The case for a new IP congestion control framework

This is a slightly revised version of CUED/F-INFENG/TR.434

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

July 10, 2002

Abstract

There is now an increased understanding of how to engineer end-system based flow controls that offer low-loss and low-delay to *all* packets even when congestion is present. These new controls use explicit congestion notification and are part of a framework that adds flexibility to the resource allocation policies that can be expressed at end-systems. Altering a well-proven protocol suite such as TCP/IP could be risky and not without costs encouraging many to ask: Why bother? This position paper examines the potential system costs and benefits of such a framework. It concludes that the effort may well be worth it.

1 Introduction

The Internet architecture[1] provides a single layer of control which has allowed rapid evolution [24] of networking technologies below the IP layer,[2] applications running on IP networks, and network management structures within the IP layer. End-system based congestion control is the standard mechanism for sharing heterogeneous network resources in an IP network. This closed loop control is instrumental in the Internet's operation, organization, and current packet delivery model. It is argued here that minimal changes to the IP congestion control framework can help the Internet support new applications and provide a basis for more flexible resource allocation policies. It is not argued that these changes will relieve all the pressures on the architecture; the changes are likely to be orthogonal or complementary to those that might be needed to address today's network security and regulatory policy expression issues.

The resource in a data network is the transmission capacity of its constituent links. Resource contention occurs when the traffic arriving at a link exceeds its capacity. Define persistent contention to be when the mean traffic arrival[3] is greater than the link's capacity and any other contention as transient; let persistent contention be termed congestion. Today congestion is signaled to end-systems implicitly through the detection of packet drops with a sensitivity to queuing delay also embedded in the TCP protocol. The congestion control algorithm which specifies how TCP stacks react to congestion signals has remained largely unchanged since its original proposal [6]. The closed loop control of TCP reduces packet loss and the possibility of congestion collapse; congestion collapse had occurred in the Internet when congestion was left uncontrolled. An outcome of this control is the allocation of a scarce resource amongst those wishing to use it. It is commonly agreed that the use of end-to-end congestion control is desirable for robustness, scalability, and implementation flexibility at network elements. However this paper argues that congestion

[1] The Internet architecture is embodied in the TCP/IP protocol suite and best described in [3].
[2] Here the IP layer will mean the TCP/IP protocol suite.
[3] A natural timescale for this is the link propagation delay.

control could be improved using a variant of the current explicit congestion notification (ECN) standard [17] which uses a codepoint in the packet header rather than packet drops to signal congestion to end-systems. The proposed framework provides flexible tools to express scarce resource allocation policies and moderates the undesirable side-effects that can affect packets traversing contended resources in today's Internet.

The Internet's current delivery model for packets traversing contended resources[4] is characterized by variable inter-packet jitter and non-negligible loss probability per packet. The bulk of deployed applications can tolerate this but a range of applications cannot and have their deployment hindered. To date the proposed mechanisms for resource allocation in the Internet that offer a low-loss and low-delay packet delivery service[5] have been unable to provide a desirable mix of simplicity, performance, flexibility, and total cost of ownership. This paper advocates the adoption of a low-loss and low-delay service model for *all* packets through a new congestion control framework; this change would enable a reliable deployment of some new applications. Some argue that such a framework is unnecessary since good network provisioning can resolve these issues; it still remains unlikely in a large scale heterogeneous network that there will not be resource scarcity somewhere. This scarcity could arise from the sudden deployment of an innovative new application, changing application usage, rapid changes in demand causing hot-spots, equipment failures that cause re-routing, provisioning miscalculations, or incentives structures that lead to under-provisioned links.

Given the difficulty and disruption associated with changing such a successful protocol suite, it is sensible to ask: Why bother? This paper takes the position that the changes yield benefits far in excess of the costs imposed. In order to argue this point we outline a low-loss and low-delay framework for the Internet in Section 2. We then consider how the network might evolve desirably above, below, and within the IP layer. Section 3 looks at how the deployment of networking technologies such as optical networks, wireless, and future high speed networks might fit into the framework described. The evolution of application usage of such a network is addressed in Section 4. A discussion of changes that might occur within the IP layer relating to behavior incentives, routing, provisioning, traffic and organization are considered in Section 5. Finally a conclusion is given in Section 6.

2 A new congestion control framework

A proposed standard [17] for the addition of ECN to IP leaves the TCP congestion control algorithm unchanged and provides minimal guidance for congestion detection. It is argued here that this simple signaling mechanism could yield more dramatic improvements with careful changes to the flow control and congestion detection algorithms.

