Seminar on Theoretical Computer Science: Discussion on Congestion Control Using a Simple Marking Scheme

Emilia Oikarinen
emilia.oikarinen@hut.fi
Helsinki University of Technology

April 18, 2002

1 Introduction

The transmission control protocol, TCP [8], is used in the current Internet to control the rate at which a source sends packets. TCP is implemented as software on the computers that act as the source and the destination of the data being sent. The TCP algorithm works as follows [2]. When a resource in the network becomes overloaded, it starts dropping packets. A loss of a packet is considered an indication of congestion. As the destination informs this to the source, the source decreases its transmission rate. TCP increases the transmit rate gradually until it receives again an indication of congestion.

This alternation of increase and decrease of the transmit rate is used to discover and utilize the available bandwidth and it has worked well in the past where applications have been quite homogeneous. However, things have changed. There is a will to modify the TCP algorithm to a more aggressive direction where it strives for a larger share of the available bandwidth or even avoids any form of congestion control. Also the needs of more heteregeneous applications vary widely. TCP might not be the best option for them.

It has been noted in [6] that the lack of end-to-end congestion control could lead to collapse due to congestion in the Internet, as the congested links may occupy themselves in sending packets that will only be dropped later in the network. This has motivated work on how to implement various measurements and scheduling mechanisms within the network itself [2, 6], so that the network could restrict the bandwidth of flows and distinguish between services offered to different users.

In this paper we present a different elegant approach Gibbens and Kelly have explored in [7]. Their main objective is to show that when the end-nodes get to know the resource implications of their own actions, the end-nodes themselves are the best to decide what their demands upon the resources of the network should be. It should be more efficient to let intelligent end-nodes to determine their needs based on information on the congestion than to require the users to classify their flows into predefined categories and then forward this information to the network. The drawback on the latter approach is that it requires the definition of some service categories even before there are any applications to use these categories. It is hard to predict the categories needed in the future.

It has been shown [9], that if the utilities of the users are concave functions then the total utility of all the users is maximized if the network allocates the resources according to a weighted *proportional fairness* criterion i.e. the network shares the resource proportionally to the price the users are willing to pay for it. It has been also shown [10] that a weighted proportional fairness in allocation of resources can be achieved by simple rate control algorithms, using increase and decrease rules similar to the ones implemented in TCP.

An approach to allocating resources described by MacKie-Mason and Varian is so called *smart market* [14]. In smart market price for each packet is set depending on the level of demand. Gibbens and Kelly [7] show that the benefits of the smart market in efficient allocation of scarce resources can be achieved by simple network mechanisms that set one bit to mark some packets. They also show that the smart market can be achieved when users use different suitable rate control algorithms.

Gibbens and Kelly [7] present that for a network with cooperative end-nodes, the information end-nodes gain from the marks is enough to make efficient use of the network. If the end-nodes are uncooperative, a fixed small charge per each received mark ensures that the end-nodes have right incentives. Wischik has been studying further the concept of fair marking [17].

In this paper we present mainly the results and experiments in [7]. Most of the explicit references to [7] have been omitted and some parts of [7] are covered only very briefly.

2 Preliminaries and Notations

We present here some notations used in this paper and basic concepts of the theory of probability.

- The positive part of a function f is f^+. The positive and the negative parts of a function are defined as follows [1]

$$\begin{aligned} f &= f^+ - f^-, \\ f^+ &= \begin{cases} f, & f > 0 \\ 0, & \text{otherwise}, \end{cases} \\ f^- &= \begin{cases} -f, & f < 0 \\ 0, & \text{otherwise}. \end{cases} \end{aligned}$$

- Step function $I\{cond.\}$ is defined as

$$I\{cond.\} = \begin{cases} 1, & \text{if } cond. \text{ holds}, \\ 0, & \text{otherwise}. \end{cases}$$

So the function $XI\{Y > N\}$ is defined as follows

$$XI\{Y > N\} = \begin{cases} X, & \text{if } Y > N \\ 0, & \text{otherwise}. \end{cases}$$

- Poisson distribution $X \sim \text{Poisson}(\theta)$ is a discrete distribution with the expected value $\mathbb{E}(X) = \theta$ and the probability $\mathbb{P}\{X = k\} = \frac{\theta^k}{k!}e^{-\theta}$. [13]
- Binomial distribution $X \sim \text{Bin}(n, p)$ is a discrete distribution with the expected value $\mathbb{E}(X) = np$. [13]

