Technical Report

UCAM-CL-TR-532
ISSN 1476-2986

Number 532

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Internet traffic engineering

Richard Mortier

April 2002

JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

This technical report is based on a dissertation submitted October 2001 by the author for the degree of Doctor of Philosophy to the University of Cambridge, Churchill College.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/TechReports/

Series editor: Markus Kuhn

ISSN 1476-2986

Abstract

Due to the dramatically increasing popularity of the services provided over the public Internet, problems with current mechanisms for control and management of the Internet are becoming apparent. In particular, it is increasingly clear that the Internet and other networks built on the Internet protocol suite do not provide sufficient support for the efficient control and management of traffic, i.e. for *Traffic Engineering*.

This dissertation addresses the problem of traffic engineering in the Internet. It argues that traffic management techniques should be applied at multiple timescales, and not just at data timescales as is currently the case. It presents and evaluates mechanisms for traffic engineering in the Internet at two further timescales: flow admission control and control of per-flow packet marking, enabling control timescale traffic engineering; and support for load based inter-domain routeing in the Internet, enabling management timescale traffic engineering.

This dissertation also discusses suitable policies for the application of the proposed mechanisms. It argues that the proposed mechanisms are able to support a wide range of policies useful to both users and operators. Finally, in a network of the size of the Internet consideration must also be given to the deployment of proposed solutions. Consequently, arguments for and against the deployment of these mechanisms are presented and the conclusion drawn that there are a number of feasible paths toward deployment.

The work presented argues the following: firstly, it is possible to implement mechanisms within the Internet framework that enable traffic engineering to be carried out by operators; secondly, that applying these mechanisms with suitable policies can ease the management problems faced by operators and at the same time improve the efficiency with which the network can be run; thirdly, that these improvements can correspond to increased network performance as viewed by the user; and finally, that not only the resulting deployment but also the deployment process itself are feasible.

Acknowledgements

I welcome the opportunity to thank my supervisor. During the course of this Ph.D., Ian Leslie read and discussed more versions of this dissertation than anyone should have had to; Ian Pratt was the instigator of much debate; and Simon Crosby provided a most enriching internship with Cplane Inc.

I wish to acknowledge the work of Christopher Clark for the initial NS implementation of the DropTailMtk queue used in Section 3.4; the work of Ian Pratt in collecting the data for Table 3.1 describing the back-off behaviour of various TCP stacks and providing the complex traffic model for Chapter 3; and the work of Austin Donnelly in modifying the VIC tool to adapt in response to receive reports for its use as an RTP source in Section 3.5. This Ph.D. was initially funded by an EPSRC CASE award in conjunction with BT, and latterly by Marconi Research, Cambridge.

Thanks are due to all who suffered to read drafts of this dissertation, but especially to Steve Hand, Rebecca Isaacs, Ian Leslie, and Ian Pratt. Any remaining errors are mine alone.

Finally, I also wish to thank all those past and present members of the SRG who made this Ph.D. such an educational, rewarding, but above all alcoholic, experience. In particular: Paul Barham, Herbert Bos, Austin Donnelly, Steve Hand, Tim Harris, Rebecca Isaacs, Paul Jardetzky, Derek McAuley, Andrew Moore, Ian Pratt, Sean Rooney, Dave Stewart, and Neil Stratford. Cheers.

Chapter 1

Introduction

Data networks exist to transport information at the behest of users. Users receive value from the network based on the various properties of this transport, such as latency, throughput, and reliability. Network providers operate networks to provide value to users by carrying data under user-specified constraints. The process of managing the allocation of network resources to carry traffic subject to constraints is known as *Traffic Engineering*.

The Internet currently provides for traffic engineering only at data timescales, with poor support for expression of policy. The thesis of this dissertation is that mechanisms for traffic engineering in the Internet are required at multiple timescales, and furthermore, pricing is a useful mechanism through which to express traffic engineering policies as it is both intuitive and flexible. This chapter motivates this thesis, and then summarises the contributions and outlines the remainder of the dissertation.

1.1 Traffic engineering

Traffic engineering is '*concerned with the performance optimization of networks*' [Xiao00]. It addresses the problem of efficiently allocating resource in the network so that user constraints are met and operator benefit is maximized. It can be performed automatically or through manual intervention, and is required at a variety of timescales discussed below.

One might consider that current technology trends remove the need for traffic engineering. Advances in optical networking are making ever-increasing amounts of bandwidth available, effectively reducing the marginal cost[1] of

[1] *Marginal cost* is defined as the increase in cost of production resulting from a small increase in output.

bandwidth to zero. The widespread deployment of such technologies is accelerating, and companies are able to sell high-bandwidth, trans-national and international connectivity simply by massive over-provisioning of their networks.

