Distributed Control and Resource Marking Using Best-Effort Routers

Richard Gibbens, University of Cambridge
Peter Key, Microsoft Research, Cambridge

Abstract

We present a method for creating differential QoS where control is in the hands of the end system or user, and the network distributes congestion feedback information to users via packet marking at resources. Current proposals for creating differential QoS in the Internet often rely on classifying packets into a number of classes with routers treating different classes appropriately. The router plays a critical role in guaranteeing performance. In contrast, there is a growing body of work that seeks to place more of the control in the hands of the end system or user, with simple functionality in the router. This is the approach outlined in this tutorial article: using insights from economics and control theory we show how cooperation between end systems and the network can be encouraged using a simple packet marking scheme. The network distributes congestion feedback information to users via packet marking at resources, and users react accordingly to obtain differential QoS.

The current Internet is based on a single best-effort class of service, where users or flows are expected to conform to a congestion control behavior based on that of TCP. The primary feedback signal is packet loss, and end systems usually have to deduce this information indirectly from acknowledgments sent by receivers. This framework has a number of disadvantages: it does not provide differential quality of service (QoS), it is challenged by streaming applications based on UDP which do not follow TCP's congestion avoidance behavior, and it relies on the end systems behaving "nicely" when there is no clear incentive for them to do so.

Incentives are naturally handled within an economic framework, and there is a growing body of work that seeks to apply economic insights to problems in communication networks. Congestion occurs in a network when there are scarce resources, and traditionally telecommunication networks have turned to engineering methods to allocate these resources, often using some form of centralized control to allocate or deny resources.

In this article we survey recent developments in distributed control and resource marking motivated by insights from economics and control theory. These can be used to create differential QoS for best effort traffic, with minimal complexity in the routers. All the routers need do is mark packets, for example by setting a simple flag in the IP header, which are returned to the sender. We concentrate on flow control schemes that can be thought of as natural generalizations of TCP. For such schemes, the feedback signals can be communicated in the acknowledgment packets (ACKs). Differential QoS is achieved by allowing different reactions to the receipt of feedback signals, in ways we make explicit later. For example, if an application can tolerate receiving marks at twice the rate of another application, it will receive roughly twice the throughput. The QoS differentiation is relative, which is appropriate for best-effort traffic, and can be thought of as providing weighted fair sharing of resources.

In the next section we outline some of the basic ideas and concepts. We explore end system behavior, which we illustrate with a specific rate adaptation scheme. We look at how marking can be applied at routers. We discuss how the interplay between the network and users can be seen as a distributed game. We also provide a specific simulation example of how differential QoS can be achieved, and how it can coexist with standard TCP. The important issue of how to ensure that different applications behave cooperatively, the connection between marking and charging, and implementation issues are also discussed.

Background Ideas and Concepts

In packet networks the various scarce resources are shared among users. Common network resources are the bandwidth of a link, the buffer capacity at an output port, or the capacity of a server, such as its processing capacity or memory. The degree of resource usage is primarily determined by the rate at which the sender transmits packets into the network. The problem considered in this article is how to determine an appropriate rate allocation and hence sharing of resources between users.

In IP networks, such as the current Internet, the determination of sending rates is carried out by a distributed algorithm at the sender that adjusts the rate of transmission while monitoring feedback information given by packet acknowledgments returned by the receiver. One source of feedback available to the sender is the successful arrival of the packet at the receiver or the loss of the packet within the network (corresponding to the absence of an acknowledgment within a timeout period). This particular form of feedback informa-

0890-8044/01/$10.00 © 2001 IEEE

tion requires no additional support from the network beyond the normal forwarding of packets from sender to receiver and the return of ACKs. Alternative forms of feedback information will also be considered in this article that explicitly signal the level of network congestion to the sender. This type of feedback information requires additional support by the network resources to convey the level of congestion; but, as we shall demonstrate, there are additional benefits. The Internet Engineering Task Force (IETF) Explicit Congestion Notification (ECN) proposal [1], which marks a packet if a router is congested, is one natural way of conveying feedback information back to the user.