It has been suggested [8] that an end-system controlled network with a low-loss and low-delay service for all packets could be constructed using ECN signaling. This suggestion is motivated by recent increases in link bandwidth and the success of end-system based congestion control; increased link bandwidth reduces the queuing delay required for transient contention and tight end-system based flow control could dramatically reduce the loss rates and queues caused by congestion. The scheme would require elastic sources to respond quickly to changing conditions but not so quickly that they introduce oscillatory behavior that might lead to queuing delays and loss. End-system based connection acceptance control methods could be used to introduce elastic behavior to the class of inelastic flows. Both types of control have now received research attention. There has been work on the control theory of elastic flow control in heterogeneous large scale networks [23, 15] with several protocols developed and evaluated [11, 16, 12]. End-system based connection acceptance control has been studied mathematically [7] with an associated protocol developed [10]. These control methods are motivated by solutions to the decentralized resource optimization problems found in large scale networks [9, 14]. Thus a coherent framework for

[4] Resource contention or scarcity will be taken to mean congestion from here.

[5] Low-loss and low-delay will mean less than 0.1% packet loss and queuing delays with 99th percentile no more than 10 packets for a packet traversing a contended resource.

analyzing and implementing resource allocation policy is provided. The proposed congestion control framework consists of the changes to the ECN standards which provide mechanism and the resource allocation models that drive policy.

2.1 Changes to the ECN standard

Changes to the ECN standard [17] are needed to allow the Internet to support a low-loss and low-delay packet service model. It is desirable for there to be some decoupling between the congestion detection algorithms used at routers and the flow control algorithms used at end-systems. This decoupling allows the framework to be more easily implemented on a variety of different network and end-system architectures. It also gives room for the congestion detection algorithms to evolve without needing changes to the flow control placed at end-systems and vice-versa. It might appear that in order to provide a scalable and stable elastic rate control providing a low-loss and low-delay packet service would require a tight coupling between the detection and response algorithms. Theory suggests that this need not be the case [23]. It is beyond the scope of this document to cover all the protocols developed. Instead an overview will be given for a congestion detection algorithm, a scalable and stable TCP variant for elastic flows, and a probe based connection acceptance control. The operating limits of these protocols will then be considered.

2.1.1 A congestion detection algorithm

The following static virtual queue based marking scheme [11] is an example of a simple and easily implementable detection algorithm. Suppose a link has a capacity C bps, a FIFO buffer of size B bits and a virtual queue which is drained continuously at rate θC. When a packet arrives to be forwarded on the link it is marked with probability

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

where b is the current size of the virtual queue in bytes, s is the predominant data packet size and ϕ sets the marking scheme's *effective buffer size*, $\frac{1}{\phi}$. Experiments [11] have shown that such a static scheme is effective at maintaining the low-loss and low-delay service model but is not able to provide high utilizations in all scenarios. Timescale decomposition approaches to ECN marking [13] could enable higher utilizations. However it is necessary to run links at utilizations below 100% if the low-loss and low-delay service model is to be maintained.

2.1.2 A scalable and stable TCP variant

A scalable and stable TCP variant [11] designed from a control theoretic standpoint could provide a protocol for elastic flows in the framework. A sender using this variant increases its send window, *cwnd*, by a constant, a, when receiving an acknowledgment without congestion indication and decreases its window by *b.cwnd* for each acknowledgment with congestion indication. The local stability criteria for such a flow control [23] allows a network with stable link arrival rates and fast convergence to be constructed. If most bytes are transferred in long flows then the control could maintain a low-loss and low-delay packet model. Furthermore TCP is currently unable to provide good performance over high bandwidth links with large round trip times. This is because TCP increases its window by one packet each round trip time while halving it in response to congestion. In contrast the variant increases its window by a for *each* unmarked packet and so has a rate of increase per round trip time which scales with sending rate; this gives better performance in scenarios where there is a high bandwidth delay product available. There are other differences in this flow control; fractional window operation, round trip time invariance, and weighted resource sharing. Lower round trip times can result from: increased link capacities that reduce serialization times, increased processor speeds that reduce delays in protocol stacks, and caching which pushes content closer to the end-system. This round trip time reduction might lead to smaller bandwidth-delay products which would cause TCP fractional window problems

to affect an increasing number of bytes transferred and total connections.[6] A protocol using a low-loss and low-delay network does not have a buffer based delay line available exacerbating the problem if it is not tackled directly. The variant implements a rate pacing scheme able to deal with fractional windows without the known drawbacks that naïve rate based flow controls can display.[7] Round trip time invariance is obtained by scaling the parameter b so that there is no bias in rate allocation towards connections with shorter round trip times. Finally the variant's *sharing constant* provides a mechanism to weight a flow's rate allocation when congestion occurs; this could be used for service differentiation.

2.1.3 A connection acceptance control

A probe based connection control has been developed [10] which detects available route capacity and current congestion levels. Its intended use was with rate inelastic applications, such as voice over IP,[8] which have good statistical multiplexing properties. The control sends probe packets at an increasing rate while maintaining an estimate of the current congestion level using feedback from the probe receiver. If the estimated congestion level rises beyond acceptable bounds the call is blocked and the probing terminates. Should the probing reach the desired rate it holds for a small time. If the congestion estimate remains below a decision threshold after holding this rate the call is accepted. Distributed connection acceptance allows classes of non-elastic flows to fit within the same congestion control framework without damaging the service model. The performance of this type of statistical acceptance control is affected by aggregation levels. Theory and simulation suggest this might not be a barrier for its use with voice over IP connections.