Notwithstanding such developments, traffic engineering remains important and efficient mechanisms for performing it are therefore valuable. There are a number of reasons it retains importance, perhaps principally that both the number of users and their expectations are exponentially increasing in parallel to the exponential increase in available bandwidth. In addition, the bandwidth available to users at the edges of the network is undergoing dramatic increase with the deployment of technologies such as xDSL, Fibre-to-the-Curb, and Fibre-to-the-Home.

Coupled with these increases in user numbers, expectation, and access bandwidth, it remains the case that companies that have invested in such over-provision of bandwidth need to recoup sunk costs. Service-differentiated pricing and usage-proportional charging are widely accepted mechanisms for doing so. Simple and cost-effective mechanisms for monitoring usage and ensuring that customers receive what they request are required to make usage-proportional charging practical.

Consequently, traffic engineering still performs a useful function for network operators and customers. Enabling it to be performed in an efficient and consistent manner is valuable.

A note on terminology

Throughout this dissertation, *packet* is used to mean the smallest unit of data considered by the network, whether a frame as in Ethernet, or an Internet datagram as in IP (INTERNET PROTOCOL). A network such as the Internet consists of *links* that connect pairs of *nodes*. If an interior node is capable of routeing IP packets, it is known as a *router*. A collection of routers controlled by a single administrative entity forms an AS (AUTONOMOUS SYSTEM). ASs which carry traffic for other ASs are known as *transit* ASs; those which don't are called *stub* ASs.

An *internet* is a network that runs IP; the (public) *Internet* is a particular instance of an internet, formed by the interconnection of many ASs. A *multi-service* network is a network such as the Internet which attempts to offer multiple services directly over the same transport technology. This contrasts with older networks such as the telephone network, which have typically offered only a single basic service to users.

When discussing pricing the following terms will be used:

Cost refers to the value expended in providing the service; in a sense the 'manufacturing cost.' This can include both the *marginal cost* of forwarding a particular packet, the *sunk cost* associated with the installation of fibres, and the costs of maintaining and managing an installed network.

Pricing is the process of associating a (potentially arbitrary) number with some service. In most real systems this will have some relation to the total real cost of the service and not, in fact, be arbitrary. Note that the consumer need not see the price, but will instead be charged some amount based on the price.

Charging is the mechanism by which the price is expressed to the consumer. The charge can be viewed as a function of the price and the consumer involved, and perhaps other parameters. This separation of pricing and charging gives greater flexibility for the operator to express policies such as time-, customer- or service-specific discounts, whilst keeping pricing mechanisms consistent and simple.

Billing is the mechanism by which charges are recovered from the customer. It is possible for this to be applied either pre- or post-consumption.

Although all these issues are connected, the research described in this dissertation concentrates on the application of pricing and charging to traffic engineering in the Internet, rather than on billing. Given the required information, billing is a problem associated with the support services provided to the operator by the network, and by the operator to the customer. The work described in this dissertation should have a positive impact on the problem of cost-effective billing, but this is not an explicit aim.

1.2 Timescales

A common scheme, and that followed in this dissertation, is to classify network resource control into three timescales: data, control, and management [Hui88].

Data timescales are considered to be of the order of packet forwarding times. They concern the behaviour and effect of individual packets or packet-trains within the network. At these timescales resource control is usually concerned with controlling transient overload in the network, and with buffer management in the network switching elements and endpoints. For example, TCP (TRANSMISSION CONTROL PROTOCOL) provides both flow and congestion control in terms of segments transmitted. Dealing with resource control at these timescales is not part of

the main contribution of this dissertation; it is however relevant and thus discussed in Chapter 2.

Control timescales concern flows, where a flow is a collection of closely-spaced packets travelling between two defined end-points. 'Closely-spaced' is not a well-defined term, but can be taken in the Internet to mean that inter-packet gaps are of the order of less than a few seconds. Flows are not only generated through open-loop control, such as single file transfers, but also as related collections of transfers, such as involved in downloading an entire web-page when using HTTP/1.0. At these timescales, traffic engineering techniques include CAC (CONNECTION ADMISSION CONTROL), and per-flow signalling and resource reservation. For example, RSVP (RESOURCE RESERVATION PROTOCOL) [RFC2205] creates paths with resource guarantees through the network.

Management timescales concern large aggregates of traffic as might be routed between ASs in the Internet. The protocols that act on these scales are routeing protocols, such as BGP (BORDER GATEWAY PROTOCOL) and OSPF (OPEN SHORTEST PATH FIRST), operating at timescales of the order of minutes or hours. In addition, they include the longer-term deployment and provisioning decisions of the network operators at timescales of the order of days, weeks and longer.