Within today's Internet the rate allocation of TCP connections is determined by the TCP congestion avoidance algorithm that operates by linearly increasing the sending rate until a packet loss is detected, when the sending rate is halved. Thus, the congestion avoidance algorithm attempts to balance the level of packet loss against the utilization of resources.

Although there are many variants of the congestion avoidance algorithm, we begin by describing the simplest model for the resulting rate allocation. Several authors have described equations giving the steady state throughput of TCP in the face of a constant packet drop probability. These all produce the familiar "inverse square root" type dependence [2]; the simplest determines the rate for a single connection, x, by the expression

$$x = \frac{1}{T}\sqrt{\frac{2}{p}},$$

where T is the round-trip time for the connection and p is the probability of packet loss along the path from sender to receiver.

Notice that this expression shows that there is a strong dependence on the round-trip time, T. The longer the round-trip time the lower the rate, x, for a given level of packet loss.

It can be shown that this expression for the rate also maximizes the objective function

$$U(x) - px$$

where

$$U(x) = K - \frac{2}{T^2 x}$$

and K is an arbitrary constant. Thus, the TCP congestion avoidance algorithm can be seen to determine a rate that optimizes a *net utility* given by the difference of $U(x)$ and a *cost* term, px, related to the rate of packet loss.

In other words, reverse engineering the TCP congestion avoidance mechanism, it behaves as if an application values bandwidth according to the utility function $U(x)$, suffers damage at a constant rate px, the rate of packet loss, and attempts to maximize net utility (utility – cost). The function $U(x)$ makes explicit the extra "value" users place on getting more bandwidth.

Variants of the basic TCP congestion avoidance algorithm will introduce alternative functions, $U(x)$, but leave the overall interpretation the same.

The approach to rate allocation followed by the TCP congestion avoidance algorithm will allocate similar rates to connections which pass through resources with similar levels of congestion and have similar round-trip times. There is no scope for different rates for connections based on the type of user or application, that is, for differential QoS given by alternative choices for the function $U(x)$. In effect, the TCP algorithms make the choice of $U(x)$ on behalf of all users.

To better understand the sharing of resources, it is helpful to draw on several economic insights. One such insight is that use of common scarce resources produces a *negative externality*, that is, the choice of a user to increase its sending rate not only affects the congestion it experiences but also detrimentally affects the level of congestion seen by other users who share common resources. In a similar way, a road user on a busy highway contributes additional delay for other road users. The cost term in the optimization above should ideally take account of this externality so that users balance the advantages to them of a higher sending rate against the full consequences of their actions on all users of the network — not just on themselves — and in this way achieve system optimal outcomes. It is possible to interpret the cost term as a *congestion price* — the higher the level of congestion, the higher the congestion price and the greater the incentive to lower the sending rate.

A related economic insight, not developed in this article, is that the congestion cost term can be related to the marginal cost of incremental capacity expansion. Thus, the network operator observes accurate signals (based on the congestion prices) to increase resource capacity.

Later, we shall describe approaches based on the use of ECN feedback marking to determine a congestion price function for the cost term that attempts to capture the externalities involved with the use of network resources.

We should also note that we are assuming cooperative behavior on the part of the users in choosing to follow the recommended TCP congestion avoidance algorithm to determine the user's sending rate and hence their share of the available resources. This is a very important assumption and one that is likely to be under increasing pressure, particularly as the range of applications using the common Internet infrastructure grows.

Ways of ensuring or encouraging user adoption of cooperative behavior, through the use of private centrally managed networks or by charging mechanisms, are discussed later.

In this article we concentrate on the problem of rate allocation for relatively long-lived TCP connections where a feedback information channel is already present to ensure reliable data transmission. While the main focus of our discussion is TCP connections, we briefly mention extensions to rate adaptation for UDP-based streaming media applications.