2.1.4 Operating limits

Despite careful design of the protocols to operate without damaging the underlying packet delivery model, it cannot cover every operating scenario. Care has been taken to ensure graceful degradation and robustness in those scenarios where the strict service model is violated. The scenarios in which this framework has difficulty maintaining the service model are predominantly those with low bandwidth links such as dial up modem access or cellular wireless systems. In many of these cases it might be easier for ad-hoc mechanisms to be applied since the domain of control will cover only a single entity. Pathological traffic patterns, such as extremely bursty single packet transfers which are highly delay insensitive, may also prevent the framework from maintaining the service model.

Implicit in the framework's use of congestion detection and feedback control is that the links must run at below 100% utilization if the service model is to be maintained. How far below 100% utilization a link needs to be run at depends on the traffic patterns seen at that link; simulation suggests that 80-90% might be sufficient. Whether this loss of 100% utilization is significant must be considered in the context of decreasing bandwidth costs and the improved service model.

Promising future extensions in such a framework could be envisioned for reliable multicast, multicast congestion control, and non-congestion based loss found in wireless environments.

2.2 Resource allocation model

Consider the problem of allocating a scarce resource amongst a variety of users to maximize network utility. It is this optimization problem that Internet end-systems attempt to solve through congestion control. However the TCP protocol implicitly defines a rigid utility for each connection,

[6] Fractional window problems occur for TCP when the average congestion window size is less than 1. In this regime back off timers with poor granularity become the only method to pace packets. This can lead to highly variable link arrival rates and unfairness in resource allocation. If ECN is not available the problem is exacerbated because a coarse timeout becomes the only mechanism to detect congestion with windows of three packets or less.

[7] Ethernet interfaces contain a retransmission timer of sufficient precision and for some cards it would be just a careful driver or firmware rewrite. For end-systems otherwise intrinsically unable to pace packets a window based implementation is being developed.

[8] If a fixed rate codec is used after call initiation user perception of call quality is kept high.

with strong dependence on round trip time, that may incorrectly reflect true application utility. This suggests that the Internet's solution to the resource allocation problem may not even be approximately optimal. It remains appealing to implement congestion control at end-systems for robustness and scalability reasons [3] but also because an end-system may be best placed to express its utility.

Theory for tackling the optimization of resource allocation in heterogeneous networks [9, 14] shows how resource allocation will occur given end-system responses to congestion. Hence by designing appropriate congestion controls a variety of allocation policies can be expressed. Furthermore a low-loss and low-delay service model allows the network to support a wider range of applications. The challenge for protocol designers is to design controls that preserve this service model in the presence of random noise, propagation delay lags, dynamic concurrent flow loads, and other implementation constraints. The protocols they are able to design determine the framework's expressiveness for policy. Despite the challenges, several admissible protocols have been designed suggesting that an end-to-end network architecture can support a stringent packet service model and range of resource allocation policies.

Exposing the resource allocation mechanism in a less opaque way might require new incentives for end-systems to reveal their utility correctly. Further discussion on incentives is postponed until Section 5.1.

3 Below: Network technology implications

The current TCP/IP protocols make implicit demands on underlying network technologies in order for there to be good throughput and resource sharing; for example a network technology must provide significant buffering both in space and time, low link bit error rates, in-order packet delivery, and sufficient bandwidth delay products for the self clocking in TCP's window based sending scheme. Optical packet switch designs challenge the wisdom of using buffer based contention dynamics in the resource sharing algorithms of TCP. At the same time existing router designs could become increasingly costly if buffer sizes must scale linearly with link speed in order to ensure good TCP resource sharing. There would be advantages to reducing the implicit buffering requirements for high-speed network equipment if data buffering could be more cheaply performed at end-systems; for example smaller buffering requirements would allow high speed router designs to use simpler buffering schemes with faster but less dense SRAM memory instead of more complex buffering architectures based on slower but denser DRAM memory. Moves towards sharing mechanisms based on ECN and the various scale invariance properties of the TCP variant described would allow less painful evolution to such high-speed networks.

Wireless networks often suffer from non-congestion related losses due to the dynamic properties of the radio channel. The decoupling of packet loss from congestion signaling would allow a richer space in which to design wireless network protocols. The application of ECN techniques to differentiate losses caused by the transmission channel from congestion induced loss in wireless networks remains an interesting line for research.