All three timescales must be addressed since they all affect the service perceived by users and the ease and efficiency with which the network can be operated. Each has a space analogue in terms of the level of aggregation – 'data' deals with packets; 'control' deals with aggregates of packets, i.e. flows; and 'management' deals with aggregates of flows.

1.3 Resource allocation and offered services

Prior approaches to the general problem of resource allocation in networks can be classified as either *service oriented* or *technology oriented*.

Service oriented approaches are typified by the offerings of telephone companies, cable providers, and ISPs (INTERNET SERVICE PROVIDERS). Providers offer a pre-defined tightly-specified range of services, to avoid complexity of management and the associated costs. However, this can lead users to consider the provider's offerings inflexible.

Conversely, if the provider offers a very wide range of such services to avoid appearing inflexible, users are likely to perceive this as overly complex; such an approach can also lead to high management costs for the provider. Conforming users will generally have to choose the 'closest match' service rather

than simply using the network as they desire. This results in incentives for users to be non-conforming, increasing policing problems for the service provider.

Technology oriented approaches typify the 'one bit fits all' paradigm, and assume that all users of a particular technology place the same value in the use of that technology. They can be further classified as *user approaches* or *network approaches*.

User approaches rely on the user accurately expressing the requirements of their traffic in some manner dictated by the technology, such as the complex traffic specifications required by the ATM Forum service classes. Experience with such systems suggests that most users are unable to specify their requirements to the required degree of accuracy, as they do not fully understand the characteristics of their traffic. *Network approaches* attempt to treat all traffic equally, relying on the protocols for fairness of resource allocation. For example, the Internet largely relies on the TCP protocol to perform per-connection resource allocation. By mandating that implementations comply with the standard, and furthermore, by recommending that newly developed Internet transport protocols implement TCP friendly congestion control, the network explicitly aims to give no flow preference over another.

Technology oriented approaches have a number of flaws. Specifically, different users typically place different valuations on transfer of data, giving many users an incentive to misuse the protocols. Ensuring that users do not do so is difficult. More generally, it is very hard to design and implement such protocols in a robust and efficient manner – even if there is no malicious intent, one implementation can give a user an unfair advantage over another. In the Internet, these protocols also give the user very little control over the service that they receive, leading to users having very little incentive *not* to use non-conformant implementations to their advantage.

The IETF's on-going DIFFSERV (DIFFERENTIATED SERVICES) effort does provide simple mechanisms to enable users to express different service requirements for their traffic. However, it does not provide any firm guarantees as to the service traffic will receive, concentrating instead on allowing simple differentiation between the service individual packets receive. Furthermore, it does not address the problems of network interconnection and how to translate between service classes at network borders, and how DIFFSERV should be used to build end-to-end services.

Even hybrids of the above approaches are not sufficient to perform resource allocation satisfactorily in multi-service networks. For example, the phone network could be considered a mixture of the service approach and the network approach. Its charging infrastructure has allowed operators to control the load on the network and to inform the provisioning of the network suc-

cessfully in the past. However, the phone network has offered the same basic service, largely unchanged for over 50 years, allowing detailed statistics to be built up about traffic patterns.

The Internet does not follow the same traffic patterns, and there is increasing evidence that the growth of the Internet is even changing traffic patterns in the phone network. These effects are leading to severe problems for operators still using old models and it may even by the case that Internet traffic is not susceptible to such techniques [Leland93, Leland94]. In addition, new services are introduced at a much higher rate and with much less operator control. These problems seem unlikely to be solvable by accruing statistics and building traffic models.

1.4 Contribution

This dissertation contends that existing approaches to traffic engineering in the Internet are not sufficient for multi-service networks, as the Internet is increasingly becoming. Although the Internet provides end-to-end connectivity, it guarantees nothing more as soon as administrative boundaries are crossed. Provision for QOS (QUALITY OF SERVICE) in the Internet is currently dependent on two mechanisms: existing data timescale approaches to resource allocation which are typically restricted to TCP-friendly congestion control mechanisms; and SLA (SERVICE LEVEL AGREEMENT) negotiation between network operators, which typically occurs very slowly.

A more flexible approach is required to address these two problems. Firstly, traffic engineering should be considered from multiple timescales: data, control, and management. Secondly, a mechanism to allow the requirements of many different traffic types to be specified and compared is required. Finally, such a mechanism should give users incentives to behave 'truthfully' in specifying their constraints to the network to help reduce management and policing costs.