3 Sample Path Shadow Prices in Slotted Model

To motivate the experiments described in Section 5, we examine the following simple resource model. First we make the following assumptions:

- Time is slotted i.e. it is considered to be discrete.
- The resource has capacity to cope with N equally sized packets per slot. Any excess of the packets is lost.
- Load upon the resource per slot is Y. Y is generated by adding together independent Poisson random variables $X_1, \ldots, X_m$ with means $x_1, \ldots, x_m$ respectively. Y has Poisson distribution with mean $y = \sum_{r=1}^{m} x_r$.

The number of packets lost is the cost to the system. We can give the expected cost per slot $C(y)$ by

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

The shadow price of a resource, $p(y)$, is defined as the marginal increment in expected cost at the resource for a marginal increment in load. Thus, the shadow price $p(y) = C'(y)$, and it satisfies

$$\begin{aligned} p(y) &= \frac{\partial}{\partial y}\left(\sum_{n \geq N} (n - N)e^{-y}\frac{y^n}{n!}\right) \\ &= \sum_{n \geq N} (n - N)\frac{\partial}{\partial y}\left(e^{-y}\frac{y^n}{n!}\right) \\ &= \sum_{n \geq N} (n - N)\left(-e^{-y}\frac{y^n}{n!} + e^{-y}\frac{y^{n-1}}{n!}n\right) \\ &= \sum_{n \geq N} (n - N)e^{-y}\left(-\frac{y^n}{n!} + \frac{y^{n-1}}{(n-1)!}\right) \\ &= \sum_{n \geq N} e^{-y}\frac{y^n}{n!} \\ &= \mathbb{P}\{Y \geq N\}. \end{aligned}$$

Let's suppose that when the number of packets arriving in a slot exceeds N, a mark is placed on each of these packets. No distinction is made between packets that are only marked and packets that are marked and actually lost. They all are treated as marked packets. The expected number of marks per unit time on packets from the r^{th} flow is

$$\begin{aligned}
\mathbb{E}(X_r I\{Y > N\}) &= \sum_{n>N} \mathbb{E}(X_r|Y=n)\mathbb{P}\{Y=N\} \\
&= \sum_{n>N} \frac{x_r}{y} n e^{-y} \frac{y^n}{n!} \\
&= x_r \sum_{n>N} e^{-y} \frac{y^{n-1}}{(n-1)!} \\
&= x_r \sum_{n\geq N} e^{-y} \frac{y^n}{n!} \\
&= x_r p(y),
\end{aligned}$$

because conditional on event $Y = n$, X_r has a binomial distributution,

$$(X_r|Y=n) \sim \mathrm{Bin}\left(n, x_r/y\right).$$

Thus, when using Poisson statistics, marking every packet arriving in an overloaded slot produces an expected charge per unit flow $p(y)$, which is exactly the shadow price at the resource. Furthermore, the expected cost per unit time to flow r is exactly the fair charge to the flow r under the model described in [10].

In general we cannot assume that the load is generated by independent Poisson processes and precise identities cannot be expected. However, we can show that the relationship between the expected increase in system cost caused by an increment in load, and the expected charge to that load increment holds for small increments.

Now let's suppose that the load Y on the resource is a positive random variable. We want to estimate the impact of an additional load X. X is non-negative random variable and it is not necessarily independent of Y. The increase in the number of packets lost is

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

Therefore it holds for the expected value of the increase in the number of packets lost that

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

Furthermore if we assume that N, Y and X are integral and the additional load is a small increment that satisfies $\mathbb{P}\{X=0 \text{ or } X=1\}=1$, then the event $\{X+Y>N>Y\}$ is impossible and it holds that

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

Thus, for small increments we have identity between the expected increase in system cost caused by the additional load and the expected charge to the additional load. It's worth noticing that inequality (1) and its consequence for small increments (2) do not require any assumptions on the distribution of X, not even the assumption that X is independent of the load Y. Also note that

Equation (2) is the charge attached to the additional load X. The user that is responsible for X may also encounter an increased charge for any other load due to that user that is already included within Y.

The sample path shadow price of a packet is defined to be 1 if the deletion of the packet from the sample path describing arrivals at the resource would cause one less packet to be dropped at the resource. We ensure that a packet is marked if and only if its sample path shadow price is 1, by marking all the packets that arrive in an overloaded slot.