Benefits arise from the framework described in Section 2 but costs are imposed on network elements with the requirement of congestion detection and ECN signaling. The research to date suggests good performance can be achieved at each link on a router using simple virtual queue based detection schemes that have no per flow state with light storage and computation requirements. Flexibility exists in the design of detection mechanisms provided they obey the theoretical constraints to function in the control theoretic framework. This flexibility allows easy implementation on a wide variety of networking technologies and allows for utilization improvements from more advanced congestion detection methods as they become available; for example timescale decomposition approaches to congestion signaling [13].

4 Above: Application implications

A design goal of the Internet architecture was the ability to deploy a wide range of applications on top of a ubiquitous network infrastructure. Unfortunately the properties of the datagram and flow service available in today's IP networks are often inappropriate for many applications. This is due to the style of best-effort packet delivery imposed by the congestion control mechanisms and the in-flexibilities of TCP; such as the bandwidth allocation bias to connections with short round trip times and fixed short-timescale dynamics in response to congestion.[9]

Consider the network from an application's point of view. The notion of unreliable information delivery is unappealing for many applications and so abstracted away to give a desired level of reliability using forward error correction (FEC) or acknowledgment based retransmission. Let the latency of small messages and the properties of sustained bulk throughput in an acknowledgment based reliable byte-stream be the primary indicator of an application's performance. Small message latency clearly increases with loss and queuing delay. Loss increases latency by at least one-way delay and the latency of the loss detection mechanism. Large queuing delay on a congested route is an implicit part of the current control mechanisms and compounds any latency effects caused by packet loss detection. Sustained throughput performance is predominantly determined by the result of a resource allocation decision on a route over a longer timescale. Stalls in a sustained byte-stream can be caused by retransmissions of lost segments and may be detrimental on finer grain timescales. The level and expression of performance degradation is clearly dependent on the application.

Deployment of the protocols in Section 2 would reduce the loss and queuing delay based degradations caused by the current congestion control framework. It also provides a mechanism to express allocation policy for sustained throughput requirements. The following discussion will look at a variety of applications which could find new deployment opportunities, reduced implementation complexity, improved performance, or increased robustness with the new framework.

4.1 Distributed systems programming

Distributed systems programming can be hard; implementation complexity arises from the presence of concurrency, variety of failure modes, heterogeneity of node types, and variable network performance. A low-loss and low-delay network is not a magic bullet for the programmers of these systems but it has the possibility to improve the performance and reliability of distributed systems. Highly variable network latency and unreliable delivery makes performance prediction in distributed systems much harder. Many application programmers must make network timing assumptions based on small message latency or bulk throughput for the system to perform acceptably and sometimes for the system to behave correctly. Reducing the variation in small message latency and the ability to express throughput preference could allow many systems to degrade more gracefully. Examples of distributed systems that could benefit from a low-loss and low-delay network are routing algorithms, searches in peer-to-peer systems, or the real-time event distribution systems found in some online games.

Consider distributed systems based around the remote procedure call abstraction; for example database applications or distributed simulation. A large variation in the invocation latency introduced by packet loss and queuing delay results in significant unpredictability in the completion times of calls to remote objects; this impacts system performance dramatically. Programmers are often forced to spend significant development effort re-structuring their programs to address performance issues introduced by highly variable invocation latency. Failure to address such issues can reduce development cost but can make the application brittle to network conditions often requiring an increase in network infrastructure cost to gain reliability. Such applications become unable to operate in conditions where the network is shared with other applications that may contend for resources.

[9] TCP-friendly approaches [5] address the problem of dynamics but have left unresolved resource allocation dilemmas and the side effects of the current congestion control mechanisms.

Take distributed routing algorithms as an example of an application that might wish to express its importance. It seems natural that during congested periods, traffic for routing algorithms should receive a much larger share of scarce bandwidth to avoid cascading network failures. By increasing the resource weight associated to BGP flows network operators could express this policy explicitly without relying on implicit mechanisms such as TCP's round trip time rate bias which can be error prone.

Networked file-systems within operating systems construct an illusion of networked storage behaving as if it were local. For the illusion to be complete request latency for meta-data and small files must be kept low while large file throughput should be high. This requires a network connecting clients to servers which can offer low-loss and low-delay for small requests and the expression of high throughput allocation for large file transfers. It might be possible to lower the level of provisioning needed to support networked file-systems in the low-loss and low-delay IP network advocated. Graceful service reduction would then occur during busy periods while lower priority greedy traffic could coexist alongside the file-system traffic. These improvements might yield cost savings through the ability to exploit scale across a wider area, reduced network infrastructure requirements, or reduced system brittleness.

