applications. We show that by appropriately marking packets at overloaded resources and by charging a fixed small amount for each mark received, end-nodes are provided with the necessary information and the correct incentive to use the network efficiently.

Keywords: charging, critical congestion interval, game theory, Internet, proportionally fair pricing, rate control, sample path shadow prices.

1 Introduction

In the current Internet, the rate at which a source sends packets is controlled by TCP, the transmission control protocol of the Internet [12], implemented as software on the computers that are the source and destination of the data. The general approach is as follows [3]. When a resource within the network becomes overloaded, one or more packets are lost; loss of a packet is taken as an indication of congestion, the destination informs the source, and the

*Correspondence: Frank Kelly, Statistical Laboratory, University of Cambridge, 16 Mill Lane, Cambridge CB2 1SB, UK. Email: f.p.kelly@statslab.cam.ac.uk . This is a preliminary draft—the latest version is available at `http://www.statslab.cam.ac.uk/~frank/evol.html` .

source slows down. The TCP then gradually increases its sending rate until it again receives an indication of congestion. This cycle of increase and decrease serves to discover and utilize whatever bandwidth is available, and to share it between flows.

The approach has worked well in the past, when most flows have implemented reasonable versions of TCP, producing broadly similar bandwidth allocations for flows sharing similar resources. But the approach is breaking down, for two related reasons. First, there is an incentive to modify the TCP algorithm so that it strives more aggressively for a larger share of available bandwidth, or even to avoid using any form of congestion control. Secondly, applications are becoming more heterogeneous, with widely differing, and constantly evolving, statistical characteristics and sensitivities to delay.

Floyd and Fall [7] note that an increasing deployment of traffic lacking end-to-end congestion control may cause congested links to occupy themselves sending packets that will only be dropped later in the network, and they describe how this could lead to congestion collapse in the Internet. They observe [7] that it is no longer possible to rely on end-nodes to use end-to-end congestion control, or on developers to incorporate end-to-end congestion control in their applications. This observation, together with the heterogeneity of applications, has motivated work (reviewed in [3, 7, 11]) on various measurement and scheduling mechanisms that might be implemented within the network itself to restrict the bandwidth of flows and to discriminate between the services that are provided to different users.

The aim of this paper is to explore a different approach. Our premise is that if the resource implications of their actions can be made known to end-nodes, then the end-nodes themselves are best placed to determine what should be their demands upon the resources of the network. The issue at stake here is important in both theory and practice. The optimal allocation of resources depends upon the utilities the various users attach to their several flows through the network, as well as upon the properties of the resources within the network. It is possible that it is easier to achieve an efficient allocation by conveying information on congestion from the network to intelligent end-nodes rather than by requiring users to classify their flows into predefined categories and conveying this information from users to the network. Certainly the development of ATM traffic classes [29] illustrates some of the drawbacks of an approach that requires the definition of a set of service categories before the applications that might use these categories have been developed or have become widespread.

Previous work [16] has shown that, if users' utilities are concave functions of their attained throughput, then their aggregate utility is maximized by the network allocating scarce network resources according to a weighted *proportional fairness* criterion: loosely, the network shares resources in proportion to how much the users choose to pay. It has also been shown [17] that a weighted proportionally fair allocation could be achieved by simple rate control algorithms, using increase and decrease rules similar to those described by Chiu and Jain [2] and Jacobson [12] and implemented in TCP. Crowcroft and Oechslin [5] have proposed ways of setting parameters of the TCP protocol to achieve weighted proportional fairness, and have presented results from simulations and prototypes. This work has demonstrated the possibility of a charge-aware TCP, with congestion control *parameters* alterable by end-nodes in an incentive-compatible manner. In this paper we take a step further: we investigate the possibility that end-nodes or developers may be allowed uninhibited access to the *algorithms* used for congestion control while still maintaining incentive-compatibility.

MacKie-Mason and Varian [19] have described a *smart market* approach to allocating resources in a network, where a price is set for each packet depending upon the level of demand. In this paper we show that the benefits of a smart market, in terms of efficient allocation of scarce resources, may be achieved by very simple network mechanisms, involving the setting of just a single bit to mark some packets; essentially we show that the intelligence necessary to run a smart market may be decentralized to users' rate control algorithms, where it may take the form of variants of existing TCP algorithms. For a network with cooperative end-nodes, the marks provide the information the end-nodes need to make efficient use of the network. For a network with potentially uncooperative end-nodes, a fixed small charge for each mark ensures that end-nodes have the correct incentive, as well as the necessary information, to use the network efficiently.

Axelrod [1], in his famous study of the iterated prisoner's dilemma, described a computer tournament where players could submit strategies which then played against each other: see Ridley [25] for a readable review of later work in this area, and of its relevance for theories of evolutionary biology. In some respects a network operating with the mechanisms described in this paper resembles Axelrod's tournament: for example, the performance of a rate control algorithm will depend upon the current population of rate control algorithms, and the interaction and evolution of different sub-populations is of great interest. The numerical results reported in this paper have been obtained from a computational environment implemented in the Java object-orientated programming language [1], and we expect competitions using such environments to provide an important stimulus to further theoretical work

[1]For a period the environment is accessible at
http://www.statslab.cam.ac.uk/~richard/research/topics/evolution .

in this area.

The organization of the paper is as follows. In the next Section we use a simple slotted model of a resource to illustrate how network shadow prices may be identified, at least statistically, on the sample path describing load on the resource: the relevant material from [17] is reviewed in an Appendix. In Section 3 we study how these sample path shadow prices may be used to transfer information and incentives to end-nodes, and we investigate the behaviour of a system comprising a resource and end-nodes equipped with various transmission control algorithms designed to achieve different user objectives.

The marking mechanism associated with the slotted model of Sections 2 and 3 is very simple; essentially a resource marks every packet arriving to an overloaded slot. In Section 4 we define sample path shadow prices for a more realistic queueing model of a resource. For a queue the critical congestion interval lies between the start of a busy period and a packet loss within the same busy period; these intervals correspond to the overloaded slots in the slotted model, and packets arriving during these intervals should, ideally, be marked. But for a queue it is, unfortunately, often not possible to be sure whether or not a packet should be marked until some time after the packet has left the resource. In Section 4 we describe several marking mechanisms which attempt to approximate the ideal behaviour, and describe further investigations with various sub-populations of transmission control algorithms. In Section 5 we compare and contrast the mechanisms defined in this paper with those of the current Internet. In Section 6 we describe briefly some analytical models which complement those of [17], before concluding in Section 7.

2 Sample path shadow prices

In this section we explore a very simple resource model, in order to motivate the experiments of the next section. The aim of our discussion is to show that the shadow price of a resource, the key variable of the model of the Appendix, is straightforward to identify, at least statistically, on the sample path describing the load on the resource.

Suppose that time is slotted, and that a resource has capacity per slot to cope with N equally sized packets, with any excess lost. Let the load upon the resource per slot, Y, be generated by adding together a number of independent Poisson random variables $X_1, X_2, \ldots, X_m$, with means $x_1, x_2, \ldots, x_m$ respectively. Then Y has a Poisson distribution with mean $y = \sum_1^m x_r$. Let the cost to the system be the number of packets lost. Then the expected cost per slot $C(y)$, is given by

$$\begin{aligned} C(y) &= \mathbb{E}(Y-N)^+ \\ &= \sum_{n\geq N}(n-N)e^{-y}\frac{y^n}{n!} \end{aligned}$$

and a simple differentiation establishes that $p(y)$, defined by

$$p(y) = \frac{d}{dy}C(y),$$

satisfies

$$\begin{aligned} p(y) &= \sum_{n\geq N} e^{-y}\frac{y^n}{n!} \\ &= \mathbb{P}\{Y \geq N\}. \end{aligned}$$

We interpret $p(y)$ as the *shadow price* of the resource: it is the marginal increment in expected cost at the resource for a marginal increment in load.

Suppose next that whenever the number of packets arriving in a slot exceeds N, a mark is placed on *each* of these packets. (We shall not distinguish between packets which are lost and those which are merely marked: we treat them all as marked packets.) Conditional on the event $Y = n$, X_r has a binomial distribution with parameters n and x_r/y. Thus the expected number of marks per unit time placed on packets from the r^{th} flow is

$$\begin{aligned} \mathbb{E}(X_r I\{Y > N\}) &= \sum_{n>N}\mathbb{E}(X_r \mid Y = n)\mathbb{P}\{Y = n\} \\ &= \sum_{n>N}\frac{x_r}{y}ne^{-y}\frac{y^n}{n!} \\ &= x_r\sum_{n\geq N}e^{-y}\frac{y^n}{n!} \\ &= x_r p(y). \end{aligned}$$

Thus, for Poisson statistics, the marking of every packet when a resource is overloaded produces an expected charge per unit of flow, $p(y)$, precisely equal to the shadow price at the resource, and the expected charge per unit time to flow r, $x_r p(y)$, is precisely the fair charge to this flow under the model of the Appendix.

Now for statistics more general than Poisson, we cannot expect such precise identities. In particular, under more general flow statistics, the resource implications of additional flow may not be easily summarized by variation of a single real number, such as the rate of a Poisson process. Yet we can show that the key relationship, between the expected increase in system cost caused by a given load increment, and the expected charge to that load increment, is more fundamental.