4 User Types

Here we define some user types used in experiments [7] with the simple slotted model described in the previous section and with a more realistic queueing model that is described in Section 6.

4.1 Elastic-user(w)

The number of packets elastic-user(w) transmits in the slot $(t, t+1)$, is

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

where $x(t)$ and $z(t)$ are internal state variables defined recursively as follows

$$x(t+1) = x(t) + \kappa(w - f(t)), \tag{3}$$
$$z(t+1) = x(t) + z(t) - X(t). \tag{4}$$

Here

- κ is a small positive constant,
- $f(t)$ is the number of marks received at the end of the slot $(t, t+1)$,
- $x(t)$ is the transmit rate of the user and
- $z(t)$ is the fraction of a packet that was not send in the previous slot.

The recursion in Equation (3) attempts to stabilize the rate $x(t)$ to a value where the expected charge per slot is w, for it holds that

$$\begin{aligned}
w - f(t) &= \frac{x(t+1) - x(t)}{\kappa} \\
\Rightarrow \sum_{t=0}^{T-1} (w - f(t)) &= \sum_{t=0}^{T-1} \frac{x(t+1) - x(t)}{\kappa} \\
\Rightarrow w - \frac{1}{T} \sum_{t=0}^{T-1} f(t) &= \sum_{t=0}^{T-1} \frac{x(t+1) - x(t)}{\kappa T} \\
\Rightarrow w - \frac{1}{T} \sum_{t=0}^{T-1} f(t) &= \frac{x(T) - x(0)}{\kappa T} \longrightarrow 0 \text{ as } T \longrightarrow \infty.
\end{aligned}$$

Thus, the expected charge per slot approaches w as time T grows larger. Elastic-user can be seen as a user who chooses to pay an amount w per unit time to get a share of the resource.

Also a more general elastic-user($U(\cdot)$) could be defined in terms of a utility function $U(\cdot)$ [10]. Such a user would behave as elastic-user(w), but w would vary with time, $w = w(t) = x(t)U'(x(t))$ [7].

4.2 Intermittent-user(w)

Intermittent-user(w) is active for a random period with mean t_{active} and behaves then like an elastic-user(w). After that it sleeps (transmits no packets) for a random period with mean t_{sleep}. Consecutive periods are independent and geometrically distributed [13]. At the end of each sleeping period the internal state variables $x(t)$ and $z(t)$ are reset to the values they had at the end of the last active period.

4.3 File-transfer(F,W)

File-transfer(F,W) is a user that has a file of size F to transfer and an amount W to spend. File-transfer wants to transfer the file as soon as possible. We present here a simple algorithm which attempts to satisfy the needs of a file-transfer user. The user using this algorithm attempts to pay $\frac{W}{F}$ per packet on average. If the average shadow price at the moment is higher than this, the user waits until the shadow price decreases. If the average shadow price is lower, then the algorithm increases its rate.

The algorithm is based on elastic-user(w), but the choice of w is adaptive. Let $F(t)$ be the size of the file remaining to be transferred at time t, and let $W(t)$ be W minus the marks received so far. Internal variables $x(t)$ and $z(t)$ are used and they are initially set to zero. The algorithm updates $x(t)$ and $z(t)$ as follows (like an elastic-user but w replaced with $w(t)$)

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

and calculates parameter $w(t)$ as follows

$$w(t+1) = \max\left\{\frac{x(t)W(t)}{F(t)}, w_{min}\right\}. \tag{5}$$

The number of packets to be transmitted $X(t)$ is calculated as in the case of an elastic-user:

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

Equation (5) can be motivated followingly. When the rate is $x(t)$, the rest of the file would be transferred in time $F(t)/x(t)$. The user could afford to pay $w(t+1) = W(t)x(t)/F(t)$ per unit time at this rate. This would increase user's rate if the amount that the user is willing to pay per packet, $W(t)/F(t)$, is greater than its current shadow price per packet $w(t)/x(t)$. If the shadow price is greater than the amount the user is willing to pay, then the rate will decrease. The algorithm varies the rate $x(t)$ so that the price per packet is approximately $W(t)/F(t)$. The lower limit w_{min} in the Equation (5) allows the algorithm to re-test the network after periods when it is too expensive to transfer the file and it forces the user to spend at least w_{min} per unit time on average. Consequently file-transfer(F,W) will eventually transfer the whole file no matter how low the value of W is, at a cost that may exceed W.