4.2 Real-time applications

Real-time interactive media applications, such as voice over IP or action games, have stringent delay bounds to provide a good user experience. A low-loss and low-delay network explicitly supports such applications provided their throughput requirements can be met. The issues for these applications then become sharing policy and connection acceptance control.[10] Additional network goodput gains would result from a lower level of FEC redundancy used in applications to protect against packet loss. Other interesting examples are interactive applications that require relatively modest throughput of a reliable byte-stream but low latency. The remote user interface interaction enabled by the Virtual Network Computing (VNC) application [18] is one example. This application is provocative because of the IT system structures it could enable. Imagine a computing system where all sessions are persistent and lie on multiplexed clusters operated from a single regional location with corresponding economies of scale. Significant cost savings might arise from scale in maintenance, support, and the statistical multiplexing of computer hardware and transmission bandwidth. New business models for IT outsourcing could be possible in such a scenario; for example a network, simple screens, input devices and printers might be the only customer side infrastructure needed. It is not argued such systems would be desirable or certain to arise. However their deployment in the current Internet is hindered even in scenarios where these IT systems could meet requirements.

5 Within: Implications for IP network operations and structure

The possible evolution in the structures which bind the heterogeneous networks and end-systems together will be divided into behavior incentives, routing, provisioning, traffic patterns and network structure.

5.1 Behavior incentives

So far it has been assumed that end-systems co-operate and adopt a usage of available protocols and weightings that expresses their true utility to received bandwidth. Dropping this assumption and introducing end-system self-interest leads to a more realistic situation. The components that affect an end-system's resource allocation behavior are embedded in protocol stacks, an application's use of protocols, and an end-user's use of applications. Protocol stack conformance

[10] This assumes that continuous rate adaption is undesirable for these applications; it does not preclude it.

to transmission protocol standards is normally determined by the operating system; changes to the TCP window increment and decrement parameters could allow an end-system to capture more resources. Programmers' use of the socket interface can vary the resources allocated to the application; a file transfer application could increase its share of resources by striping the transfer across multiple TCP connections. The use of an application by a user can also affect resource allocations; the concurrent download of the same file from multiple sources can be used to get the file from whichever transfer completes first. It becomes clear that without controls or incentives a collection of end-systems acting in self-interest may lead to a situation which is far from the social optimum with the added potential of network collapse.[11]

5.1.1 Current incentives

Given the large scale decentralized nature of the Internet, how does it function so well? Firstly there is a culture of social responsibility dating back to the DARPA Internet amongst users, application writers and operating system distributors. Operating systems contain TCP stacks with standard congestion control. An alignment of commercial interests has made it beneficial for operating system distributors to follow the Internet standards resisting pressure to differentiate themselves by introducing a "faster" TCP. The situation at the application level is more cloudy. Games and other interactive real-time applications have used UDP to overcome performance difficulties encountered with TCP. Some of these applications using UDP perform no congestion control and this has lead some administrators to employ port based controls. Web browsers quickly adopted parallel TCP connections to improve page download speeds with an implicit resource aggressiveness detrimental to FTP, Telnet, and Gopher users at the time. Application programmers' restraint to not enter into tit-for-tat connection parallelization in new application designs can be explained by the additional complexity it introduces, the social awareness of the developers, and other external constraints. It is unclear this restraint will hold in the current wave of bandwidth hungry peer-to-peer file-sharing applications where a number of applications are competing aggressively to capture users. Some of these applications are already avoiding operator controls through the use of dynamic port allocation to escape rate throttling and port masquerading to pierce firewalls. Striping data across multiple connections would improve throughput and provide a way to differentiate a peer-to-peer program from the rest with clear consequences.[12] Policy over what the users should be allowed to do with networked applications is a sticky issue. The scope of this paper covers resource allocation aspects rather than the issues of expressing regulatory policy. At the user level, some applications provide mechanisms to control resource usage, other applications do not or find it restricts user functionality. Anecdotal evidence suggests that users, without access bandwidth limitations, quickly learn that a large file download can be achieved faster by connecting to multiple sites and letting the sites compete to deliver the file. Structural influences outside an end-system's realm of control can also apply. Communication involves two parties, so it is necessary for senders and receivers to agree on the resource allocation. This can maintain order because senders are often in control of their sending rates but this quickly deteriorates if the interests of communicating parties align.[13] Today the large servers accounting for the bulk of traffic can be regulated by network operators through pricing and social enforcement. Network structure can also provide incentives since the access link is often the bottleneck for many end-systems. An access limited end-system finds it only competes with itself; this leads to an incentive to behave sensibly. Increasing access bandwidth and a proliferation of peer-to-peer applications, where interests involve only the communicating parties, provide new challenges to the current delicate balance.

[11] These same controls and incentives are also connected to network security issues.
[12] One peer-to-peer program is already advertising this as a feature [4].
[13] The problem of a misbehaving TCP receiver is tackled in [19].