Suppose that the load Y on the resource is a positive random variable, and that we wish to assess the impact of an additional load, X, where X is a non-negative random variable, not necessarily independent of Y. Then the increase in the number of packets lost is

$$[X+Y-N]^+ - [Y-N]^+ = XI\{X+Y>N\} - (N-Y)I\{X+Y>N>Y\}.$$

Thus

$$\mathbb{E}[X+Y-N]^+ - \mathbb{E}[Y-N]^+ \leq \mathbb{E}(XI\{X+Y>N\}). \qquad (1)$$

Further, if N, Y and X are integral, and if the additional load is a *small* increment, satisfying $\mathbb{P}\{X=0 \text{ or } 1\}=1$, then the event $\{X+Y>N>Y\}$ is impossible, and we have that

$$\mathbb{E}[X+Y-N]^+ - \mathbb{E}[Y-N]^+ = \mathbb{E}(XI\{X+Y>N\}). \qquad (2)$$

Thus, for *small* increments, we have our desired identity between the expected increase in system cost caused by the additional load and the expected charge to the additional load. It is worth emphasizing that the inequality (1), and the consequent equation (2) for small increments, do not require any distributional assumption on the increment X, not even independence of the load Y.

Note that expression (11) is the charge attached to the *additional* load X: the user responsible for X may also face an increased charge for any other load, already included within Y, due to that user (we shall discuss this point further in Section 6.2).

It is natural to define the *sample path shadow price* of a packet to be one if the packet's deletion from the sample path describing arrivals at the resource would result in one less packet drop at the resource; by marking *all* packets arriving in overloaded slots we ensure that a packet is marked if and only if its sample path shadow price is one.

3 Experiments with the slotted model

Our first experiments concern the simple slotted model described in the previous section, where the resource has capacity per slot to cope with N packets, with any excess lost. Again suppose the load upon the resource, Y, in a given slot, is generated by adding together the loads generated in that slot, $X_1, X_2, \ldots, X_m$, by m users,

$$Y = \sum_1^m X_i,$$

and let the number of marks fed back to source i at the end of the slot be

$$X_i I\{Y > N\}.$$

Later, in Section 5, we shall discuss the mechanism by which this feedback may be delivered. Note that the feedback delay, between the generation of packet and the receipt of feedback concerning that packet, is just one slot: we shall also discuss more general feedback delays in Section 5.

The loads produced in successive slots by a user may depend upon the earlier feedback of marks by the network as well as upon characteristics of the user. In this section we describe some simple models of a user and examine the interaction between users and a resource.

Elastic-user(w)

Elastic-user(w) transmits

$$X(t) = \lfloor x(t) + z(t) \rfloor^+$$

packets in the slot $(t, t+1)$, where $x(t)$ and $z(t)$ are internal state variables updated as follows:

$$\begin{aligned} z(t+1) &= x(t) + z(t) - X(t) \\ x(t+1) &= x(t) + \kappa\,(w - f(t))\,. \end{aligned} \tag{3}$$

Here $f(t)$ is the number of marks received at the end of slot $(t, t+1)$ and κ is a small positive constant. We interpret $x(t)$ as the rate of the user, and $z(t)$ as a fraction of a packet held over until the next slot. The recursion (3) attempts to stabilize the rate $x(t)$ around a value where the expected charge per slot is w: observe that

$$w - \frac{1}{T}\sum_{t=0}^{T-1} f(t) = \frac{x(T) - x(0)}{\kappa T},$$

an expression that will generally become negligible as T increases.

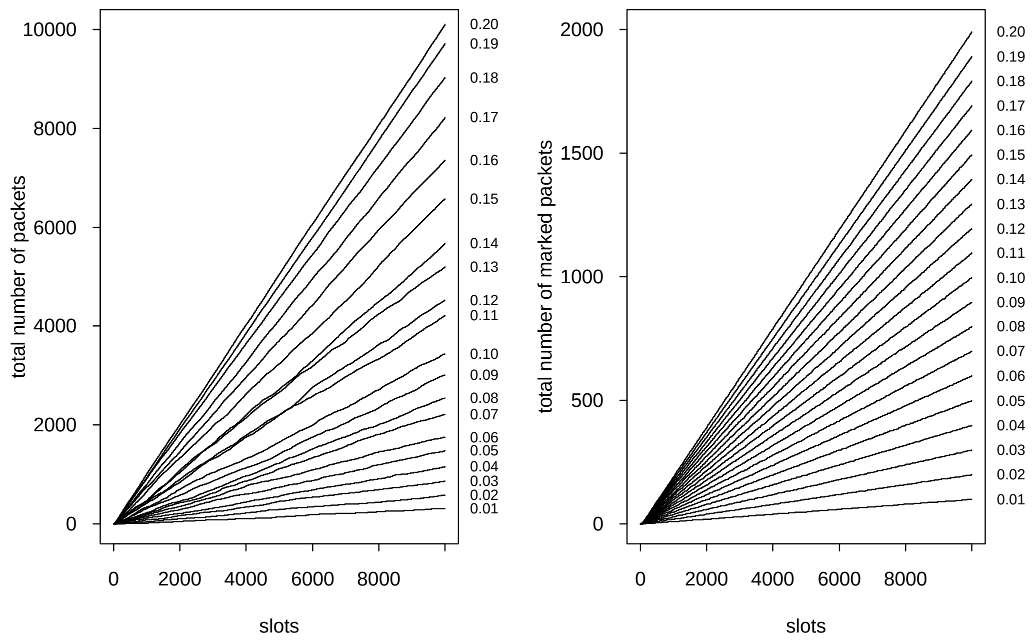

Figure 1: Scenario 1. Twenty users share a single slotted resource. The first panel describes the throughput achieved, and the second panel the charges incurred, by each of the users: the numbers on the right of each panel label users according to their choice of the parameter w.

Scenario 1

In our first scenario we simulated a resource with a capacity to serve $N = 10$ packets per slot, handling 20 users. The i^{th} user, $i = 1, 2, \ldots, 20$, behaved as Elastic-user(w_i), where $w_i = i \times 0.01$ and $\kappa = 0.1$.

Figure 1 shows the throughput achieved and the charges incurred by the 20 users. The second panel shows that the users' algorithms managed to fulfil their respective aims of a charge per unit time of w_i, $i = 1, 2, \ldots, 20$. The first panel shows that by varying its parameter w a user can influence its throughput. In general the throughput achieved by a user will depend on the level of aggregate demand from other users as well as the choice of w by the user; but given the level of aggregate demand, we see that the throughput of a single user is highly responsive to that user's choice of w.

It is interesting to note from Figure 1 that the user with $w = 0.2$ achieves a throughput of approximately one packet per slot, while a throughput of 0.5

packets per slot requires a choice of approximately $w = 0.13$. The latter user is more difficult per packet for the resource to multiplex, and consequently pays more per packet. To develop this point further, suppose the load due to a user is $X \in \{0, 1\}$, and Y is the total load due to all users. Then the expected charge to the user responsible for X is

$$\mathbb{E}(XI[Y > N]) = \mathbb{P}\{X = 1, Y > N\} = \mathbb{P}\{Y > N \mid X = 1\}P\{X = 1\},$$

and so the expected charge per packet to the user responsible for X is $\mathbb{P}\{Y > N \mid X = 1\}$. This conditional probability is somewhat lower for the user who submits a regular stream of packets, one per slot, than for the user who submits packets intermittently.

No initial conditions for Elastic-user(w) have been specified: Figure 1 was constructed from a stationary realization of the process.

We have seen that Elastic-user(w) may be viewed as representing a user who chooses to pay an amount w per unit time for a share of the resources of the network. A more general Elastic-user($U(\cdot)$) could be defined, as in the Appendix, in terms of a general utility function $U(\cdot)$: such a user might behave as Elastic-user(w) but with a time-varying $w = w(t)$ given by equation (13). Courcoubetis *et al.* [4] describe a framework within which the relevant information about the utility function $U(\cdot)$ may be inferred from the choices made by an application or a human user.

The Appendix formalizes the result that if each user makes an individually rational choice of its parameter w, perhaps varying w in accord with its experience of the network and its own requirements, then the system as a whole operates efficiently; here efficiency is defined in the economic sense that the share of resources allocated to each user and the overall level of resource utilization are such as to maximize the aggregate utility of the entire system.

For our simple model of a single resource, with feedback received at the end of each slot, the choice of the gain parameter κ is relatively unproblematic: a higher value of κ will result in a higher variance at equilibrium, but will allow a more rapid convergence to equilibrium. In a more complex network the variance and speed of convergence depend upon the delays between resources and end-nodes [17], and care will be needed if time-lags are large and uncertain. We return to this point in Section 5.

Scenario 1 describes a rather static environment: next we consider the effect of fluctuations in demand for the resource.

Intermittent-user(w)

Intermittent-user(w) is active and behaves as Elastic-user(w) for a random period with mean 1000, and then sleeps (that is, transmits no packets) for a random period with mean 4000. Successive periods are independent and geometrically distributed. Following the end of each sleep period, the internal state $(x(t), z(t))$ is reset to its value at the end of the last active period.