4.4 Unresponsive-user(g)

When unresponsive-user(g) is active, it transmits a packet in each slot with probability g, independently from slot to slot. When the unresponsive-user is sleeping, it sends no packets. Consecutive active and sleeping periods are independent and geometrically distributed with mean t_{period}.

5 The Simple Slotted Model

In this and the following section we describe experimental results obtained by Gibbens and Kelly [7]. For more detailed results and for the figures, see [7]. Gibbens and Kelly obtained their numerical results from a computational environment implemented in Java [1].

The experiments in this section concern a simple slotted model described in Section 3. Time is slotted and the resource has the capacity to cope with N packets per slot, with any excess lost. In a given slot the load on resource Y is generated adding together the loads $X_1, \ldots, X_m$ generated by m users in that slot. It holds that

$$Y = \sum_{i=1}^{m} X_i$$

and the number of marks the source i gets at the end of the slot is

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

The feedback delay between the generation of a packet and the receiving of the feedback concerning that packet is one slot. For discussion on more general feedback delays, see [7].

The load a user produces in successive slots depends on the earlier marks received from the network and on the user type. Some simple models of different user types were introduced in the previous section and they are used in the following scenarios.

5.1 Scenario 1

In the first scenario Gibbens and Kelly simulated a resource with a capacity to serve $N = 10$ packets per slot. A population of 20 users was created and the i^{th} user acted as elastic-user(w_i), where $w_i = i \times 0.01$, $i = 1, \ldots, 20$ and $\kappa = 0.1$. For results, see Figure 1 in [7]. Figure 1 in [7] represents a process starting from a stationary state, thas is, the output of the users is at a level where their rate corresponds to an average cost of w_i per slot from the first time slot onwards.

The users obtained throughput proportional to their choice of w, so by varying its parameter the user can influence its throughput. In general the throughput of a user depends also on the total demand of the resource from other users. The experiment also ensured that the average charge per slot was w. It is worth noticing that the user with $w = 0.20$ achieved a throughput of approximately one packet per slot whereas a throughput of 0.5 packets per slot on average required a choice of $w \approx 0.13$. The latter user pays more per packet, as the cost

[1] The environment is accessible at `http://www.statlab.cam.ac.uk/~richard/research/topics/evolution`.

to the user who submits a regular stream of packets is less that the cost to the user who submits packets more intermittently.

This can be motivated mathematically as follows. Suppose that the load due to a user is $X \in \{0, 1\}$ and the total load due to all users is Y. The expected charge to the user that is responsible for the load X is

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

and the expected charge per packet to the user responsible for X is then $\mathbb{P}\{Y > N|X = 1\}$. This conditional probability is a bit lower to a user submitting a regular stream of packets than to a user submitting packets more intermittently.

In this simple model the effect of the choice of κ, the gain parameter, is quite straightforward. A higher value of κ will result in more variance at the equilibrium but it also allows a more rapid convergence to the equilibrium.

5.2 Scenario 2

In the second scenario Gibbens and Kelly simulated a system with intermittend users. As in the first scenario the resource had capacity to serve $N = 10$ packets per slot, but the size of the user population was 100. There was five independent copies of intermittent-user(w_i), where $w_i = i \times 0.01$, $i = 1, \ldots, 20$ and $\kappa = 0.1$. The mean of the random active period was $t_{active} = 1000$ and the mean of the random sleeping period was $t_{sleep} = 4000$. For results, see Figure 2 in [7].

The environment in Scenario 1 was quite static. This second scenario attemps to describe the effects of variation in demand of the resource. The total demand W is defined as the sum of the w's of users not sleeping. The total demand fluctuated between approximately half of and twice its mean value. The average number of packets transmitted per slot was quite stable. The average number of packets marked per slot (that is, the average shadow price at the resource) traces the total demand. The percentage of packets lost fluctuates also in step with the percentage of packets marked, but it is at a lower level.

The set of active users varied but when examining the set of active users, their relative throughput and the rate they received marks was similar to the one obtained in Scenario 1. It can be concluded that even though the demand on the resource fluctuates relatively much, users manage to share the resource between them and keep the total throughput about constant.