5.1.2 New incentives

Consider the situation that could be introduced with the congestion control framework of Section 2. A trust of operating systems to implement the protocol standards will be assumed. This provides applications with a set of communication primitives with predictable resource implications that are known to improve the packet service model. There are then two choices for the protocol standards: should a flow be allowed to express resource allocation weightings or should there be one default weighting for every flow? The single weight approach would make the proposal more of an Internet performance upgrade which keeps resource allocation opaque to applications and users. Applications can react by using multiple connections and users can express desire through multiple concurrent downloads. If weights can be set from a finite set then balanced incentives are needed for the operating system, application, or user to choose the weight that best reflects their utility. The differentiated weights approach is more expressive but if incentives are not balanced the highest weight may always be used. This regression to a single weight system might damage the service model if care is not exercised in the choice of the maximal weight. From here the incentive structure under either of the two operating system interfaces will be treated as the single problem of behavior incentives for end-systems to reveal the true social utility of their actions. Throughout it will be assumed that many of the incentives present in the current system remain.

The negotiation of mutual lowest weight allocation for a flow can be enforced by a single end-system. The receiver can use its window advertisements to set the rate it wishes to receive at. The sender is in control of its sending rate and the use of ECN nonces [20] ensure that a receiver cannot deceive the sender of the true congestion level without detection. These two mechanisms provide powerful self regulation if a significant proportion of the end-systems accounting for the bulk of the traffic can be trusted. This suggests that the new congestion control framework would work provided the current incentive structures remain applicable in the future.

Designing incentives for individuals to reveal their true utility for a scarce resource is what economists understand as *mechanism design*. The emphasis in this context is on support for the deployment of heterogeneous incentive designs that interact well with room for innovation and evolution. This is in contrast to imposing one incentive mechanism which will clearly not fit all deployment scenarios or be able to exploit the appropriate social relationships that may exist. It is worth noting that incentives can apply both to an end-system and an aggregate depending on the ownership, social, and economic relationships that groups the aggregate.

The abstraction of *congestion cost* will prove to be powerful. The optimization process described in Section 2 results in shadow prices being determined at each network resource. These shadow prices are conveyed to end-systems through the congestion signals generated by scarce resources. Let the congestion cost of a flow on a route be the number of congested bytes received at the receiver. A congestion cost is measured in congested bytes and not a monetary unit. Congestion costs are easily measured at the receiver or anywhere on the acknowledgment path of the reliable transport protocols. A partial congestion cost can be measured anywhere on the forward path and subtracting an upstream partial cost from a downstream partial cost gives the cost incurred on a route segment. The ease with which congestion cost can be measured from multiple positions makes this abstraction a powerful tool for network management. Congestion cost allows controls to be designed that account and then take action based on who responsibility has been attributed to.[14]

An infrastructure that could be deployed in a range of situations is *congestion token control*. In this scheme, traffic is identified for control and a measurement device deployed. The device continually receives tokens that are scarce and time dependent; they can not be saved indefinitely. The device then measures the congestion costs of the flows it is responsible for and surrenders tokens equivalent to these costs. Should the device have insufficient tokens an exception is raised to a policing agent which carries out policy enforcement. This accounting and policing scheme bears some similarities to those used by ATM, IntServ and DiffServ networks with several crucial differences. The packet service model is a product of end-system responses to simple network

[14]The monitoring of congestion costs for flows and aggregates might provide a natural indicator for denial of service attacks or other abnormal behavior.

signaling; this makes policing a question of catching those not employing congestion control, capturing an inappropriate share of the resources, or any other undesirable resource usage outcome. There is considerable flexibility in the deployment and complexity requirements of such a control; this allows social and economic relationships to be exploited at an appropriate granularity. In this scheme policing is based on congested bytes not the raw number of bytes transferred which is beneficial as it encourages use based on the network's level of contention. Furthermore policing and incentives could evolve independently of the packet service model and individual protocols.

Consider an operating system based implementation of congestion token control.[15] Each user is assigned a policing entity which continually receives new tokens over time. The tokens are scarce and valid for a fixed time; this token process could be easily implemented using a leaky bucket. Flows terminating or originating from a user can then be measured and policed. When the user is unable to surrender sufficient tokens they are punished by constraining the weights of their connections or the possibility of connection dropping and blocking. Administrator based intervention can be raised if the situation is persistent. This policing abstraction can achieve a range of policy goals at various granularities; a user can be defined as a process, session, machine, or some other aggregate, incoming flows could be considered independently of outgoing flows, the rate which tokens are granted to different users could be varied to express priority, the token granting and settlement process could be centrally administered, etc. If the end-systems is untrusted then a proxy can be placed at the access link to achieve the same goal. The administration overhead of such policing depends on the need for it and the way in which it is implemented. Often the threat of effective policing deployment is a sufficient incentive for end-systems to behave. A network level implementation is also possible with aggregates classified as required but the scaling properties and enforcement options would be different.