Scenario 2

Our second scenario explored the behaviour of the system with intermittent users. Again we simulated a resource with capacity to serve $N = 10$ packets per slot but now handling 100 users. There were five independent copies of Intermittent-user(w_i), for each of the twenty values of w_i used in Scenario 1.

Figure 2 shows various aspects of the stationary behaviour of the system. The first panel shows the demand W, defined as the sum of the w's over those users not sleeping. Observe that this panel shows the demand fluctuating between about half and twice its mean value, of about 2, over the period illustrated. The average number of packets transmitted per slot, shown in the second panel, is much less variable. The third panel shows the the number of marked packets per slot, calculated as a moving average over 50 slots. The third panel is thus a moving average of the shadow price at the resource; note that the average shadow price tracks the demand W shown in the first panel.

The set of users who are active fluctuates, but given those users who are active, their relative throughputs and marking rates are much as illustrated in Figure 1. We conclude from this experiment that as the demand on the resource fluctuates users manage to share the resource between them, keeping the total throughput approximately constant despite relatively large variations in the demand.

The percentage of packets lost, shown in the fourth panel, fluctuates in step with, but at lower levels than, the percentage of packets marked. A larger capacity than 10 packets per slot would lead to a lower packet loss probability and a lower ratio of lost to marked packets. In Section 4 we shall study a queueing model roughly comparable to a slotted model with a capacity of about 100 packets per slot: the queueing model will achieve a much lower packet loss probability.

Next we define a user with a rather different behaviour.

File-transfer(F, W)

This user has a file of size F to transfer, an amount W to spend, and wants to transfer the file as soon as possible. We shall describe a simple algorithm which attempts to satisfy the requirements of such a user. The algorithm tries to pay a price W/F per packet on average. If the average shadow price is currently higher than this, the algorithm waits until the shadow

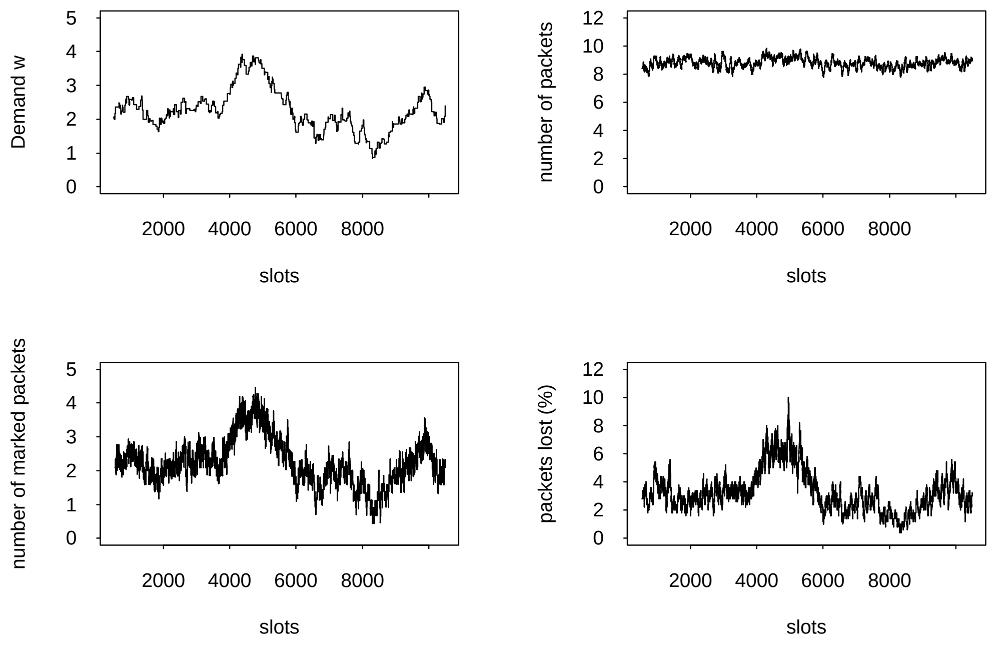

Figure 2: Scenario 2. One hundred users share a single slotted resource. Each user alternates randomly between active and sleep periods, and this produces the fluctuating demand W for the resource shown in the first panel. The second panel shows a moving average, calculated over 50 slots, of the number of packets transmitted per slot. The third panel shows the number of marked packets per slot, and the final panel the percentage of packets lost, again calculated as moving averages over 50 slots.

price decreases, otherwise the algorithm increases its rate. We shall see that File-transfer(F, W) generates a form of bistable behaviour, in contrast to the smoother behaviour of Elastic-user(w).

More precisely, the algorithm is based on that of Elastic-user(w), but with an adaptive choice of w. At time t, let $F(t)$ be the size of the file remaining to be transferred, and let $W(t)$ be W less the number of marks so far received. The algorithm maintains two further state variables $x(t), z(t)$, both initially set to zero. It calculates a parameter $w(t)$ according to the following equation:

$$w(t+1) = \max\{x(t)W(t)/F(t), w_{min}\} \tag{4}$$

and updates $x(t), z(t)$ and the number of packets transmitted $X(t)$ just as Elastic-user(w), but with w in equation (3) replaced by $w(t)$.

We can motivate equation (4) as follows. At a rate $x(t)$ the remainder of the file would be transferred in a time $F(t)/x(t)$: at this rate the user could afford to pay an amount $w(t+1) = W(t)x(t)/F(t)$ per unit time. This would produce an increase or a decrease in the rate $x(t)$ according as the amount the user has to pay per packet, $W(t)/F(t)$, is greater than or less than the currently perceived shadow price per packet, $w(t)/x(t)$. Thus the algorithm attempts to vary its rate $x(t)$ so as to achieve a price per packet of about $W(t)/F(t)$. The lower limit, w_{min}, in equation (4), allows the algorithm to occasionally retest the network following periods when it appears too expensive to transfer the file.

Scenario 3

Our third scenario reran the set of 100 users from scenario 2, but with an additional population of 10 users making file transfers. Specifically, the 10 additional users were File-transfer(F, W_j), for $j = 1, 2, \ldots, 10$, where $F = 1000$ and $W_j = 200 + 20j$, all starting at slot 0.

The first panel of Figure 3 shows the progress of the file transfers. Those with the higher values of W_j tend to complete sooner, although there is not a strict order. The second panel shows the shadow price at the resource, and should be compared with the third panel of Figure 2: the additional population in scenario 3 increases the shadow price at the resource at times when file transfers are particularly active. The amounts spent by the eight completed file transfers were equal or less than their respective budgets (although the algorithm does not guarantee this).

The file transfers labelled 280 and 320 were caught by sudden surges in demand caused by other file transfers (observable in the first panel of

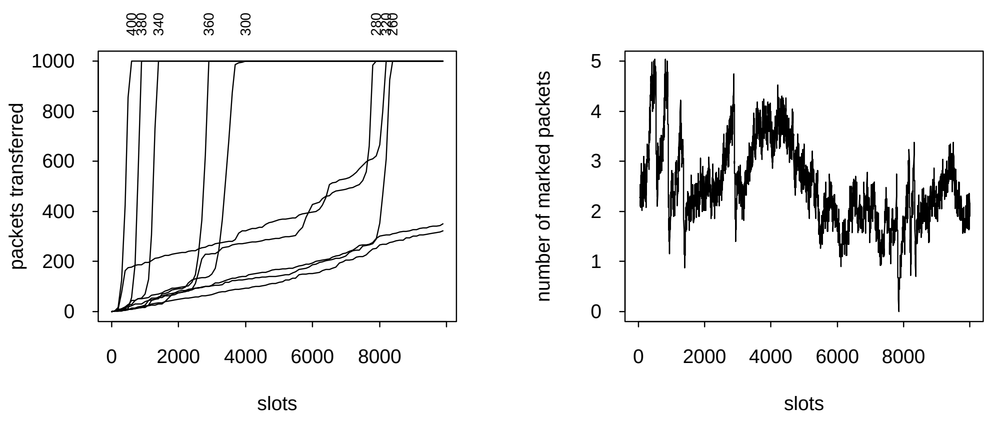

Figure 3: Scenario 3. In addition to the 100 users whose demand is summarized in the first panel of Figure 2, there are 10 users attempting file transfers starting at slot 0. The first panel shows the progress of the ten file transfers, labelled by the amounts the users are prepared to pay. The second panel shows a moving average, calculated over 50 slots, of the number of packets marked per slot.

Figure 3) and the more gradual variation in demand caused by intermittent users (observable in the first panel of Figure 2): the delayed file transfers sensed that they could no longer afford the increased shadow price, and were forced to wait until later to complete.

The lower limit, w_{min}, in equation (4), was set equal to 0.01, the lowest value of w used in the first scenario. This forces users to spend at least 0.01 per unit time, on average. In consequence, File-transfer(F, W) will eventually complete the file transfer, no matter how small the value of W, at a cost that may exceed W. If each time the lower limit were crossed the value of the lower limit were halved, then the resulting algorithm would produce exponentially growing backoff intervals following periods when it appeared too expensive to transfer the file.