End System Behavior

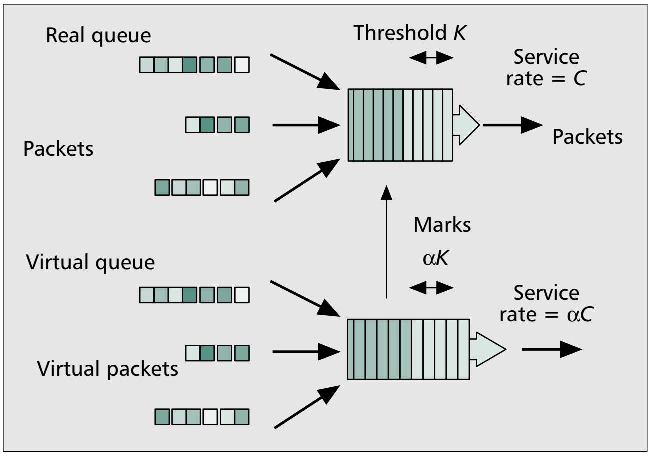
The work of Kelly *et al.* [3, 4] has shown how the optimization framework described above may be applied in a network where users adapt their sending rate based on ECN feedback marking information.

Suppose that user r wishes to determine its sending rate, x_r, such that the level of marking it receives is w_r marks/unit time. Thus, the quantity w_r can be interpreted as a willingness to pay parameter for marks received per unit time.

In a distributed context, suppose each user, r, adapts its sending rate, x_r, to match its willingness to pay parameter, w_r, by means of the following simple rate adaptation algorithm:

$$\frac{dx_r(t)}{dt} = \kappa_r(w_r - x_r(t)p_r(t))$$
$$p_r(t) = \sum_{j:j\in r} p_j(y_j),$$

where y_j is the aggregate load through resource j, $p_r(t)$ is the sum of the congestion prices at resources j along the user's route, and κ_r is a small positive constant.



■ Figure 1. *Real and virtual queues. The virtual queue has a service rate of αC and marks when above the threshold αK.*

Here the congestion price refers to the proportion of offered packets marked by the resource. These prices can be contrasted with the term p used in the simple model of TCP of the previous section that measured cost by probability of packet loss. Hence, users observe feedback in terms of marked packets at rate $x_r p_r$ and adjust their sending rate according to the difference between their expected and actual marking rates. This algorithm has a linear increase and a multiplicative decrease. The detailed choice of marking function, $p_j(y_j)$, used to indicate congestion levels at a resource is the subject of a later section.

The parameter κ_r acts as a gain parameter that controls the rate of adaptation, and has to be set small enough to ensure stability. It is possible to show that if all users follow such a rate adaptation algorithm, the rates will converge to the equilibrium point where they achieve their expected marking rate of w_r marks/unit time and send at rates given by

$$x_r(t) = \frac{w_r}{p_r(t)}$$

independent of round-trip times.

The above discussion assumes that the parameters w_r are fixed. Allowing users to dynamically select w_r according to their own preferences (e.g., according to their own private utility function) has been shown to generate outcomes that are system optimal [3]. Here system optimal means maximizing the aggregate net utility of the users.

This description is only a simplified model that ignores the time-delayed nature of feedback information. The exact choice of the gain parameter, κ_r, is a current research topic, but may be taken as a suitable fraction of the inverse round trip time.

Marking Algorithms for Best-Effort Routers

In this section we look at ways of determining the congestion marking function, $p_j(y_j)$, at resource j in terms of the aggregate load y_j. This function needs to record the cost of the externality resulting from any additional load placed on the resource. Active queue management (AQM) schemes such as Random Early Detection (RED) seek to give early warning of increasing congestion by use of ECN marks that mark a single bit in a packet. We would also wish the level of marking to signal the congestion costs involved.

AQM schemes are usually based on some form of threshold marking: for example marking packets with the single ECN bit when a threshold is exceeded. In a virtual queue (VQ) marking scheme [5] we assume that packets which arrive to the real queue are also fed into a VQ with a service rate and buffer length each reduced by multiplying by a factor $\alpha < 1$. Packets are then marked when the VQ exceeds some threshold value. This is illustrated in Fig. 1.