The congestion token based design could be adapted to implement end-system based *congestion pricing*. In this process the tokens have an explicit money value, the token granting process is the granting of credit, and the settlement process is billing. This explicit linking of congestion to billing is controversial and changes the incentive structures. Free market dynamics enter the system delivering a strong incentive to reflect true utility. This transmits incentives to network operators and might increase competition between operators improving performance in the IP service provision market with benefits to the wider economy. However the stakes are raised; new incentives to try and subvert the system are introduced. This leads to stringent requirements on the control and billing system: operator abuse of end-system bills must be avoided, attributing responsibility for the value of communications between sender and receiver must be much clearer, network security must be tight to avoid disorder, strong mechanism designs or regulation of inter-network peering might be needed to avoid market failures in the provider market, etc. It is not argued that such congestion based pricing systems are impossible to design but care is needed to ensure they deliver benefits in excess of the implementation costs. The M3I project [21] has been addressing such issues with an architecture presented in [2]. Pricing systems operating on longer timescales that use congestion cost and bytes transferred as inputs to a wider billing process appear more tractable in the near term. For example the mobile telephone operators in the UK use a variety of fixed rate, usage based, or hybrid pricing plans to aggressively optimize their cost-revenue tradeoffs within the scarce capacity of their networks.

The emotive issue of charging policy is one that the Internet design community has carefully tried to avoid. It is right to avoid the imposition of a one size fits all charging model which may not reflect the heterogeneity in IP network structures and usage. However it is important to support mechanisms from which flexible charging structures can be constructed. Failure of system architects to address the issue may lead to operators taking the matter into their own hands; a rigid Internet architecture could be the result. The key advantage of the new congestion control framework is that it provides for more flexible construction of balanced incentives to control resource usage than are presently available. These scarce resource control mechanisms provide a system evolution in which the original design philosophies underlying the Internet architecture

[15] This scheme is similar and complementary to the Congestion Manager [1]; while the Congestion Manager itself is similar to the TCP control block interdependence proposal [22]. The temporal congestion information sharing techniques in both schemes could also be used here.

remain intact.

5.2 Routing

Routing between two end-systems in the Internet is essentially static on the timescale of most connections. Previous experience of congestion aware intra-domain routing revealed the need for such protocols to account for the elastic nature of the traffic and the need for routing stability. Internet traffic is often elastic; attempting to reduce congestion on one link by re-routing some flows can result in more of the network becoming congested. Re-routing of flows can lead to transient loss of connectivity while the routing algorithm stabilizes and end-to-end latency changes due to path changes. Routing stability is then desirable in itself and even more so if end-to-end flow controlled protocols are used. The framework presented does not solve routing problems directly but its presence introduces new mechanisms for quantifying link congestion adding some predictability to the level of demand on a link. This might lead to traffic matrices that contain an element measuring the elasticity of demand. More complex questions might then become tractable such as: "what would be the effect of customer A's traffic on customer B's traffic after a routing change?"

At present the impact of traffic from one network on another network during congestion can be opaque. The abstraction of congestion cost provides a mechanism for operators to account for the impact of peer and customer traffic on their network. Such accounting could prove valuable in clarifying peering arrangements, service level agreements, and inter-domain settlement processes. These economic influences might carry through to changes in the way inter-domain routing is conducted from provisioning to BGP timescales.

5.3 Provisioning

The framework in Section 2 could find uses in the provisioning process. For example a highly utilized link might just be one that is not heavily demanded and so not warrant an upgrade. In the new congestion control framework highly demanded links will have a high rate of congestion marking and those with low demand will have low levels of marking. The more transmissive nature of load due to the reduction in drop probability makes it easier to see the consequences of upgrading an upstream bottleneck on downstream links. A link which performs no marking but sees a large amount of its load marked on arrival might be the next bottleneck when the previous bottleneck is removed. Further cost savings might be possible if networks can be built that run robustly at higher levels of utilization. Provisioning still remains key to ensuring that applications receive sufficient throughput to function correctly after the allocation of scarce resources.

5.4 Traffic patterns and network organization

Traffic patterns as observed at many different scales may change. At the microscopic timescale the phenomenon of *acknowledgment compression* leading to bursts in self-clocking window based traffic might be greatly reduced by the low-loss and low-delay packet delivery model. The use of rate basing for some flows either intrinsically, such as in a voice over IP flow, or as embedded in data transmission protocols may also change packet scale behavior. New application deployment from the improved service model could change traffic patterns in many ways. For example voice over IP traffic would be more symmetric than today's Internet flows. Remote user interface interaction applications may make flows more long lived with downstream characteristics dependent on user interface content. Resource greedy applications such as pre-caching could be deployed without fear of using bandwidth when demand is high altering the time of day traffic distributions. One thing is certain; increasing bandwidth and the constant innovation of new applications will result in changing traffic patterns. The framework presented is robust to such changes through its flexibility; it assumes only more increases in bandwidth to come.