5.3 Scenario 3

In the third scenario Gibbens and Kelly had the same population of 100 intermittent-users as in the second scenario and an additional population of 10 users making file transfers. These 10 users were namely file-transfer(F,W_j)s, where $F = 1000$ and $W_j = 200 + 20j$, $j = 1, \ldots, 10$. All file transfers started at slot 0. The lower limit was set to $w_{min} = 0.01$. For results, see Figure 3 in [7].

In this scenario the intermittent-users provided a demand similar to the one in Scenario 2 and we can compare the results from Scenario 2 with the results from Scenario 3 to see the effect the additional population of file-tranfers has on the average shadow price. The population of file-transfers increased the shadow price at the resource at times the file-transfers were specially active. The file-transfers with a higher value of W_j tended to complete sooner, but the order was not strict.

There are several possible alterations of the algorithms *elastic-user* and *file-transfer*. For example, the file-transfer algorithm could set an upper limit on $w(t)$ to protect the user from paying too much. Because of the lower limit w_{min} the file-transfers always finish, but the price may be higher than W. If the lower limit was halved each time it was crossed, the resulting algorithm would provide exponentially growing backoff intervals following the periods when it was too expensive to transfer the file.

6 The Queueing Model

We now extend the simple model described in Section 3 as we introduce a queue with a finite buffer. If Y_{t-1} is the number of packets arriving at the resource in the time interval $(t-1, t]$ and Q_t is the queue size at time t, then we can use the following recursion to describe a queue with a buffer capacity of N.

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

With this recursion the queue is able to serve one packet per unit time. The number of packets that are lost at time t is

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

A *busy period* [3] is defined to start at time t if $Q_{t-1} \leq 1$ and $Q_t \geq 1$, and a busy period ends at time t if $Q_{t-1} = 1$. According to this definition it is possible in our discrete time model for two successive busy periods to adjoin.

A *critical congestion interval* is defined as a period between the start of a busy period and the loss of a packet within the same busy period. When an additional packet is included in a queue's sample path, it increases the number of packets lost by 1, if and only if its time of arrival lies within a critical congestion interval. Otherwise an additional packet does not influence the number of packets lost. Hence critical congestion intervals are equivalent to the overloaded slots of the simple slotted model. Packets arriving during the critical congestion intervals should ideally be marked. However, it is not easy to determine the shadow price of a packet while it is still in queue, for in general it will not be clear whether or not the current busy period will end before a packet is lost.

We introduce here one possible marking mechanism that approximates the ideal behaviour, for others see [7]. All the packets in the queue are marked at the time of a packet loss and in addition a sufficient number of immediately following packets are marked. When a busy period starts, the queue starts counting how many packets arrive in the queue before a packet loss. This is the number of the following packets that the queue needs to mark after a packet loss occurs. This ensures that the correct number of packets is marked in total. Gibbens and Kelly used this marking mechanism in the experiments of Scenario 4.

When Gibbens and Kelly repeated the experiments on Scenarios 1–3 with a queueing resource, they received similar results as with the simple slotted model. However, the proportion of packets lost was lower. [7]

6.1 Scenario 4

In this scenario there were three sub-populations of users.

- Intermittent-users: two independent copies of intermittent-user(w_i), where $w_i = i \times 0.001$, $i = 1, \ldots, 20$ and $\kappa = 0.001$. Active periods had mean periods of $t_{active} = 10000$ and sleeping periods of $t_{sleep} = 40000$.

- Intermittent-file-transfers: these users behave as file-transfer(F,W_j), $j = 1, \ldots, 10$ while the file transfer is in progress, and then sleep a random period with mean $t_{sleep} = 40000$. The file size was $F = 1000$, $w_{min} = 0.001$ and $W_j = 150 + 20j$, $j = 1, \ldots, 10$.

- Unresponsive-users: unresponsive-user(g_k), where $g_k = k \times 0.01$, $k = 1, \ldots, 10$ and $t_{period} = 1000$.

For results, see Figures 4 and 5 in [7].

In this experiment the proportion of marked packets was approximately 0.18 and the proportion of lost packets was 0.014. The utilization rate of the resource was approximately 0.84. Transmitted packets distributed to the three sub-populations followingly: 32% to unresponsive-users, 14% to intermittent-file-transfers and the rest, 54%, to intermittent-users. The overall performance was not significantly disrupted by the presence of the unresponsive-users at this level. As in Scenario 3 the file-transfers willing to pay more achieved faster file transfers, and as in Scenario 2, the throughput of intermittent-users varied respect to their choice of w.