Of course the shadow price may fluctuate over the time taken to transfer the file. One might, for example, multiply the right hand side of equation (4) by a factor less than one, to protect against the possibility that the shadow price will drift upwards during the course of the file transfer. With this alteration, file transfers are slower to start, but less likely to be interrupted. It may also be in the interest of the user (as well as the network) for the algorithm to place an upper limit on $w(t)$ to protect the user from paying more than necessary, since if $w(t)$ becomes too large the user is simply competing against itself (we return to this point in Section 6.2).

These and many other alterations to the algorithms Elastic-user(w) and File-transfer(F, W) are of course possible. Our aim in this paper is not to suggest that these algorithms are optimal, but rather to argue that the feedback of marks allows an algorithm to take into account its knock-on effects on other users, and thus provides an environment which encourages the evolution of efficient algorithms.

4 The queueing model and experiments

In this Section we extend the simple model described in Section 2 to allow a queue with finite buffer. Let Y_{t-1} be the number of packets that arrive at the resource in the interval $(t-1, t]$, and let Q_t be the queue size at time t. Then the recursion

$$Q_t = \min\{N, Q_{t-1} - I\{Q_{t-1} > 0\} + Y_{t-1}\}$$

describes a queue with a buffer capacity of N that is able to serve a single packet per unit time; the number of packets lost at time t is

$$[Q_{t-1} - I\{Q_{t-1} > 0\} + Y_{t-1} - N]^{+}.$$

We shall define a busy period to end at time t if $Q_{t-1} = 1$, and a busy period to begin at time t if $Q_{t-1} \leq 1$ and $Q_t \geq 1$. Thus two consecutive busy periods may abut, in our discrete time model.

The impact of an additional packet upon the total number of packets lost is relatively easy to describe. Consider the behaviour of the queue *with* the additional packet included in the description of the queue's sample path. Then the additional packet increases the number of packets lost by one if and only if the time of arrival of the additional packet lies within a *critical congestion interval*, defined as a period between the start of a busy period and the loss, within the same busy period, of a packet; otherwise the additional packet does not affect the number of packets lost. Critical congestion intervals correspond to the overloaded slots of the slotted model, and packets arriving during these intervals should, ideally, be marked. Unfortunately it will often be difficult to determine the shadow price of a packet while the packet is passing through the queue; it will, in general, be unclear whether or not the current busy period will end before a packet is lost.

But it is possible to develop marking mechanisms which approximate the ideal behaviour. The first mechanism we describe marks all the packets in the queue at the time of a packet loss, and then marks a sufficient number of immediately subsequent packets to ensure that, in total, the correct number of packets are marked. By this means the mechanism ensures that the proportion of packets marked at a resource is equal to the probability that the removal of a randomly chosen packet would reduce by one the number of packets lost at the resource. This is the mechanism we used to construct Scenario 4, below.

A slight variant on the above mechanism is to mark every packet leaving the queue from the time of a packet loss until the queue becomes empty. This variant does not require the queue to keep a counter of how many subsequent packets it needs to mark, and produces numerical results which are almost identical to those described below. (For insight into why this might be so, we note that if the queue size were a reversible stochastic process, and if a busy period were to contain packet losses, then over this busy period the number of packets arriving at the queue prior to the last packet loss would have the same distribution as the number of packets departing the queue subsequent to the first packet loss, [14].)

We repeated the earlier experiments with a queueing resource which has a buffer capacity of $N = 10$. If a packet is marked then the mark is fed back to the source 100 time units after the packet was transmitted by the source: this is intended to model transmission delay as well as queueing delay. Note that for the queue it takes ten time units, rather than one slot, to serve ten packets: to maintain rough comparability the earlier parameters w_i were reduced by a factor of 10 and the mean active and sleep periods were increased by a factor of 10; to cope with the larger feedback lag (100 time units, rather than one slot), the parameter κ was reduced by a factor of 100. The results were broadly similar to those reported in Figures 1–3: a lower proportion of packets were lost. We conclude that our earlier discussion of simple user types and their interaction extends to the case where the resource is a queue. There are, however, several new issues that need to be explored concerning the marking mechanism at the resource. First we introduce a new user type.

Unresponsive-user(g)

While active this user transmits a packet in each slot with probability g, independently from slot to slot; while sleeping, this user transmits no packets. Successive active and sleep periods are independent and geometrically distributed with mean 1000.

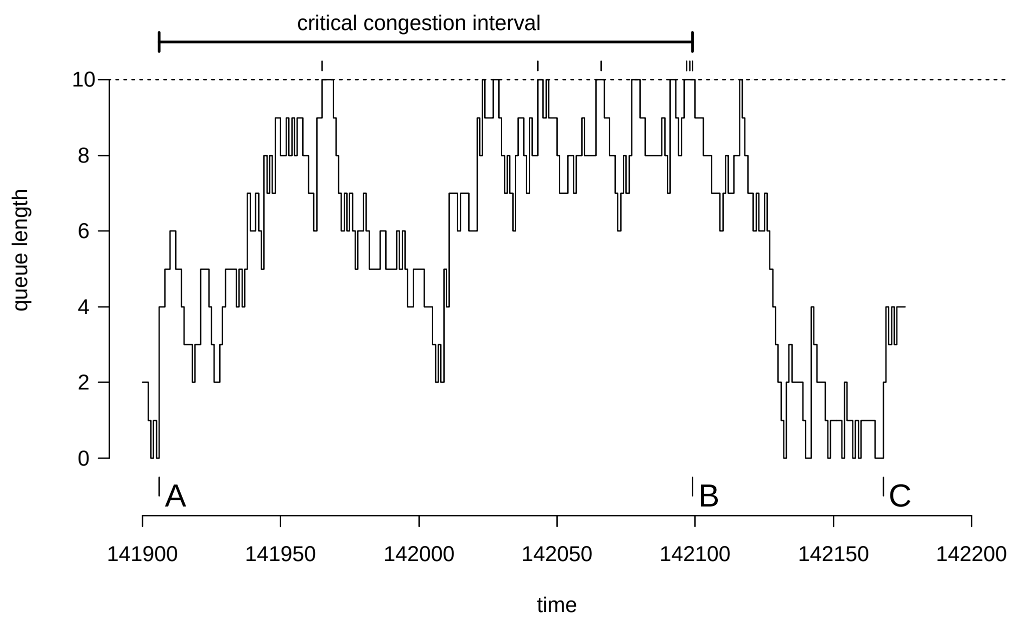

Figure 4: Sample path shadow prices for a queue. This figure shows part of the queue length sample path in Scenario 4. The ticks at the top of the diagram indicate time units when loss occurred. The sample path shadow price of a packet is one or zero according to whether or not the packet lies between the start of a busy period and a packet loss within the same busy period. Thus packets arriving during the critical congestion interval between times A(=141906) and B(=142099) have a sample path shadow price of one, while those arriving between times B and C(=142168) have a sample path shadow price of zero. It is not clear from the sample path up to the end of the section shown whether the packets arriving after time C will cause a packet loss or not, and hence their sample path shadow price is not (yet) determined.

Scenario 4

This scenario had three sub-populations of users.

(i) Intermittent-users. There were two independent copies of Intermittent-user(w_i), where $\kappa = 0.001$ and $w_i = i \times 0.001$, for $i = 1, 2, \ldots, 20$. Active periods had mean 10000, while sleep periods had mean 40000.

(ii) Intermittent-file-transfers. Such a user behaves as File-transfer(F, W_j) while a file transfer is in progress, and then sleeps for a random period with mean 40000. Sleep periods are independent and geometrically distributed. There were ten intermittent-file-transfers, with $F = 1000$, $w_{min} = 0.001$ and $W_j = 150 + 20j$, for $j = 1, 2, \ldots, 10$.

(iii)Unresponsive-users. There were ten such users, namely Unresponsive-user(g_k) for $k = 1, 2, \ldots, 10$, where $g_k = 0.01k$.

Figures 4 and 5 were constructed from the fourth scenario. The proportion of packets marked was approximately 0.18, while the proportion of packets lost was about 0.014. The utilization of the resource was about 0.84. For busy periods containing a loss, the median length of time between the start of the busy period and the last loss within that busy period was about 120 time units.

In Scenario 4, Unresponsive-users accounted for about 32% of the packets transmitted. At this level, they did not significantly disrupt the performance of the system, since there was ample responsive traffic to cope with the various fluctuations in load. The behaviour of Intermittent-file-transfers (accounting for about 14% of the packets transmitted) is well illustrated by Figure 5: those that are willing to pay more achieve faster and more frequent file transfers. Similarly we observed that the throughput achieved by Intermittent-users is highly responsive to their choice of their parameters w_i.

In Scenario 4 the ratio of packets marked to packets delivered, and the ratio of packets lost to packets marked, are both small. This we expect to hold much more generally. In Scenario 4 the critical congestion interval at the queue (median length about 120 time units) and the feedback delay (100 time units) are comparable in length: more generally we might expect connections over a wide area network to have a relatively longer feedback delay, while local connections might have a relatively shorter feedback delay.