Note that the VQ only performs marking, no real packets are scheduled in it, and hence it can often be implemented by means of a simple counter.

Having the VQ run at a slower rate means packets are served more slowly, hence queues build up faster during periods of congestion producing an early warning of problems in the real queue. The parameters of the VQ can be thought of as tuning parameters set and monitored by the network operator in order to appropriately trade-off packet loss against utilization. An illustration of this tradeoff is given in [6] where the threshold value is denoted by K and packets are marked when the VQ size exceeds K. Using a simple queueing model, [6] gives the relation

$$\alpha = \left(\frac{1}{K+1}\right)^{1/K},$$

where α is the reduction factor in service rate for the VQ. Here the VQ is attempting to signal the congestion cost of the real queue exceeding the value K. For example, if $K = 10$ then $\alpha = 0.78$, so the VQ runs at 78 percent of the rate of the real queue, implying we aim to run the real system with utilization below 80 percent. We can increase K and hence the utilization of the system: for example, setting $K = 50$ equates to $\alpha = 0.92$. However, the system becomes harder to control the larger the value of κ.

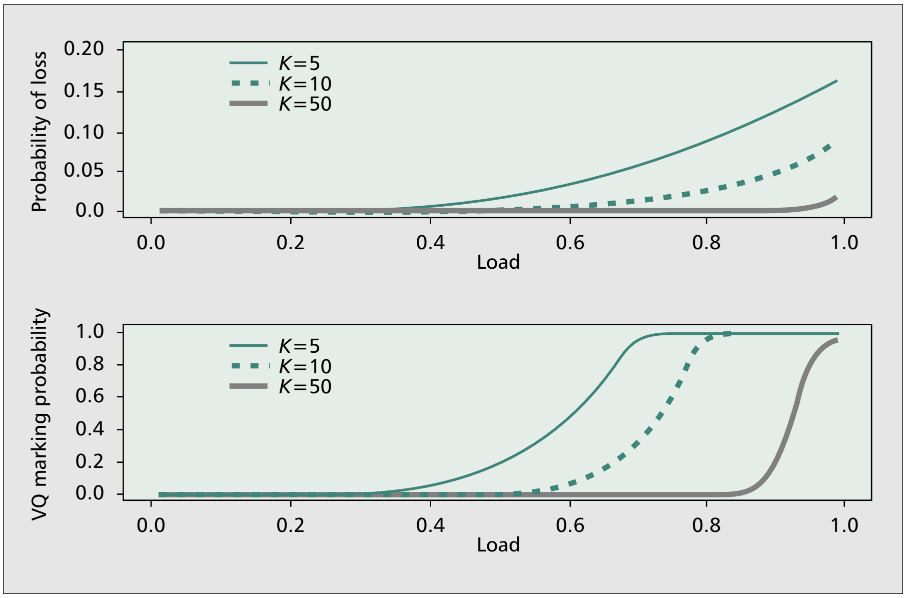
Using the model of [6], Fig. 2 illustrates the effect of using a threshold marking in either the real queue or the VQ, with feedback based on loss (top) in the real queue, or marks in the VQ (bottom). The top diagram shows how loss increases as a function of load when the (real) queue discards packets if the buffer size exceeds K, where the different curves (labeled 5, 10, and 50) correspond to different values of K. Except when K is very small, the loss probability increases sharply with load, illustrating the difficulty in trying to control load via feedback signals based on loss. Only a fraction of packets are lost if the load is less than capacity; hence, the control system has to rely on a few users backing off strongly to prevent overload.

Contrast this with VQ marking, shown in the bottom of Fig. 2, where the buffer size is 100, and packets are marked as the VQ threshold exceeds K. Now we start to mark much earlier: indeed, for a threshold of 10, we start to mark heavily once the load goes above 0.6, even though we lose hardly any packets at this load (as shown by the top figure). There is strong pressure to keep the load below the design level, and if users can tolerate a marking rate of say 20 percent, the control will keep the load below the appropriate design level. The high level of marking (compared to, say, either loss-based or pure threshold marking) enables all the users to react lightly, rather than a few having to react strongly.