The marking mechanism used in this experiment ensures that a correct proportion of packets is marked by the resource but the marks are not always allocated right according to the exact sample path shadow price of a packet. A packet that arrives early in a busy period (before any loss) may avoid been marked. On the other hand, a packet arriving late at a busy period may receive a mark even though its shadow price is zero.

There are several ways for the resource to anticipate the possible overflow of its buffer and mark packets earlier within the busy periods. Floyd and Jacobson have proposed a probabilistic version for a marking mechanism [4, 5]. The idea is that the resource is able to estimate the probability that a given packet's sample path shadow price will be one. The resource could mark packets randomly as they leave the queue. The probability of a mark could be calculated as the conditional probability, that considering the entire history at the resource until the packet leaves, a later packet will be lost before the end of the present busy period.

Another way to anticipate possible buffer overflow is to adjust the marking mechanism used in this section. This is achieved with a virtual queue. The resource has a virtual queue which has the same arrival process of packets as the real queue but its service rate and buffer capacity are scaled down by the same factor (e.g. 0.9). The marking mechanism is modified as follows: when there is a packet loss in the virtual queue, the real queue starts to mark every packet that is leaving it. It proceeds this until the virtual queue becomes empty.

To illustrate the behaviour of a system using this adjusted mechanism Gibbens and Kelly reran the Scenario 4, modified as follows:

- Real queue: service rate 1.1 packets per unit time, buffer capacity of 11.

- Virtual queue: service rate 1 packet per unit time, buffer capacity of 10.

The virtual queue had a service rate and a buffer capacity that were scaled down by $10/11$ and the virtual queue was equivalent to the queue in the earlier experiment in Scenario 4. The proportion of marked packets remained the same, approximately 0.18, but the proportion of packets lost dropped to about 0.003.

7 Comparison with the Current Internet

In the current Internet the congestion control is provided by dropping packets. The end-nodes often use Jacobson's slow-start and congestion avoidance algorithms [8]. In this section we compare these mechanisms with the ones described in this paper.

In short the slow-start algorithm operates as follows: at the beginning of transmission or after timeout, start by sending one packet. For each acknowledgement received, send two packets. If the network is fast, the slow-start algorithm does not slow down the connection considerably, but if the network is congested, it will take time for acknowledgements to arrive to the source and the slow-start algorithm is actually slow.

The congestion avoidance algorithm works essentially as follows: if the source receives a congestion indication signal (a packet is dropped), it halves the size of the flow. This is called multiplicative decrease. If there is no congestion, then the flow is increased by some contant amount, i.e. additive increase.

To provide congestion signals by dropping packets is quite extreme and better performance should be obtained if the network marked packets without dropping them. An interestion question is also how the rate, at which the feedback signals are generated at the resource, scales with the size of the resource. Consider e.g. the simple slotted model that has a capacity to handle N packets per slot. The load is a Poisson process with mean y. Then the expected number of marks per unit flow is the shadow price $p(y)$ and the expected number of dropped packets per unit flow is $C(y)/y$. Let's consider a heavily loaded resource with $y = N$. Then the ratio of dropped packets to marked packets [7] is

$$\frac{C(N)/N}{p(N)} \approx \frac{(2\pi N)^{-0.5}}{0.5} \longrightarrow 0, \text{ as } N \longrightarrow \infty.$$

Thus, the number of dropped packets per unit flow unterestimates the true congestion at larger resources, if dropped packets at resources of varying size are considered equivalent congestion control signals.

Let's compare the response to congestion indication signals in different algorithms. The elastic-user algorithm responds to a congestion indication signal by reducing the flow by a constant amount. We call this additive reduction. Because congestion indication signals are provided with rate proportional to the size of the flow, it ensures with the additive reduction, that the overall decrease in rate is multiplicative.

On the other hand in Jacobson's congestion avoidance algorithm [8] the responce is to halve the flow. Another multiplicative effect takes place because the response is scaled to the number of congestion indication signals. Both elastic-user and congestion avoidance algorithm have an additive increase.

File-transfer and slow-start also have similar features. They both allow an exponential growth from a low initial rate. An interesting question is how the rate of a new connection should grow. Should it grow from a slow initial rate or should the connection be able to start immediately transmitting at a higher rate possibly depending on earlier experiences of the network? The answer the marking mechanism presented in this paper gives is that the end-nodes can decide themselves. With the marks they receive from the network they have enough information to make a reasonable choice.