Next we discuss the marking mechanism at a queueing resource. The mechanism used to construct Scenario 4 ensures that the correct proportion of packets are marked by the resource, but the marks are not always allocated in accord with the exact sample path shadow price of a packet. Thus a packet that arrives early in a busy period, before any loss has occurred, may well escape without a mark, while a packet that arrives towards the end of a busy period, whose sample path shadow price is zero, may receive a mark. If the

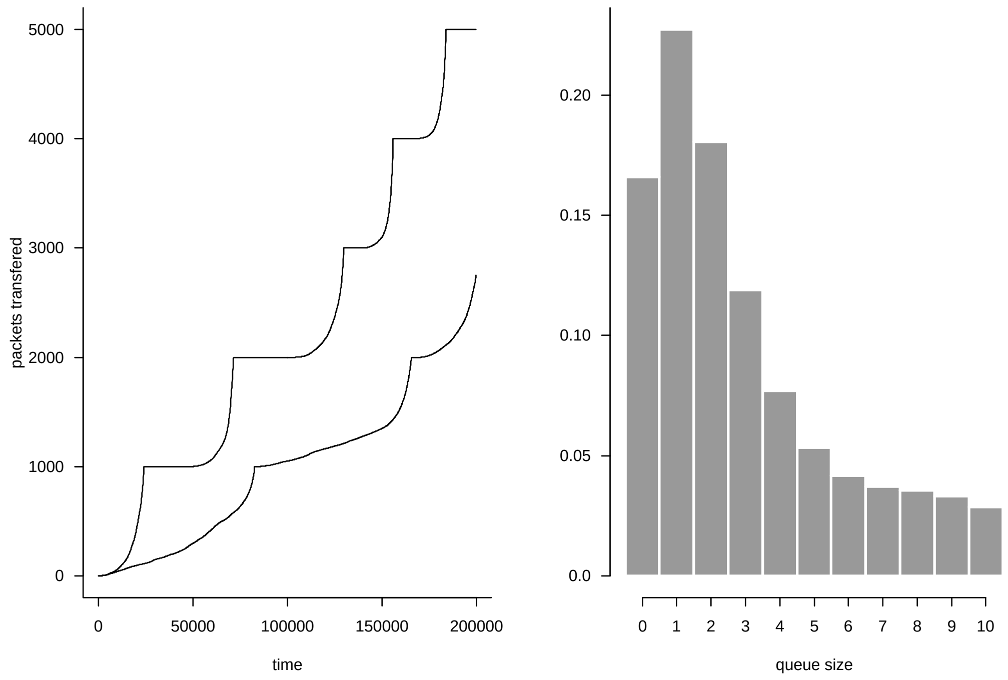

Figure 5: Scenario 4. The first panel shows the progress of two of the intermittent file transfers. The higher curve shows an algorithm prepared to pay 350 per file transfer: it transfers 5 files in the period shown. The lower curve shows an algorithm prepared to pay 170 per file transfer: it transfers almost 3 files in the period shown, despite rather short sleep periods (the horizontal sections of the curve). The second panel shows a histogram of the frequencies of queue sizes formed from a queue length sample path similar to that illustrated in Figure 4, but extending over two million time units.

critical congestion interval is short in comparison with the the feedback delay, then the inaccuracy in marking the exact sample path shadow price may not matter, since closed-loop control of the queue size will not be feasible. But if some feedback delays are relatively short, or if it is desired to achieve a very low packet loss ratio, then it will be appropriate for the resource to anticipate the possible overflow of its buffer, and to mark packets somewhat earlier within busy periods. There are several ways in which this might be achieved: we discuss two potential mechanisms, the first based on Floyd and Jacobson's proposals for Random Early Discard and Explicit Congestion Notification [6, 8], and the second a variation of the mechanism described earlier in this Section.

The resource may not have foreknowledge of its future, but it may nonetheless be able to estimate the probability that a given packet's sample path shadow price will be one. The resource might then mark packets randomly as they leave the queue, with the probability of marking a given packet calculated as the conditional probability, given the entire history at the resource up to the moment that packet departs, that a later packet will be lost before the end of the current busy period. As a possible approximation to this, suppose that the resource marks a packet with a probability that increases with the average queue size, where the average queue size is calculated from an exponentially weighted average of the recent past; for example, the probability of marking might increase linearly between a minimum threshold and a maximum threshold, with every packet marked above the maximum threshold [6, 8]. Floyd and Jacobson [8] provide *inter alia* a detailed discussion of the choice of the the time constant of the average queue calculation, and of the two thresholds. We simply note that their proposals are entirely feasible as a mechanism for the probabilistic marking of sample path shadow prices. The thresholds could be occasionally reset or dynamically controlled so that approximately the correct proportion of packets are marked, and interesting questions concern the extent to which such a mechanism could improve the performance of the overall system.

Next we describe how the marking mechanism used earlier in this Section may be amended to anticipate possible overflow. Suppose that the resource keeps track of a virtual queue, which has exactly the same arrival process of packets as the real queue, but has a service rate and a buffer capacity which are both scaled down by the same factor (for example 0.9). The marking mechanism then operates as follows: when the virtual queue loses a packet, the real queue starts to mark every packet leaving it, and continues to do so until the virtual queue becomes empty. To illustrate the behaviour of such a system, we reran Scenario 4, with a real queue having a service rate of 1.1 packets per unit time and a buffer capacity of 11, and with a virtual queue with a service rate of 1 packet per unit time and a buffer capacity of 10 (thus the virtual queue has a service rate and a buffer capacity which are both scaled down by the factor 10/11, and is identical with the 'real' queue discussed earlier). The proportion of packets marked, about 0.18, remained the same, but now the proportion of packets lost was reduced to about 0.003. Note that a mark is now interpreted as a shadow price for the virtual queue. The real queue may be viewed as having an additional margin of capacity "unseen" by the marking mechanism, and the choice of this margin may be influenced by considerations of, for example, robustness to sudden overload or failure.

We end this Section with a brief comment on the theoretical framework suitable for a network with infinite buffers and no packet loss. The impact of a single packet on other packets passing through a resource must now be assessed in terms of added delay rather than additional loss. Indeed the sample path shadow price of a given packet is simply the number of packets that leave the resource between the given packet's departure and the end of the current busy period, since each of these packets spends exactly one extra unit of time in the queue as a consequence of the existence of the given packet. The sample path shadow price of a packet is not known at the time the packet leaves the queue, but if, for example, the packet is marked with the length of time since the *start* of the busy period, then the expected charge at the resource is equal to the expected sample path shadow price, for a randomly chosen packet. We have not pursued this framework here; it seems much less straightforward to interpret in terms of existing Internet mechanisms.

5 Comparison with the current Internet

Currently resources in the Internet provide congestion indication signals by dropping packets, and the response of the end-nodes is (often) defined by the slow-start and congestion avoidance algorithms of Jacobson [12]. In this Section we compare these mechanisms with those of the model described in this paper.

Dropping packets is an extreme mechanism for providing congestion indication signals, and one should expect better performance in a network than can mark packets without dropping them. But a subtler point concerns the *rate* at which feedback signals are generated at a resource, and how this scales with the size of the resource and the statistical characteristics of traffic through the resource.

Consider, for example, the simple slotted model of Section 2, where the resource has capacity per slot to cope with N packets, and suppose the load upon the resource is Poisson with mean y. Then the expected number of dropped packets per unit of flow is $C(y)/y$, while the expected number of marks per unit flow is $p(y)$, the shadow price. To illustrate the divergence between these these two quantities, consider the case of a heavily loaded resource with $y = N$. Then the proportion of packets dropped is approximately $(2\pi N)^{-1/2}$, while the shadow price is approximately $1/2$. The ratio of these two quantities thus decreases with the size of the resource as measured by N in the slotted model or by the number of packets served per critical congestion interval in the queueing model. If dropped packets at resources of widely varying sizes are viewed as equivalent congestion indication signals, then the number of dropped packets per unit flow at larger resources will significantly underestimate the true congestion costs at these larger resources.

Next we consider the response of the end-nodes to congestion indication signals. The steady state behaviour of Jacobson's transmission control protocol has been analysed extensively. In equilibrium the rate allocated to a user is approximately

$$x \simeq \frac{c}{T\sqrt{p}}, \tag{5}$$

where T is the round-trip time of the connection, p is the packet loss probability over the connection, and c is a known constant [7, 21]. The origin of the square root dependence is interesting. A flow through the current Internet will receive congestion indication signals at a rate roughly proportional to the size of the flow. The response of Jacobson's congestion avoidance algorithm to a congestion indication signal is to halve the size of the flow. Thus there are *two* multiplicative effects: both the number of congestion indication signals received and the response to each signal scale with the size of the flow. The square root dependence on p was essentially observed in [12, footnote 6 on pages 4 and 5], where it is noted that a trivial modification of the algorithm can make the relationship linear.

Equations (3) and (10) describe algorithms in which the response to a congestion indication signal is a simple reduction in flow, by a constant amount: call this an additive, as opposed to a multiplicative, reduction. The marking mechanisms employed at resources will naturally produce congestion indication signals at a rate proportional to the size of a flow, and this, together with an additive reduction by the user, is enough to ensure that the overall decrease in rate is multiplicative, and that the equilibrium rate allocated to a user has the inverse linear dependence on $p = \sum_{j\in r} \mu_j$ given by equation (12). Alternatively, a multiplicative reduction in flow could be combined with algorithmic features that mitigate the proportional dependence of the number of congestion indication signals received on the flow. For example, if an end-system reacts to at most one congestion indication signal per round-trip time (as suggested in [6]), or if it requires at least a certain proportion of packets within a window to be marked (as in the DECbit scheme [24]), then a near linear relationship may be achieved.