Observe that having the actual buffer size larger than the threshold at which we mark in the VQ enables us to have low loss and low delay: if connections do react to mark signals, the delay is kept small and very few packets can be lost, since excursions of the real queue above the real buffer size are rare.

Flow Control Experiments and Distributed Games

The previous sections considered the optimization framework with users adapting their sending rates in response to feedback signals in such a way as to keep the marking rate fixed. Howev-



■ Figure 2. *The top panel shows the loss probability in the real queue as a function of the load. The bottom panel shows the marking probability for the virtual queue as a function of load.*

er, not all users will want to adapt to congestion levels in this relatively smooth fashion. In this section we explore some of the consequences of alternative rate control strategies.

The choice of rate control strategy forms a game. Not only does the user's choice of strategy affect themselves; it also affects the other users of the common network resources.

Key and McAuley [7] explore the idea of QoS as a distributed game, where users "play" against each other and the network. A real distributed game is used to assess the effectiveness of strategies: a game server creates an artificial network, recreating nodes, link delays, and marking strategies, and users compete in such an environment. Such a distributed game simulator has been built at Microsoft Research, Cambridge, and allows real users to compete against each other. A strategy is a process that decides when to send packets (with some associated token acting as a packet identifier), receives information in the form of ACKs and marks, and decides how to adjust the sending rate in the light of this information. A simple text-based protocol is used to communicate with the game server over a TCP connection.

An example game might be to transfer a certain amount of data in a given time at minimum cost, a type of file transfer game, and [8] gives examples of how simple strategies perform. If users are only prepared to pay a fixed amount per packet [4], the users are observed to stall (stop sending) if the price goes too high, in the manner of an auction. This contrasts with the willingness to pay (WTP) strategy considered earlier where users adjust their sending rate. More generally, we can envisage strategies where users adapt their rate in some more complex way, and perhaps need some minimum rate for real-time service. Key and Massoulie [9] describe the interaction of such streaming and file transfer users in a large system.

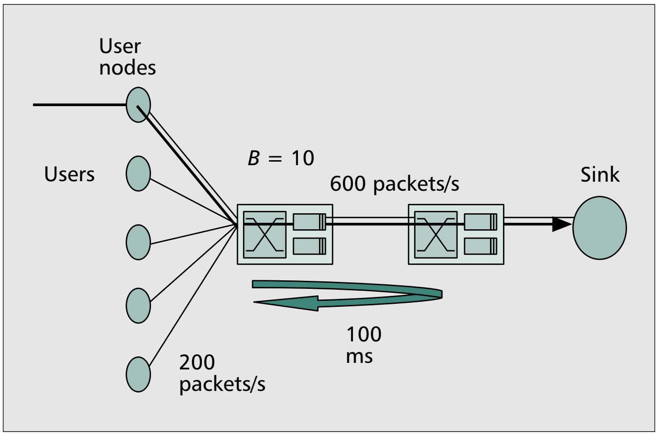
As a simple example of interactions between different strategies, consider a number of WTP strategies and a TCP strategy that is ECN-aware. Such an example is helpful in discussions of incremental deployment of new strategies.

Figure 3 shows a simple network where a number of users are connected to one of five user (or access) nodes, each connected by a 200 packet/s link to a router. There is a single bottleneck link connecting the two routers, of capacity 600 packets/s, and a single sink. The first router uses VQ marking, with a real buffer of 10 but a virtual buffer of size 9 packets, and a virtual rate of 550 p/s given by 90 percent of the bottleneck link rate. The bottleneck link has a 100 ms round-trip time delay and the access links each have small round-trip times of 10 ms.