There are two principal contributions of this dissertation. Firstly, it proposes and evaluates mechanisms for control timescale traffic engineering in the form of admission control for TCP and an ECN (EXPLICIT CONGESTION NOTIFICATION) proxy for RTP (REAL-TIME TRANSPORT PROTOCOL). These allow the operator to control contention for network resources on a per-flow basis enabling them to offer different levels of service at a more useful granularity than per-packet differentiation.

Secondly, management timescale traffic engineering is discussed, leading to proposal and evaluation of a *price path attribute* for BGP. This enables operators to advertise prices for transit of traffic, allowing them greater control

over the flow of traffic between ASs, as well as more automated SLA settlement, reducing network management costs.

Additionally, this dissertation discusses possible consequences of deployment of these proposals, and the effects their deployment might have on the network. In general, such mechanisms should allow information about both the users' value judgements and the network's state to flow freely, and to be used by many layers of the protocol stack.

1.5 Outline

The detailed structure of this dissertation is as follows. Chapter 2 describes background and related work, covering the basic Internet protocols, approaches to resource control in networks and the Internet in particular, and pricing approaches to network resource control. Additionally, the assumptions made about the structure of the network for the work described in this dissertation are detailed. Chapter 3 discusses the application of control timescale traffic engineering to the Internet in the form of flow admission control for TCP and an RTP-ECN-proxy. It evaluates these mechanisms and also discusses the use of pricing to implement flow admission policies.

Management timescales are considered in Chapter 4, and the application of pricing to inter-AS traffic engineering is discussed. Results from a prototype implementation of a price path attribute for BGP are presented. Deployment and integration issues and consequences are considered in Chapter 5, and finally conclusions are drawn and further work suggested in Chapter 6.

Chapter 2

Background

This chapter provides general background to the work presented in this dissertation. It briefly introduces the principal protocols and technologies referenced throughout this dissertation, in addition to discussing prior approaches to resource control and pricing in networks. It also notes underlying assumptions about the structure of the network.

2.1 Internet protocols

This section describes the basic Internet protocols relevant to the rest of the work in this dissertation. As alluded to in Section 1.1 the Internet is very loosely structured, functioning as an ad hoc collection of ASs, providing connectivity between networks and thus users. This looseness of structure is often considered in large part responsible for the success of the Internet and its associated technologies. By concentrating on *connectivity*, and by making few assumptions about the services to be run, there is a great deal of flexibility in development and deployment of new services. At the same time, the basic service of enabling communication between all connected users remains well-supported, as demonstrated by the continuing increase in popularity of services such as email [Odlyzko00].

2.1.1 Internet Protocol

IP (INTERNET PROTOCOL) [RFC791] is a network layer protocol providing a lowest common denominator for network interconnection. It makes very few assumptions about the underlying network, and so provides only a basic unreliable datagram delivery service with no ordering guarantees between

individual hosts[1]. Due to its simplicity and the few assumptions it makes about the underlying network layers, IP can be implemented over almost any network layer and has therefore been extensively deployed.

An IP packet consists of a header and payload. The header contains the source and destination addresses, and the TOS (TYPE OF SERVICE) for the packet, along with other information relevant to the transport of the packet. In 'classical' IP, a packet traverses the network based solely on its destination address. A decision is taken at each router as to where the packet should next be sent based on the destination address contained in the packet header and the current contents of the router's routeing tables. These tables may be maintained manually, or by a separate *routeing protocol* as discussed in Section 2.4. Packet delivery is *best effort*, with the TOS byte providing only a hint to the router concerning the packet's desired treatment.

2.1.2 Explicit Congestion Notification

ECN (EXPLICIT CONGESTION NOTIFICATION) [Floyd94,RFC2481] is a proposed extension to IP and TCP. It allows the network to provide the user with extra feedback concerning its congestion state using two bits in the IP TOS byte. These indicate firstly whether or not the originator of the packet is ECN aware, and secondly whether or not this packet has experienced congestion. A variety of router marking strategies can be implemented to signal the onset of congestion to end-points using these bits, discussed briefly in Section 2.2.1. They allow higher layer protocols, currently TCP and RTP in particular, to make decisions about their use of network resources without having to experience loss, previously the only available feedback signal.

There are a variety of proposals concerning the treatment of ECN marks, generally falling into one of two categories. The first category is similar to the 'technology oriented' approaches discussed in Section 1.3, and mandates that the protocol should have some standard behaviour in the face of ECN marks, perhaps even behaviour identical to that in the face of loss [RFC2481]. The second category is 'price driven,' where the network counts the marks it sees, and presents the user with a bill for the marks they receive [Key99a, Gibbens99b, Kelly00]. This category is discussed further in Section 2.2.4.