The algorithms of File-transfer and of Elastic-user share several features of the Slow-Start and Congestion Avoidance algorithms respectively of Jacobson [12]: File-transfer and Slow-Start both allow an exponential growth from a low initial rate; Elastic-user and Congestion Avoidance both involve additive increase, but use respectively an additive or multiplicative decrease. It is an interesting question whether a new connection in the current Internet should grow from a low initial rate, or whether it could immediately begin transmitting at a higher rate that might depend upon earlier experiences of the network. Of course the point of the marking mechanisms described in this paper is that the end-nodes can decide for themselves: they have the information and the incentives from the network to make reasonable choices.

It is possible that the square root dependence (5) of a user's rate on p may be a reasonable choice for a user: for example, if the user were to have the implicit utility function

$$U(x) = \text{constant} - \frac{c^2}{T^2 x}$$

then the solution to the user's optimization problem (14) would produce precisely the rate (5), with an associated cost

$$\frac{c\sqrt{p}}{T} \tag{6}$$

per unit time. The problem is that the user has no means to influence these outcomes. Crowcroft and Oechslin [5] have proposed that users be allowed to set a parameter m, which would *inter alia* multiply by m the rate of additive increase and make $1 - 1/2m$ the multiplicative decrease factor in Jacobson's algorithm. The resulting algorithm, MulTCP, would behave in many respects as m single TCP connections; in particular, it would have a rate and charge given by m times expressions (5) and (6) respectively. Thus MulTCP could provide a natural incremental path by which TCPs used by applications might evolve.

An important feature of Jacobson's algorithm [12] is that it is *self-clocking*: the sender uses an acknowledgement from the receiver to prompt a step forward, and this explains the inverse dependence on the round-trip time T in relation (5). In more detail, TCP maintains a window of transmitted but not yet acknowledged packets; the rate x and the window size *cwnd* satisfy the approximate relation $cwnd = xT$. During Slow-Start each acknowledgement increments *cwnd* by one; in the absence of congestion indication this produces a doubling in window size per round-trip time T. Similarly File-transfer produces an exponentially growing rate; for the doubling time to be proportional to T, a sensible scaling requirement for stability, it is necessary for κ to be inversely proportional to T.

A self-clocking form of Elastic-user(w) can be constructed by incrementing *cwnd* by

$$\bar{\kappa}\left(\frac{\bar{w}}{cwnd} - f\right)$$

per acknowledgement, where $f = 1$ or 0 according as the packet acknowledged was marked or not. Since the time between update steps is about $T/cwnd$, the expected change in the rate x per unit time is approximately

$$\frac{\bar{\kappa}\left(\frac{\bar{w}}{cwnd} - p\right)/T}{T/cwnd}$$
$$= \frac{\bar{\kappa}}{T}\left(\frac{\bar{w}}{T} - px\right),$$

corresponding to a linear increase and multiplicative decrease of the form (10), with $\kappa = \bar{\kappa}/T$ and $w = \bar{w}/T$. Thus $\bar{\kappa}$ and $\bar{w}$ are respectively the gain and charge *per round-trip time* rather than per unit time. Self-clocking may well be chosen as a sensible strategy for routes with uncertain round-trip times: it certainly simplifies the choice of the gain parameter κ, which might otherwise be expected to depend upon an estimate or bound on the round-trip time of a connection.

Just as different end-nodes may choose w or $\bar{w}$ to suit their needs, they may have different requirements for the gain parameter: higher values of κ or $\bar{\kappa}$ will result in a higher variance for x in equilibrium, but will allow a more rapid convergence to equilibrium [17]. Similarly, different applications may choose more or less sophisticated methods to deliver feedback information. For example, several acknowlegements might be grouped together into a single packet to be returned from the receiver to the sender, or acknowledgements might wait to be piggybacked onto other packets due to be sent between end-nodes. By these means the number of reverse packets, each themselves vulnerable to marking, may be reduced, at the cost of slower feedback.

In our simulations we have not distinguished between packets which are lost and those which are merely marked: we have treated them all as marked packets. The distinction is important for the retransmission behaviour of a protocol, but is less important for congestion avoidance behaviour. If a nominally marked packet is in fact dropped then a missing sequence number could provide the congestion information that would have been carried by the mark, as well as the packet identification necessary for any retransmission. The absence of a charge for that packet corresponds very approximately to the actual loss of the packet to that connection, and seems unlikely to influence incentives since, in general, congestion will cause many more packets to be marked than to be dropped. At a subtler level, end-nodes may attempt to learn network characteristics from the ratio of dropped to marked packets. For example, if a connection includes a noisy radio link at which packets are dropped for reasons other than congestion, then end-systems could deduce, from a preponderance of dropped to marked packets, that it is in their interest (and is not damaging to others) to make generous use of forward error correction.

In a network it is logically possible that a packet may be marked multiple times, and the shadow prices of the Appendix are certainly additive; but if the proportion of packets marked is small it will be enough to have a single bit record whether or not a packet has been marked at *any* resource. Note that if a packet on a certain route were to be marked twice during its progress through the network, then the existence of that packet would have increased by *two* the number of packets lost by the network: one might expect such events to be unlikely except under implausibly high levels of loading and of expected cost for a route.

In earlier Sections we have assumed that all packets are of the same size. In an environment with packets of different sizes there are several approaches compatible with our framework. If the load on resources is proportional to the length of a packet, then a marked packet could be charged a fixed amount *per byte* of the packet. Alternatively, if there is a maximum packet size, then a packet passing through a resource at a time of congestion could be marked with a probability proportional to its length. Either approach makes the expected charge proportional to the length of the packet; the latter approach has the advantage that it could be used in a network where at some resources the load is proportional to the length of a packet, while at other resources it is not.

6 Analytical models

In this Section we briefly outline some analytical models which complement those of [17], shedding light on other aspects of the interaction between users and the network.

6.1 User queues

The algorithm File-transfer(F, W) attempts to pay a price W/F per packet on average. If the average shadow price is currently higher than this, then the algorithm waits until the shadow price decreases. How long might the algorithm have to wait? In this sub-section we use a simple queueing model to cast some light on this issue.

Suppose that arrivals of users at a single resource of unit capacity form a Poisson process, and that each user has a volume of packets to transfer through the resource which is exponentially distributed with unit mean and independent from user to user. Suppose that user r is prepared to pay α_r per unit volume transferred, and has a rate control algorithm able to effect this choice. Assume that the values $\alpha_r, r = 1, 2, \ldots$, associated with successively arriving users are independent and identically distributed, and independent of users' volumes: let

$$\rho(z) = \int_z^\infty f(\alpha) d\alpha$$

be the arrival rate of the Poisson stream of users with values of α greater than z.

Under the above assumptions the stream of users with values of α greater than z will effectively be served by an M/M/1 queue with traffic intensity $\rho(z)$. The mean sojourn at the resource experienced by this stream of users will thus be $(1 - \rho(z))^{-1}$. This relationship holds for each value of z, and hence the mean sojourn at the resource experienced by a user with a given value of α, $D(\alpha)$, satisfies

$$f(\alpha) D(\alpha) = -\frac{d}{d\alpha} \left(\frac{\rho(\alpha)}{1 - \rho(\alpha)} \right),$$

and thus

$$D(\alpha) = \frac{1}{(1 - \rho(\alpha))^2} \quad \text{if } \rho(\alpha) < 1.$$

If the parameter α associated with a user is such that $\rho(\alpha) > 1$, then, in our stationary model, that user does not receive service at the resource.

Other queueing models are, of course, possible. For example suppose that each user has a volume of packets to transfer through the resource which is arbitrarily distributed and independent from user to user, and a value of α

that is inversely related to the volume: thus each user has the same amount to spend in total. Then the system operates as a priority queue in which the server concentrates attention on the user with the smallest volume to transfer: see [13] for general results on priority queues, and [20] for a discussion in the context of congestion control algorithms.

Treating the network as a single resource of unit capacity has allowed several insights from queueing theory, but clearly oversimplifies the dynamical behaviour of the underlying rate control algorithms. Tan [28] has developed the model of [17] to include the arrival and departure of various types of user, and has applied the resulting diffusion model to study the combined stability of rates and user numbers in a network comprising many resources.

6.2 Nash equilibria

The optimization model of the Appendix, and in particular the user optimization problem (14), assumes that a user does not anticipate the effect of its own actions on prices. This may be a reasonable assumption for a user who occupies at most a small fraction of any single resource of the network, but as a user occupies more and more of a resource, eventually the user will have an impact on prices. Thus if an Elastic-user(w) is providing only a small fraction of the total load on each resource along its route, and most other congestion control algorithms are similar to Elastic-user, then doubling w will nearly, but not quite, double the user's received rate. But the larger the fraction of a resource occupied by the user, the greater the disincentive for that user to further increase its choice of w. The magnitude of the effect depends upon the strategies used by other users: even a moderately sized user may not affect prices in a network where most other congestion control algorithms, for example File-transfer(F, W), attempt to pay a given price per packet on average.