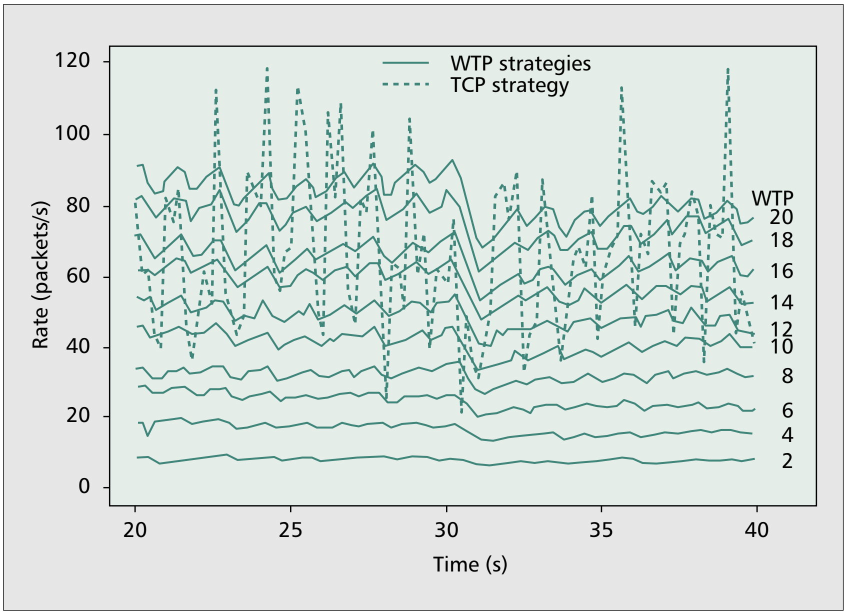
We now connect 10 users to the system, distributed among the user nodes, with different willingness to pay parameters evenly distributed in the range 2–20. In addition, there is a single TCP user. Note that the aggregate willingness to pay (110/s) is less than the rate at which marks can be charged (600/s). Four extra users arrive at time t = 30, each having a WTP parameter of 10 in the middle of the range, effectively producing a 40 percent increase in offered load.

Figure 4 shows the achieved rates for the various connections recorded over a 20 s period. The solid lines give the rates for the various WTP connections: as expected, those users who having a higher willingness to pay get higher throughputs. Indeed, those who are willing to pay twice as much get approximately twice as much, but not exactly twice as much since we are dealing with a system with stochastic fluctuations. After time 30, when there is a sudden increase in load, the rates drop, but very quickly converge to their new values. The dashed curve shows that the behavior of TCP (ECN-aware TCP) is much more variable than the WTP strategy: by its nature, the variability of the throughput is much higher since it has a more dramatic decrease reaction. It also responds less well to the change in offered load, and loses several packets. But overall, TCP still sees the same per packet marking probability as the WTP schemes, a value which is approximately 20 percent in this example.

Figure 5 shows the evolution of the VQ size, where the marking periods are the times the queue size exceeds the VQ threshold value of 9. There is a large positive excursion in the queue size around t = 30, when the load is suddenly increased, and in fact this is the only time loss occurs in the real queue where just two packets were lost in total. Hence, with a relatively small buffer, we have been able to achieve almost no loss, and yet run the system at above 80 percent utilization.



■ Figure 3. *This figure shows the simple network of users and routers.*



■ Figure 4. *This figure shows the achieved rates of the WTP users over a 20 s period with willingness to pay parameters as shown. In addition, the achieved rate of a single TCP user is shown for comparison.*

Note that the TCP user achieves a long-run mean rate similar to that of the WTP strategy with parameter w = 16, but has a far higher variance. Unlike a WTP user, the TCP user's throughput depends on the associated round-trip time, T_r. This example shows that TCP can coexist perfectly well with other strategies, and indeed the presence of smoother strategies like WTP benefits TCP by giving it a higher throughput than replacing all the WTP schemes with TCP. Compared to TCP, WTP or other similar rate adaptive strategies give a smoother behavior, a better differentiated service model, and a throughput related to $1/p_r$, where p_r is the marking rate, compared with $1/\sqrt{p_r}$ for the case of TCP.