2.1.3 User Datagram Protocol

UDP (USER DATAGRAM PROTOCOL) [RFC768] adds two features to IP: a data checksum so that the receiver may verify that data has been correctly

[1] Rather, between individual IP addresses as a host may support many IP addresses.

received, and port numbers to enable the receiver to distinguish between multiple sources or destinations at the same IP address. This allows the transmitter's operating system to multiplex the traffic of multiple concurrently executing processes into the network, and the receiver's operating system correspondingly to demultiplex this traffic. UDP is commonly used where the reliable ordered byte stream nature of TCP is inappropriate, such as when real-time data is being transported, or where the latency of the handshake process in TCP is too great for the application in question.

2.1.4 Transmission Control Protocol

TCP (TRANSMISSION CONTROL PROTOCOL) [RFC793] is designed to provide a connection oriented ordered reliable byte stream on top of the connectionless unreliable IP. It implements window-based flow control and allows multiple processes at a single IP address to communicate concurrently with a process or processes at other IP addresses. It uses a three-way handshake to establish a connection, and data is then transferred and acknowledged in terms of *segments* measured in octets. This enables retransmission of missing segments when notified of loss by reception of acknowledgements for previously acknowledged data, or by expiry of a timer. The timer values are adjusted as each end-point of the connection estimates the RTT (ROUND TRIP TIME) of the connection.

Modern TCP implementations also support a variety of *congestion control* mechanisms [RFC2914]. Congestion occurs due to contention for limited network resources, typically buffer space or transmission bandwidth. If it is not detected and prevented, then *congestion collapse* may occur; this is where the network, or some subset of the network, is loaded to such a level that *goodput* – the throughput of data, disregarding retransmissions – falls to negligible levels [Jacobson88, Morris99]. Following the rapid increase in the use of TCP and enormous changes in the topology and size of the Internet, a succession of congestion control mechanisms have been proposed and implemented for TCP.

These began with the 'Slow Start' and 'Congestion Avoidance' schemes [Jacobson88], and include the TCP varieties known as Tahoe, Reno, and SACK (SELECTIVE ACKNOWLEDGEMENT) [Fall96]. In general, they are based on the idea that when the TCP sender notices congestion, it will multiplicatively decrease its transmission rate or *back-off*. Congestion is traditionally detected through loss[2] of a packet, but alternatives where packets are marked to signal the onset of congestion are under consideration [Floyd94, RFC2481,

[2]Throughout this dissertation it is noted that *loss* of a packet might not simply be the packet being dropped in the network either through congestion or error at a lower layer, but also its excessive delay, detected through timeout.

Laevens00], as was discussed in Section 2.1.2. The connection will subsequently linearly increase its transmission rate until another loss event is detected. This gives TCP its characteristic sawtooth transmission pattern, as it probes for bandwidth, experiences loss, backs-off, and repeats the cycle.

A more recent attempt to improve the congestion behaviour of TCP resulted in Vegas [Ahn95, Brakmo95, Low01]. Rather than continually probing the network to see if it may increase its congestion window when it reaches steady state, it attempts to estimate the correct congestion window size. This is done by accurate estimation of the RTT using the assumption that the lowest RTT is the RTT that the network would allow if it was not carrying the traffic associated with this connection. This allows the protocol to estimate the amount of data it has in flight, and then to adjust its transmission rate (and RTT estimate) to ensure that its estimated fair share is not exceeded. This behaviour only applies in the congestion avoidance, or steady-state, phase; when loss is detected, the standard TCP congestion control mechanisms are applied. In Chapter 3 it will be shown that there are situations in which neither the behaviour of standard TCPs nor that of Vegas TCP is sufficient to guarantee acceptable performance of the network.

2.1.5 Real-time Transport Protocol

Although TCP is a satisfactory protocol for the transfer of elastic data across the Internet, its reliable byte stream nature makes it unsuitable for the transfer of real-time 'streaming' media. This type of data transfer commonly uses RTP (REAL-TIME TRANSPORT PROTOCOL) [RFC1889], a protocol defined for the purpose of providing support for real-time media conferencing over IP, and usually implemented over UDP. It consists of two parts: RTP itself, supporting the media stream, and RTCP (REAL TIME CONTROL PROTOCOL) supporting transfer of meta-information about the RTP stream. A third related protocol exists, RTSP (REAL TIME STREAMING PROTOCOL), whose purpose is to control the RTP/RTCP streams, providing setup, port negotiation, teardown, and conference subscription and unsubscription facilities.