Finally, let's consider some possibly problematic points in the model presented here and solutions to them.

There is no distinction between packets lost and merely marked

In the model presented there is no distinction between packets that are marked and lost and packets that are merely marked; they all are marked packets. However, this distinction is not as important to congestion avoidance as it is to retransmission behaviour. The number of marked packets is significally higher than the number of dropped packets. The end-nodes may even try to learn network characteristics from the ratio of dropped to marked packets. If there is, for example, a connection that has a noisy radio link dropping packets for other reasons than congestion, then the end-node can notice a change in the ratio and maybe decide to use more forward error correction.

It is possible that a packet is marked multible times

In the network it is certainly possible that a packet is marked more than once, as the shadow price is additive. However, if the proportion of marked packets is small it will be enough to know if the packet is marked or not at any resource, so a single bit record is enough. Also if the load of the network is not extremely high, it is relatively unlikely that a packet would be marked even twice. This would mean that on a certain route the packet's existence would cause an increase of two in the number of packets lost by the network.

Packets are not always of the same size

We have assumed in our discussions that all the packets in the network have the same size. In general, this is not the case since packets have different sizes. There are several possible approaches in an environment with packets of different sizes. If the load on a resource is proportional to the length of the packet, then a feasible solution would be to charge a marked packet a fixed amount per byte of the packet. If there is a maximum packet size, then an appropriate solution would be to mark a packet passing through the resource at a time of congestion with a probability that is proportional to its length. Both these approaches make the expected charge proportional to the length of the packet. The latter marking mechanism can also be used in networks where the load on other resources is proportional to the length of a packet and on others not. The resources where the load is not proportional to the length of a packet can then mark the packet arriving at a time of congestion with a fixed probability, e.g. 1.

8 Game-theoretic Approach: Nash Equilibria

In the previous sections of this paper we have discussed a model where a user does not anticipate the effect of its own actions on prices. This is a reasonable assumption if the user occupies only a small fraction of any single resource in the network. However, if the user occupies more and more of a resource then eventually it will have an effect on prices.

In this section we use game-theoretic models [15] to illustrate how users might behave if they can anticipate their own impact on prices. The users act as players in a game where the aim is to allocate a scarce resource as efficiently as possible. Different user types represent the different strategies the players have. The discussion here is rather informal and brief, and readers interested in more details and formalisms are suggested to consult [7].

Let the utility of user r be linear, that is,

$$U_r(x_r) = \alpha_r x_r, \tag{6}$$

where α_r is a constant. The users share the unit capacity of resource in proportion to the w's they choose. In situations where the user r assumes that its choice of w has no effect on the price of the resource, it chooses between selecting w_r very large or small. If the current charge per unit flow is greater than α_r, the user chooses w_r to be large, and if the current charge is less than α_r then the user chooses w_r to be small.

However, if the user realises that its choice of w_r has an effect on the flow per unit charge, it is motivated to choose w_r so that

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

is maximized, where W_r is the sum of other user's ws. This means that the user tries to maximize its "profit" (the utility the user gains - the price user is willing to pay). When each individual user maximizes its choice, the *Nash equilibrium* [15] is found. A Nash equilibrium is a set of strategies with the property that no player can benefit by changing his strategy while the other players keep their strategies the same. In this case it holds at the Nash equilibrium that

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

where $\widehat{W}$ satisfies

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

Thus, $\widehat{W}$ is a threshold. If the user has α_r that is less than $\widehat{W}$ then it will choose $w_r = 0$. It can be shown [7] that when users can predict their own effect on prices, they will pay less than they otherwise would. As in auction theory [18]: if users have market power, they shade their bids.

This analysis modelled users that are able to choose the parameter w i.e. their strategy was essentially that of elastic-user(w). As the Nash equilibrium depends upon the strategies that are allowed for players, other algorithms have different Nash equilibria. When using the linear utility function (6) an appropriate algorithm for user r is file-transfer(F, πF). If we suppose that user r can choose the amount it pays per packet, π_r, the resource allocates all its capacity to the user(s) choosing the highest price. This is fundamentally a simple auction [16, 18]. At the Nash equilibrium the whole capacity is provided to the user willing to pay the highest price, i.e., the user s with highest α_s will choose π_s to be a bit higher than the second highest value in $\{\alpha_r | r \in R\}$.