Implementation Issues

Crowcroft and Oechslin [10] demonstrate an early attempt at creating user behavior which mimics that of multiple TCP streams. These experiments still used loss as the feedback signal. We now look at several examples where ECN is used as the feedback signal.

The current ECN RFC is experimental, and only defines a behavior for TCP. By using ECN to carry the same binary congestion information, exactly as in the RFC but with a slightly different marking algorithm in the routers than RED, and with flexible end system behavior, we can create differential QoS. ECN and ECN/RED support is likely to exist across a range of operating systems including Linux/FreeBSD and Windows.

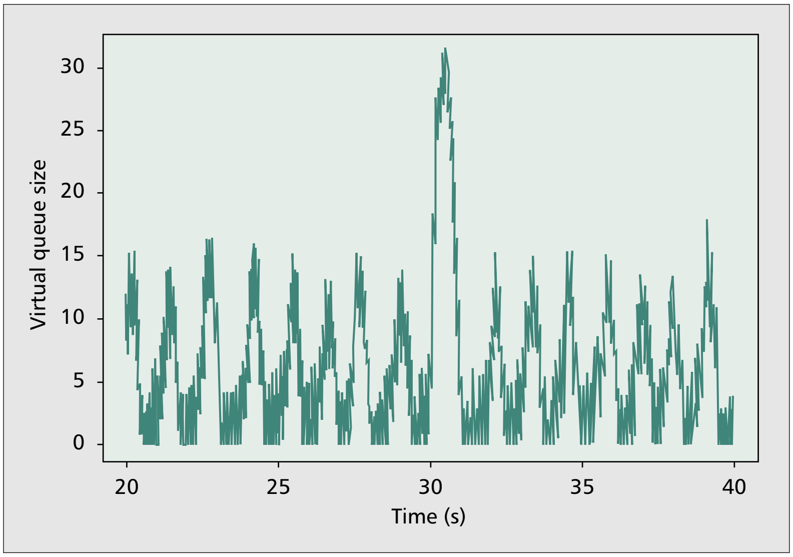
We have described a general framework that assumes aggregation of marks along a route. ECN provides a single bit mark; however, it is believed that little is lost by using a single bit (effectively taking the maximum of marks along a route) compared with the sum, provided the overall marking rate is not too high. An alternative to using single bit feedback at the IP level is to use some form of direct feedback at an aggregate level. Barham and Stratford [11] describe an experimental implementation built on top of Windows2000 which uses a third-party traffic controller to alter the rates of the Windows2000 traffic shapers in response to feedback signals. This has the attraction of working with existing applications.

The end system behavior may be mandated by the operating system (as is currently the case), or may be under system/network management control in a private intranet. In this article we describe a very simple rate adaptation scheme for elastic flows. In a similar way, rate adaptation schemes together with marking can be considered for streaming real-time services. Such end system behavior can coexist with TCP. If the TCP is not ECN-aware, packets could be dropped instead of marked. Alternatively, the behavior we describe could be part of a DiffServ service class, or partitioned to exist within a virtual private network. The question of congestion pricing for aggregates is discussed in [12].

The preceding sections have assumed users cooperate by conforming to an agreed strategy, such as a WTP strategy with a specific choice of parameter w. The choice of w could be mandated by the system; for example in a private intranet management may fully specify the behavior of users and end systems. In situations where such cooperation breaks down, other mechanisms may be required. For example, marks may be interpreted as a pricing signal to the users, and if they represent real or virtual money, incentives can be aligned using charging, as described in Gibbens and Kelly [4]. The precise manner in which charging is implemented depends on many complex factors beyond the scope of this article, but may include a substantial level of aggregation.

Summary

In this article we discuss an approach to providing differentiated quality of service within a network of best effort routers. The routers are assumed to be able to signal congestion to end systems by means of packet marking during congested periods and to forward packets through simple FIFO queues without the need for complex priority scheduling mechanisms. The network makes



■ Figure 5. *This figure shows the evolution of the virtual queue size before, during, and after time* t *= 30 when four additional users join the system. The virtual queue marks whenever the virtual queue size exceeds the threshold value of 9.*

no guarantees to end systems concerning performance, but provides feedback information that may be expected to produce system optimal behavior on the part of cooperative end systems.