The RTCP protocol principally uses SRs (SENDER REPORTS) and RRs (RECEIVER REPORTS) to provide feedback between the parties in a conference. These contain protocol meta-data and per-conference report blocks. In addition, the SRs contain extra information about the transmission of data, and are only sent by participants that have transmitted traffic. This allows (multiple) receivers to synchronise with an RTP stream. The report block structure contains information pertaining to the quality of the data stream. Based on the information contained within the SRs and RRs, the receiver can decide where it should be in the stream, and the transmitter can tailor the bandwidth of the transmitted stream to the prevailing network conditions.

2.1.6 Discussion

The Internet is a worldwide network supporting an enormous number of users and services, from simple data transfer to more demanding soft real time multimedia applications. In large part the success of the Internet has been attributed to the simplicity of the service provided by IP, and the flexibility that this allows [Odlyzko00].

However, this flexibility does come at a price – precisely because IP is so simple, it provides very little support for more demanding applications. For example, congestion control had to be implemented in TCP after serious problems with congestion in the Internet arose [Jacobson88], and the schemes implemented could not rely on support from IP. This led to congestion control schemes reliant on packet *loss* as the congestion signal, and gave rise to schemes which tend to react suddenly and harshly to congestion. Similarly, RTCP transmitters have to rely on loss information from the RTP stream being returned to them in RRs. Moreover, the TOS byte and DIFFSERV notwithstanding, classical IP traditionally provides little support for differentiated forwarding treatment of packets.

ECN attempts to retro-fit support for congestion avoidance to IP by enabling packets to be marked as having been in the network at a time when routers were becoming overloaded. This enables ECN-aware applications to behave more intelligently, as they can now choose to react to congestion before it becomes serious rather than relying solely on packet loss to signal that the network is busy. As well as potentially allowing a smoother reaction to the onset of congestion, it also allows applications to better hide the onset of congestion from users. For example, RTP applications can still display the information contained in marked packets, whilst noting that the codec should perhaps alter its behaviour in the face of oncoming congestion.

Those ECN-aware protocols that mandate specific behaviour in the face of received marks still restrict user responses in the face of congestion. They do not allow different users to express their differing valuations of the traffic they transmit – all marked packets are treated identically, as mandated by the protocol specification. Additionally, both the newer ECN-aware protocols and earlier protocols running over IP rely on the end-system to ensure that users receive only their fair share of network resources. The network has few mechanisms for enforcing such behaviour, and users have little or no incentive to conform.

More recent approaches within this paradigm have exposed the congestion information as a price to be charged to users. This allows users to make the decision as to whether they should continue using the network based on the current price of the resource. The price of the resource is calculated

based on the mark probability, and users are effectively charged according to the marks they cause to be generated. This creates an incentive for users to behave fairly.

The rationale is that every marked packet that a user receives contributed to causing congestion, otherwise it would not have been marked. Since the user receives benefit by receiving that packet, they should pay for it. When there is no congestion in the network, no packets will be marked, and so no-one pays any per-packet fee; this assumes that the marginal cost to the network operator of carrying packets is zero. It also assumes that the utility to a user of a marked packet is identical at a given instant, for all marked packets and all users. Hence, even though the actual price charged may change in time with network conditions, and the marking policy may vary from router to router, neither the user nor the network has any way of explicitly differentiating *to the network* between two packets emitted at the same time to the same destination.

Support for this type of differentiation is the subject of the next sections, which discusses network resource control in general, and then specifically in the Internet.

2.2 Network resource control

A computer network comprises many resources; this dissertation principally considers link bandwidth as this is the most important for a wide range of applications, and the most controllable from the operators' point of view. This section introduces the basic mechanisms for controlling bandwidth allocation.

2.2.1 Fair share resource allocation

Network resources can be allocated in a number of ways. Perhaps the simplest from the point of view of the computation required by the network nodes[3] is to embed the resource allocation algorithm in the end-system protocol. Users are then expected to run a conforming instance of the protocol, which will attempt to share the bandwidth fairly between users of the protocol.

Such protocols generally define a fair share at a node as a bandwidth share equal to that achieved by other users. Since flows typically traverse many links in the network, each user will aim to achieve an end-to-end bandwidth

[3] As opposed to the end-systems.

equal to the minimum such share. A flow can enter the network at any time, but should attempt to discern the bandwidth it may use and not exceed this amount. Simultaneously, flows that have already entered the network must ensure that they detect when their fair share allocation has reduced, and reduce their use accordingly.

TCP is perhaps the most widely-known example of a protocol that follows such rules. More recent work has developed schemes whereby traffic transported over UDP can also be made to behave fairly. Such schemes can be generally split into two: equation/model based, and sender/receiver based. *Model based* schemes use mathematical models of TCP to define behaviour that leads to a fair share resource allocation [Padhye99, Floyd00]. *Sender/receiver based* schemes perform rate control as in TCP, utilising some system of acknowledgement for successfully received data [Sisalem98, Rejaie99,Rhee00]. This enables the relevant end-point to implement an additive-increase, multiplicative-decrease rate control in a similar manner to TCP.