As a concluding remark, we can say the following. If the user accounts for a large part of the load on the network, then its rational behaviour is to back off slightly relative to the algorithm where the user does not anticipate the effects of its own actions.

9 Summary

We have discussed a simple model using which it is possible to achieve an efficient use of a scarce resource as the resource marks packets at times when it is overloaded. In a network with cooperative end-nodes the information the end-nodes gain from the marks is enough to ensure efficient allocation of resources. If the end-nodes in the network are uncooperative, a small fixed charge per each marked packet is needed to ensure that the end-nodes have right incentives.

The marking mechanism discussed here provides a rational basis for a usage-related charging scheme for a network. Even small usage-related costs influence the users to use congestion control, what improves the efficiency of the network.

We introduced several algorithms that represent different user types with different needs. We compared these algorithms to the ones used in the current Internet.

We also discussed briefly a game-theoretic approach where the users can anticipate their own effects on prices. It turns out that in such cases the user pays less than it would pay if it did not realize the effect its own choices have on prices.

The work of Gibbens and Kelly [7] in the field of congestion control has inspired further work, for example [11, 17]. Kunniyur and Srikant [12] have also improved the queueing mechanism proposed by Gibbens and Kelly as they introduced Active Virtual Queue which updates dynamically the virtual queue capacity.

References

[1] Bazaraa M. S., Sherali H. D., Shetty C. M. 1993. Nonlinear Programming, Theory and Practice. 2nd edition. John Wiley and Sons.

[2] Clark D. D. 1996. Adding Service Discrimination to the Internet. Telecommunications Policy, 20(3), pp. 169–181.

[3] Cooper R. B. 1981. Introduction to Queueing Theory. 2nd edition. Elsevier, New York.

[4] Floyd S. 1994. TCP and Explicit Congestion Notification. ACM Communications Review, 24(5), pp. 10–23.

[5] Floyd S., Jacobson V. 1993. Random Early Detection Gateways for Congestion Avoidance. IEEE/ACM Transactions on Networking, 1(4), pp. 397–413.

[6] Floyd S., Fall K. 1999. Promoting the Use of End-to-End Congestion Control in the Internet. IEEE/ACM Transactions on Networking, 7(4), pp. 458–472.

[7] Gibbens R. J., Kelly F. P. 1999. Resource Pricing and the Evolution of Congestion Control. Automatica, 35(12), pp. 1969 –1985.

[8] Jacobson V. 1988. Congestion avoidance and Control. In Proc. ACM SIGCOMM '88, pp. 314–329.

[9] Kelly F. P. 1997. Charging and Rate Control for Elastic Traffic. European Transactions on Telecommunications, 8(1), pp. 33–37.

[10] Kelly F. P., Maulloo A. K., Tan D. K. H. 1998. Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability. Journal of the Operational Research Society, 49(3), pp. 237–252.

[11] Key P., McAuley D. 1999. Differential QoS and Pricing in Networks: Where Flow Control Meets Game Theory. IEE Proc Software, 146(2), pp. 39–43.

[12] Kunniyur S., Srikant R. 2001. Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management. Sigcomm 2001, San Diego. Available via `http://comm.csl.uiuc.edu:80/~srikant/Papers/sigcom.ps`.

[13] Laininen P. 1998. Todennäköisyys ja sen tilastollinen soveltaminen. Otatieto, Helsinki.

[14] MacKie-Mason J. K., Varian H. R. 1995. Pricing the Internet. In B. Kahin and J. Keller, editors, Public Access to the Internet. MIT Press, Cambridge, MA, pp. 269–314.

[15] Myerson R. B. 1997. Game Theory: Analysis of Conlict. 3rd edition. Harvard University Press, Harvard.

[16] Vickrey W. 1961. Counterspeculation, Auctions and Competitive Sealed Tenders. The Journal of Finance, 16(1), pp. 8–37.

[17] Wischik D. 1999. How to Mark Fairly. In MIT Workshop on Internet Service Quality Economics. Available via `http://www.statslab.cam.ac.uk/~djw1005/Stats/Research/marking.ps`.

[18] Wolfstetter E. 1994. Auctions, an Introduction. Journal of Economic Surveys, 10(4), pp. 367–420.