Acknowledgments

The authors wish to acknowledge the helpful comments of the referees. Richard Gibbens wishes to acknowledge the support of the Royal Society and the EPSRC under grant GR/M09551.

References

[1] K. Ramakrishnan and S. Floyd, "A Proposal to add Explicit Congestion Notification (ECN) to IP," RFC 2481, 1999; ftp://ftp.isi.edu/in-notes/rfc2481.txt

[2] M. Mathis *et al.*, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm," *Comp. Commun. Rev.*, vol. 27, no. 3, 1997.

[3] F. P. Kelly, A. K. Maulloo, and D. K. H Tan, "Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability," *J. Op. Res. Soc.*, 1998, no. 49, pp. 237–52; http://www.statslab.cam.ac.uk/~frank/rate.html

[4] R. J. Gibbens and F. P. Kelly, "Resource Pricing and the Evolution of Congestion Control," 1999, *Automatica*, no. 35, pp. 1969–85; http://www.statslab.cam.ac.uk/~frank/PAPERS/evol.html

[5] R. J. Gibbens and F. P. Kelly, "Distributed Connection Acceptance Control for a Connectionless Network," *Proc. Int'l. Teletraffic Cong. 16*, 1999, P. B. Key and D. G. Smith, Ed., Elsevier, pp. 941–52.

[6] F. P. Kelly, P. B. Key, and S. Zachary, "Distributed Admission Control," *IEEE JSAC*, Dec. 2000, vol. 35, no. 12, pp. 2617–28; http://research.microsoft.com/research/network/publications/dac.htm

[7] P. B. Key and D. R. McAuley, "Differential QoS and Pricing in Networks: Where Flow Control Meets Game Theory," *IEEE Proc. Software*, 1999, vol. 146, no. 2, pp. 39–43.

[8] R. J. Gibbens and P. B. Key, "The Use of Games to Assess User Strategies for Differential Quality of Service in the Internet," *1999 Wkshp. Internet Service Quality Economics*, MIT, http://research.microsoft.com/research/network/publications/gibkey1999

[9] P. B. Key and L. Massoulie, "User Policies in a Network Implementing Congestion Pricing," *Wkshp. Internet Service Quality Economics*, 1999, MIT, http://research.microsoft.com/research/network/publications/

[10] J Crowcroft and P Oechslin, "Differentiated End-to-end Internet Services Using a Weighted Proportionally Fair Sharing TCP," *ACM Comp. Commun. Rev.*, 1998, no. 28, pp. 52–67.

[11] N. Stratford and P. Barham, MSR Cambridge, MSN 2000, Coseners House, http://www.acu.rl.ac.uk/msn2000/talks/Neil_Stratford.pdf

[12] P. Key *et al.*, "Congestion Pricing for Congestion Avoidance," Microsoft Res. tech. rep., MSR-TR-99-15, 1999, http://research.microsoft.com/pubs/

Biographies

RICHARD GIBBENS (R.J.Gibbens@statslab.cam.ac.uk) is a Royal Society University Research Fellow in the Statistical Laboratory, University of Cambridge. He has worked on the design and analysis of dynamic routing strategies in telecommunication networks, resulting in the development of the dynamic alternative routing strategy now in operation in the British Telecom trunk network. His main area of current interest is quality of service issues in IP networks.

PETER KEY (peterkey@microsoft.com) is a senior researcher at Microsoft Research's European Research Centre in Cambridge, which he joined in 1998. From 1982 to 1998 he was at BT Labs, where he managed the Network Performance team, and was involved with the development and introduction of dynamic alternative routing (DAR), ATM traffic management, and network design. He was Technical Co-Chair of the 16th International Teletraffic Congress in 1999. His current research interests focus on distributed control, application performance, and quality of service.