The so-called *death of distance* in the telecommunications industry caused by optical transmission led to a dramatic reduction in the cost of long-haul bandwidth and resulted in a major

reorganization in the industry's structure and traffic patterns. Analogously the removal of distance dependent resource allocation advocated in Section 2 could bring about a similar death of distance effect in data networks. For example the benefits of web caches are in reducing network traffic, improving users quality perceptions through reduced request latency, and a preferential network resource allocation. Operators selling a content distribution service are also selling to customers a favorable network resource allocation through geography! A network with resource allocation not dependent on physical distance and latencies closer to true propagation speeds may have a different structure.

6 Conclusion

The primary costs of changing the congestion control framework are:

- A possible reduction in link utilizations to support the improved service model.
- The cost of modifying end-system protocol stacks to support the new protocols.

These changes are compatible with the design goals of the Internet architecture; most significantly flexibility towards applications and networking technologies. The costs are minimal in the context of increasing link bandwidths with a corresponding decrease in the cost per bit and the natural evolution of protocol standards and operating systems. The main benefits of the new framework are:

- A significantly improved packet service model which provides low-loss and low-delay to all packets.
- The ability to make resource allocation decisions explicit.
- The decoupling of resource accounting and policing from the packet service model.

The potential benefits for new innovative applications as described in Section 4 are large. The decoupling of packet service from resource allocation provides a rich space for network administrators to build accounting and policing structures as required. Given the pressure on the architecture to support new applications and provide more assistance for resource allocation, accounting, and policing; it is concluded that these benefits exceed the costs. The bother of changing appears worth it.

7 Acknowledgments

This paper was written while a UCLA IPAM research fellow attending the program on Large Scale Communication Networks with additional funding from the Royal Commission for the Exhibition of 1851. Thanks go to Jon Crowcroft, Frank Kelly and Andy Hopper for useful advice and discussions. Alastair Beresford gets thanks for his proof reading. While Damon Wischik and Richard Mortier deserve credit for many useful and often probing discussions. Thanks also goes to the support of those at the Laboratory for Communication Engineering and AT&T Laboratories, Cambridge.

References

[1] D. Andersen, D. Bansal, D. Curtis, S. Seshan, and H. Balakrishnan. System support for bandwidth management and content adaptation in Internet applications. In *4th Symposium on Operating Systems Design and Implementation*, San Diego, CA, October 2000. USENIX.

[2] B. Briscoe, M. Rizzo, J. Tassel, and K. Damianakis. Lightweight policing and charging for packet networks. In *Proceedings of Third IEEE Conference on Open Architectures and Network Programming*, pages 77–87, Tel Aviv, Israel, March 2000.

[3] D. Clark. The design philosophy of the DARPA Internet protocols. In *ACM SIGCOMM 1988*, pages 106–114, Stanford, CA USA, August 1988.

[4] eDonkey2000 Website. eDonkey2000: Overview. `http://www.edonkey2000.com/overview.html`.

[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equation-based congestion control for unicast applications. In *SIGCOMM 2000*, Stockholm, Sweden, August 2000.

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

[7] F. Kelly, P. Key, and S. Zachary. Distributed admission control. *IEEE Journal on Selected Areas in Communications*, 18(12):2617–2628, December 2000.

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

[9] 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.

[10] T. Kelly. An ECN Probe-Based Connection Acceptance Control. *Computer Communication Review*, 31(3), July 2001.

[11] T. Kelly. On engineering a stable and scalable TCP variant. Technical Report CUED/F-INFENG/TR.434, Laboratory for Communication Engineering, Cambridge University, June 2002.

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

[13] S. Kunniyur and R. Srikant. A time-scale decomposition approach to adaptive ECN marking. In *IEEE INFOCOM 2001*, Anchorage, Alaska, April 2001.

[14] S. H. Low and D. E. Lapsley. Optimization flow control, I: Basic algorithm and convergence. *IEEE/ACM Transactions on Networking*, 7(6):861–875, December 1999.

[15] F. Paganini, J. Doyle, and S. H. Low. Scalable laws for stable network congestion control. In *IEEE CDC*, Orlando, FL, December 2001.

[16] F. Paganini, S. H. Low, Z. Wang, S. Athuraliya, and J. C. Doyle. A new TCP congestion control with empty queues and scalable stability. Submitted for publication.

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

[18] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. Virtual network computing. *IEEE Internet Computing*, 2(1):33–38, February 1998.

[19] S. Savage, N. Cardwell, D. Wetherall, and T. Anderson. TCP congestion control with a misbehaving receiver. *Computer Communication Review*, 29(5), October 1999.

[20] N. Spring, D. Wetherall, and D. Ely. Robust ECN signaling with nonces. Technical report, IETF Internet Draft, October 2001. `http://www.ietf.org/internet-drafts/draft-ietf-tsvwg-tcp-nonce-02.txt`.

[21] The M3I Consortium. Market Managed Multiservice Internet. `http://www.m3i.org/`.

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

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

[24] W. Willinger and J. Doyle. Robustness and the Internet: Design and evolution. Pre-print available via `http://netlab.caltech.edu/internet/`.