In order to make both TCP and UDP based schemes more fair, a variety of router marking disciplines has been investigated. Most use various queue properties to infer which flows are receiving more than their fair share and to penalise them through preferential marking or dropping. Examples include RED [Floyd93], FRED [Lin97], WRED [Bodin00], and SFB [Feng99]. Other schemes use more active methods, such as matching incoming packets against already queued packets to see if two packets are from the same flow [Pan00], or using ICMP (INTERNET CONTROL MESSAGE PROTOCOL) *source quench* messages to allow routers to control transmitters based on router queue occupancy [Rangarajan99].

2.2.2 Admission control

Admission control is the name given to the process of network nodes deciding to grant or deny access to the network for users' traffic. Any admission control function requires knowledge of both the state of the network and the potential impact on existing flows of the admission of another flow before it may decide whether or not a new flow should be admitted. Admission control must be performed by the network, since it cannot generally rely on co-operative behaviour of the sources in competition for the resource[4]. In traditional networks using CAC (CONNECTION ADMISSION CONTROL), the source explicitly signals the network to request access [ATMF-UNI96].

Access to the network is only part of the problem. The network must also ensure that resources are available to carry the accepted traffic. CAC in

[4]Proposals for *distributed admission control* exist and are discussed in Section 2.2.4.

conventional telephony systems is simplified by the fact that connections require unit resource and are established end-to-end. This makes it easy for the network to know if it may accept a connection, since it is of a known, constant bandwidth, with a route determined at connection setup time. Any switch on the route may reject a connection during the connection setup phase. Typical ATM (ASYNCHRONOUS TRANSFER MODE) signalling methods [ATMF-UNI96] use a similar end-to-end system, but require that connections desiring QOS guarantees should declare certain parameters such as the peak and sustained transmission rates in order that resources may be reserved at connection setup [ATMF-TM99].

2.2.3 Measurement based admission control

An alternative to requiring that the connection explicitly declare its traffic parameters is to use MBAC (MEASUREMENT-BASED ADMISSION CONTROL) [Gibbens95, Floyd96, Gibbens97, Jamin97a, Jamin97b, Wang99]. In this case, the network measures its current load and then uses these measurements to make a decision about whether it should accept a new connection. This approach has the advantage that it relaxes the requirement that users or applications know *a priori* the statistical details of the traffic to be sent. In many cases these parameters cannot be known in advance because the content of the connection may be dynamically generated (e.g. by a voice-over-IP conversation). Moreover the packet flow may be modified *en route* to a bottleneck due to buffering at intermediate nodes.

Obviating the need for applications to parameterize themselves is highly desirable in an environment like the Internet, where new applications are frequently developed and deployed. In addition, since the Internet is a public access network which currently has poor support for network charging or policing, it is unlikely that the network would be able to trust traffic parameters declared by users.

2.2.4 Incentive compatible resource pricing

As an alternative to the above approaches which involve co-operation between users or across the network, a number of proposals for incentive compatible pricing have been put forward [Cocchi91, Cocchi93, Shenker93, Shenker94, Shenker95, MacKie-Mason95, Shenker96, Clearwater96, Paschalidis00, Falkner00]. The basis of all of these is that since congestion has a negative effect on all users of the network, those causing congestion should be forced to pay for doing so.

All attempts to provide an incentive for users to co-operate to prevent congestion rather than simply mandating co-operation lead to the associated problem of enforcement. At the same time they enable users to use the network even during times of congestion simply by paying for their contribution to the congestion. This gives greater flexibility in network access than allowed by traditional CAC schemes. A variety of pricing algorithms have been proposed and studied, based on effective bandwidth [Courcoubetis97, Courcoubetis98a, Courcoubetis98c], traffic priority [Sairamesh95, Odlyzko99a, Odlyzko99b], and on achieving proportional fairness [Kelly97a, Kelly98].

Implementation varies based on the underlying technology. Proposed schemes for ATM networks include use of pricing at connection admission to encourage users to correctly declare QOS parameters [Kelly97b], and use of prices to cause users to exert control over their cell transmission rate [Murphy94].

In the Internet, schemes have been proposed that would allow users to place the price that they would be willing to pay for a packet's transmission into the packet's header [MacKie-Mason95]. The network would then make a decision as to whether the packet should be transmitted or dropped, and would charge users the highest price of all the dropped packets. Alternatively, a price can be charged at connection setup time under RSVP [Tassel97] and TCP [Edell95].