In this sub-section we use some simple game-theoretic models to explore how users might behave if they can anticipate their own effect on prices.

Suppose that the utility of user r takes the form

$$U_r(x_r) = \alpha_r x_r, \qquad (7)$$

and suppose the unit capacity of the resource is shared out over users in proportion to the w's the users choose. If user r assumes that it has *no* effect on the price of the resource, then it will flip between choosing w_r very large or very small, according as the current charge per unit flow appears greater or less than α_r. But if the user realises that its own choice of w_r affects the flow per unit charge (a realisation that if not explicit may be an implicit consequence of its control algorithm), then the user may be motivated to choose w_r to maximize

$$U_r\left(\frac{w_r}{W_r + w_r}\right) - w_r,$$

where W_r is the sum of the w's chosen by the *other* users. Then a simple analysis gives that the collection of individually maximizing choices for the various users (the Nash equilibrium) is given by

$$\widehat{w_r} = \widehat{W}\left[1 - \frac{\widehat{W}}{\alpha_r}\right]^+$$

where $\widehat{W}$ is the unique value satisfying

$$1 = \sum_r \left[1 - \frac{\widehat{W}}{\alpha_r}\right]^+ .$$

Thus $\widehat{W}$ is a threshold: users with values of α_r smaller than $\widehat{W}$ will choose $w_r = 0$. If $r = 1, 2, \ldots, n$ label the users for which $\alpha_r > \widehat{W}$ then the total amount paid per unit time to the resource is

$$\widehat{W} = \left(1 - \frac{1}{n}\right)\left(\frac{1}{n}\sum_{r=1}^{n} \alpha_r^{-1}\right)^{-1},$$

slightly less than the harmonic mean of $\alpha_1, \alpha_2, \ldots, \alpha_n$.

We can illustrate some consequences most easily in the symmetric case when $\alpha_r = \alpha$, for all r. Then the total amount charged per unit time when $n > 0$ users are present is

$$\widehat{W} = \alpha\left(1 - \frac{1}{n}\right),$$

a quantity that increases to an asymptote at α, the aggregate utility of the link over all users present. Thus when users can anticipate their own effect on prices, they pay less. This is an example of a familiar result from auction theory: users shade their bids if they have market power.

Next suppose that arrivals of users form a Poisson process, with successive file sizes that are independent identically distributed random variables. Then the stationary number of users present, n, has a geometric distribution with parameter ρ, the traffic intensity. if each user's control algorithm is effectively able to track n then the expected amount charged per unit time will be

$$\sum_{n=1}^{\infty}(1-\rho)\rho^n\alpha\left(1-\frac{1}{n}\right)=\alpha[\rho+(1-\rho)\log(1-\rho)].$$

This expression may be compared with $\alpha\rho$, the expected aggregate utility per unit time.

Nash equilibria depend sensitively upon the space of strategies allowed for players: in our context a strategy may correspond to the choice of a parameter in an algorithm, or more generally the choice of the algorithm itself. Thus the above analysis essentially models users able to choose the parameter w of the algorithm Elastic-user(w), while other algorithms may have different Nash equilibria.

Given the linear utility function (7) a more appropriate algorithm for user r may be File-transfer($F, \pi F$). Let us suppose, then, that user r has an algorithm that essentially allows user r to choose the amount it pays, π_r, per packet, for each $r \in R$. The resource will then allocate its entire capacity to the user or users choosing the highest price. This is essentially a simple auction [31], and the Nash equilibrium provides the entire capacity to the user prepared to pay the highest price, that is the user s with the highest value of α_s: this user chooses π_s to be slightly higher than the second highest value in the set $(\alpha_r, r \in R)$.

Another interesting class of algorithms has users essentially choose a rate. Under a naive formulation any vector $(x_r, r \in R)$ with non-negative components summing to the capacity of the link will be a Nash equilibrium: a more appropriate model must explore the behaviour of the resource when it is slightly overloaded. Let us suppose, then, that user r chooses rate x_r to optimize

$$U_r(x_r)-x_r p\left(\sum_s x_s\right)$$

where p is differentiable, increasing and convex and U_r is the linear utility function (7). Then the Nash equilibrium now has

$$x_r=\left[\frac{\alpha_r-p(y)}{p'(y)}\right]^+$$

where y is the unique value satisfying

$$y=\sum_r\left[\frac{\alpha_r-p(y)}{p'(y)}\right]^+.$$

Thus the Nash equilibrium has the form $x_r = [c\alpha_r - d]^+$ where c, d depend upon the parameters $(\alpha_r, r \in R)$ of the set of users and the characteristics p of the resource.

Our last model explores the interaction between users of different types. We can illustrate the main point by assuming just three users sharing a resource of unit capacity: the first user has a utility function $U_1(x) = w \log(K_1 x)$, where K_1 is large, and uses an algorithm, like Elastic-user, that allows the user to choose the amount it pays per unit time, w_1; the second user has a utility function $U_2(x) = \alpha x$ and uses an algorithm, like File-transfer, that allows the user to choose the amount it pays per packet, α_2; the third user has a utility function $U_3(x) = K_3 I\{x > q\}$, where K_3 is large, and uses an algorithm, like Unresponsive-user, that allows the user to choose its rate, q_3. Then a simple analysis of the Nash equilibrium shows that, provided $\alpha(1-q) > w$ and K is large enough, the three users choose, respectively, $w_1 = w$, $\alpha_2 = ((w\alpha/(1-q))^{0.5}$ and $q_3 = q$. Observe that only the second user, the price setter, anticipates the effects of its own actions on prices.

Our general conclusion from this brief exploration of Nash equilibria is that if a user accounts for a major component of the load on the network, then rational behaviour by that user's rate control algorithm will be to back off slightly, relative to a more naive algorithm that does not anticipate the effect of its own actions on prices.

7 Concluding remarks

We have seen that, by appropriately marking packets at overloaded resources, end-nodes are provided with the necessary information to make efficient use of the network. This may be enough in a network with cooperative end-nodes: otherwise a fixed small charge for each mark ensures that end-nodes also the correct incentive to use the network efficiently. Thus the marks we have described provide a rational basis for a usage-related charging scheme. If usage-related costs are not a large proportion of network costs, then one might expect that these usage-related charges would be small in comparison with non-usage-related charges such as connection and subscription charges. Nonetheless even small usage-related charges should influence developers to incorporate end-to-end congestion control in their applications, and thus lead to a substantial improvement in the efficiency of network operation.

The approach of this paper has not explicitly considered the market structure within which the network operates. (See [10] for an analysis of Odlyzko's [22] "Paris Metro Pricing" proposal within an economic model of competition.) The model of [17] allows routing choices by users, and this provides some insight into how a geographically structured competitive environment might operate. An oft-expressed concern about congestion related pricing is that a monopolist might deliberately allow congestion in order to increase revenue. This is really a concern about monopoly rather than congestion pricing: we simply note that even an unregulated monopolist would have an incentive to balance charges between, for example, connection, subscription and usage components so as to maximize the efficiency of the network. The key point here is familiar from economic theory [30]: if a monopolist has sufficient freedom over pricing then it can maximize net benefit *and* appropriate this benefit for itself.

Shenker *et al.* [27] provide a valuable discussion of architectural issues of pricing IP networks, and in particular the topics of multicast and receiver charging. We note that the marks carrying shadow prices can be distributed to the receivers of a multicast quite naturally. Consider a point in the network where a packet is replicated to produce, say, k daughter packets. If the packet to be replicated carries a mark, then assign this mark to just one of the daughter packets, randomly selected from amongst the k possibilities. (The random distribution may be uniform or weighted in accordance with any other information available concerning, for example, subsequent replications.) The shadow price of each scarce resource is thus shared over the users benefiting, and end-nodes have incentives to join multicasts rather than to use inefficient unicasts. An interesting open question concerns how the scheme would operate for layered multicasts: here we simply note that the mark received by a user would be informative only about the layer currently subscribed to by the user. A related architectural issue concerns how users might be given some measure of control over the routing of packets: for example a user may be happy to accept a higher marking rate on a route with lower transmission delay. Routing algorithms based on shadow prices are described in [9, 15, 23], but for networks where the user has little influence on routing decisions.

The work described in this paper provides a rich area for further research. Besides further refinements to the analysis and the numerical exploration of more complex networks there is much scope to make improvement to the flow control strategies themselves. Currently work is in progress on open competitions to elicit strategies for given user objectives. A particularly simple and attractive implementation of this work is possible as classes in the Java object-orientated programming language. Further work currently under way [18] envisages the creation of a network server accepting load from clients and feeding back their marks accordingly. The intention here is to allow multiple players to interact, under conditions of uncertainty, on common and scarce resources.

Acknowledgements: Richard Gibbens was supported by a Royal Society University Research Fellowship; computing equipment was provided by the EPSRC under grant GR/M09551, and further support was provided by the Commission of the European Communities ACTS project AC039, entitled Charging and Accounting Schemes in Multiservice ATM Networks (CA$hMAN). The authors are grateful to Fred Baker, Costas Courcoubetis, Jon Crowcroft, Sally Floyd, Peter Key, Derek McAuley, Robin Mason, K.K. Ramakrishnan, David Songhurst, George D. Stamoulis, David Tan, Stephen Turner, Ian Wakeman, Richard Weber and Damon Wischik for valuable discussions and suggestions, and to two anonymous referees for their perceptive comments.

References

[1] R. Axelrod. *The Evolution of Co-operation.* Penguin, London, 1984.

[2] D.-M. Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17:1–14, 1989.

[3] D. D. Clark. Adding service discrimination to the internet. *Telecommunications Policy*, 20, 1996. http://ana-www.lcs.mit.edu/anaweb/abstracts/TPRC2-0.html.

[4] C. Courcoubetis, G. D. Stamoulis, C. Manolakis, and F. P. Kelly. An intelligent agent for optimizing QoS-for-money in priced ABR connections. Preprint, 1998.

[5] J. Crowcroft and P. Oechslin. Differentiated end-to-end Internet services using a weighted proportionally fair sharing TCP. *ACM Computer Communications Review*, 28:53–67, 1998.

[6] S. Floyd. TCP and Explicit Congestion Notification. *ACM Computer Communications Review*, 24:10–23, 1994. http://www-nrg.ee.lbl.gov/floyd/ecn.html.

[7] S. Floyd and K. Fall. Promoting the use of end-to-end congestion control in the Internet. http://www-nrg.ee.lbl.gov/floyd/end2end-paper.html, 1998.

[8] S. Floyd and V. Jacobson. Random Early Detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1:397–413, 1993. ftp://ftp.ee.lbl.gov/papers/early.pdf.

[9] R.G. Gallager. A mininum delay routing algorithm using distributed computation. *IEEE Transactions on Communications*, 25:73–85, 1977.

[10] R. J. Gibbens, R. Mason, and R. Steinberg. Multiproduct competition between congestible networks. University of Southampton Discussion Paper in Economics and Econometrics, No. 9816, http://www.soton.ac.uk/~ram2, 1998.

[11] A. Gupta, A. O. Stahl, and A. B. Whinston. Priority pricing of integrated services networks. In L. W. McKnight and J. P. Bailey, editors, *Internet Economics*. MIT Press, Cambridge, Massachussetts, 1997.

[12] V. Jacobson. Congestion avoidance and control. In *Proc. ACM SIGCOMM '88*, pages 314–329, 1988. A revised version, joint with M.J. Karels, is available via ftp://ftp.ee.lbl.gov/papers/congavoid.ps.Z.

[13] N. K. Jaiswal. *Priority Queues*. Academic Press, New York, 1968.

[14] F. P. Kelly. *Reversibility and Stochastic Networks*. Wiley, Chichester, 1979.

[15] F. P. Kelly. Routing in circuit-switched networks: optimization, shadow prices, and decentralization. *Advances in Applied Probability*, 20:112–144, 1988.

[16] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997. http://www.statslab.cam.ac.uk/~frank/elastic.html.

[17] 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. http://www.statslab.cam.ac.uk/~frank/rate.html.

[18] P. Key and D. McAuley. Differential QoS and pricing in networks: where flow control meets game theory. *IEE Proc Software*, 146, 1999.

[19] J. K. MacKie-Mason and H. R. Varian. Pricing the Internet. In B. Kahin and J. Keller, editors, *Public Access to the Internet*. Prentice-Hall, Englewood Cliffs, New Jersey, 1994.

[20] L. Massoulié and J. Roberts. Fairness and quality of service for elastic traffic. Preprint, 1998.

[21] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The macroscopic behaviour of the TCP congestion avoidance algorithm. *Computer Communication Review*, 27, 1997.

[22] A. Odlyzko. A modest proposal for preventing Internet congestion. http://www.research.att.com/~amo/doc/modest.proposal.ps, 1997.

[23] T.J. Ott and K.R. Krishnan. State dependent routing of telephone traffic and the use of separable routing schemes. In M. Akiyama, editor, *Proc. 11th International Teletraffic Congress*. North-Holland, Amsterdam, 1985.

[24] K.K. Ramakrishnan and R. Jain. A binary feedback scheme for congestion avoidance in computer networks. *ACM Transactions on Computer Systems*, 8:158–181, 1990.

[25] M. Ridley. *The Origins of Virtue*. Penguin, London, 1997.

[26] S. Shenker. Fundamental design issues for the future Internet. *IEEE Journal Selected Areas Communication*, 13:1176–1188, 1995.

[27] S. Shenker, D. Clark, D. Estrin, and S. Herzog. Pricing in computer networks: reshaping the research agenda. *Telecommunications Policy*, 20:183–201, 1996.

[28] D.K.H. Tan. Rate control and user behaviour in communication networks. http://www.statslab.cam.ac.uk/~dkht2/conf.ps, 1998.

[29] International Telecommunications Union. Recommendation I371: Traffic control and congestion control in b-isdn. Geneva, 1996.

[30] H. R. Varian. *Microeconomic Analysis*. Norton, New York, third edition, 1992.

[31] W. Vickrey. Counterspeculation, auctions and competitive sealed tenders. *The Journal of Finance*, 16:8–37, 1961.

A Rate control of elastic traffic

In this Appendix we review some of the results of [17].

Consider a network with a set J of *resources*. Let a *route* r be a non-empty subset of J, and write R for the set of possible routes. Associate a route r with a user, and suppose that if a rate x_r is allocated to user r

then this has utility $U_r(x_r)$ to the user. Assume that the utility $U_r(x_r)$ is an increasing, strictly concave and continuously differentiable function of x_r over the range $x_r \geq 0$ (following Shenker [26], we call traffic that leads to such a utility function *elastic* traffic). Suppose that when a resource is heavily loaded the network incurs some cost, perhaps expressed in terms of delay or loss: specifically, suppose that $C_j(y)$ is the rate at which cost is incurred at resource j when the load through it is y. Assume further that utilities and costs are additive, so that the aggregate utility of rates $x = (x_r, r \in R)$ is

$$\mathcal{U}(x) = \sum_{r \in R} U_r(x_r) - \sum_{j \in J} C_j \left(\sum_{s : j \in s} x_s \right). \tag{8}$$

Suppose that $C_j(y)$ is differentiable, with

$$\frac{d}{dy} C_j(y) = p_j(y), \tag{9}$$

where, for $j \in J$, the function $p_j(y)$, $y \geq 0$, is a non-negative, continuous, increasing function of y, not identically zero. Thus $C_j(y)$ is a convex function, and so $\mathcal{U}(x)$ is strictly concave. Assume the functions U_r and C_j are such that $\mathcal{U}(x)$ achieves a maximum in the region $x \geq 0$; by the strict concavity of $\mathcal{U}(x)$ such a maximum is unique.

Next consider the system of differential equations

$$\frac{d}{dt} x_r(t) = \kappa \left(w_r(t) - x_r(t) \sum_{j \in r} \mu_j(t) \right) \tag{10}$$

for $r \in R$, where

$$\mu_j(t) = p_j \left(\sum_{r : j \in r} x_r(t) \right) \tag{11}$$

for $j \in J$. We interpret the relations (10)–(11) as follows. Suppose that resource j generates a continuous stream of feedback signals at rate $y p_j(y)$ when the total flow through resource j is y; that resource j sends a proportion x_r/y of these feedback signals to a user r with a flow of rate x_r through resource j; and that user r views each feedback signal as a congestion indicator requiring some reduction in the flow x_r. Then equation (10) corresponds to a response by user r that comprises two components: a steady increase at rate proportional to $w_r(t)$, and a steady decrease at rate proportional to the stream of feedback signals received.

It is shown in [17] that if $w_r(t) = w_r$ for $r \in R$ then the system of differential equations (10)–(11) has a stable point, to which all trajectories converge. The variable $\mu_j(t)$ is the *shadow price* per unit of flow through resource j at time t, and at the stable point

$$x_r = \frac{w_r}{\sum_{j \in r} \mu_j}. \tag{12}$$

The rates x determined by equation (12) have an interpretation as a set of rates that are *proportionally fair per unit charge*, as discussed in [16] and [17].

Next suppose that user r is able to monitor its rate $x_r(t)$ continuously, and to vary smoothly the parameter $w_r(t)$ so as to satisfy

$$w_r(t) = x_r(t) U_r'(x_r(t)) : \tag{13}$$

this would correspond to a user who observes a charge per unit flow of $\lambda_r = w_r(t)/x_r(t)$, and chooses $w_r = w_r(t)$ to solve the optimization problem

$$\begin{array}{ll} \text{maximize} & U_r\left(\dfrac{w_r}{\lambda_r}\right) - w_r \\ \text{over} & w_r \geq 0. \end{array} \tag{14}$$

It is shown in [17] that the expression (8) is a Lyapunov function for the system of differential equations (10)–(11), (13), and hence that the vector x maximizing $\mathcal{U}(x)$ is a stable point of the system, to which all trajectories converge. The impact of random effects and time lags on stability is also considered in [17], and arrivals and departures of users are treated in [28].