With the advent of packet marking techniques in the Internet, where individual routers can mark packets based on their own load and hence the congestion in the network, a number of related pricing schemes have been proposed. They make use of a small number of bits in the IP TOS byte to signal to end-systems that a packet caused congestion [Key99a, Kelly00]. These signals can either be used purely for congestion control at end-points, or can be used by ISPs to charge users for emitting traffic at times of congestion.

By combining incentive compatible pricing with admission control, *distributed admission control* schemes are produced [Gibbens99a, Breslau00, Kelly00]. These allow the edge nodes to probe the network and then make their own admission control decisions as to whether or not to transmit at this time. These decisions can be based on the reported price and implemented at edge devices [Gibbens99a], or as part of the end-system protocol [Kelly00]. This removes the problems of per-flow monitoring and traffic characterization to the edge of the network, either to edge devices or to the end-systems themselves.

In situations where users wish more control over their traffic, schemes such as WTP (WILLINGNESS TO PAY) [Key99a] are appropriate. By paying based on a price set by the network for marked packets received, users can use their 'willingness to pay' as a signal that they assign traffic a high or low value. This allows differentiation between users' traffic by the network, and allows

users flexibility in choosing the service they receive. The price for this is that users now have to deal directly with fluctuation in the price being applied.

To ameliorate the complexity associated with the more dynamic resource pricing schemes, agent based systems [Courcoubetis98b, Courcoubetis98d] automate the process of dealing with price fluctuation. This insulates the user from the details of short timescale fluctuations in price, while still allowing them to make decisions to transmit based on the network advertised price, and hence based on congestion in the network.

Whether the transmitter or receiver pays generally depends on the service being provided; out-of-band mechanisms for settlement may be used. For example, users might pay monthly subscription charges for a real-time media service, with the service provider (as the originator of the majority of the traffic) paying network operators for marked packets on a day-to-day basis. This simplifies the billing problem from the service provider's point of view as they now only have to bill users a fixed monthly amount and deal with traffic monitoring and fluctuating prices from the far smaller number of operators. Additionally, it makes the service much easier to use from the user point of view as they no longer have to deal with rapidly varying prices for services.

2.2.5 Discussion

This section briefly presented the principal categories of resource control in computer networks. The first three – fair share resource allocation, admission control, and measurement based admission control – have different trade-offs, dependent on whether the extra accounting work required by admission control is an acceptable price for the tighter control of resource allocations, and on how easily and accurately traffic sources may be characterized. It should be noted that the choice is not exclusive – it is possible to deploy protocols that implement different resource allocation schemes within the same network.

Specific applications of network resource control to the Internet are discussed in the following section. Techniques such as queue management and TCP-friendly rate control discussed in Section 2.2.1 are orthogonal to the admission control mechanisms. Queue management and rate control aim to share bandwidth fairly and smoothly between competing flows, whereas admission control aims to ensure that there are not so many flows competing for the resource that the fair share mechanisms fail.

The final category, incentive compatible pricing, is most akin to the work developed in this dissertation. Its principal failing is in the complexity of the implementation details. Although much theoretical work has been done on

pricing algorithms and marking strategies, this has only addressed the problem from the end-to-end point of view of the network, and not considered interconnection of different operators.

It is also still not clear how accurately the theory models reality, and how well many of the results concerning such attributes as price and network stability will translate into implementation. Furthermore, the implementation details of user interaction with the various network pricing and brokerage schemes have yet to be fully addressed [Oliver00]. Aggregation of marks, futures schemes [Semret99], user interfaces [Bouch99, Bouch00], and billing systems [Edell95, Chu99] are all implementation details still being addressed.

2.3 Internet resource control

This section briefly describes proposals for extending IP and the Internet protocols in general to better support traffic engineering.

2.3.1 Integrated Services

The IETF's INTSERV (INTEGRATED SERVICES) effort grew out of a desire to provide support for multimedia and other enhanced services in the Internet [RFC1633, IntServ00]. Influenced by connection oriented networks such as ATM, an end-to-end signalling protocol, RSVP [RFC2205, RFC2750], was developed. This allows QOS to be associated with paths through the network and policy to be used for admission control at the edges and in the core of the network.

RSVP requests that resources be reserved in routers along a uni-directional path at the instigation of receivers. It provides support for a number of reservation styles, including unicast and many-to-many multicast traffic. It does not provide routeing support, expecting to use existing routeing information. A reservation is made using a *flow spec* to specify the QOS to be delivered, and a *filter spec* to specify the traffic to which it should be applied. The filter spec identifies the transmitter or transmitters and allows wildcards. Reservations may be merged in certain cases by routers to reduce the amount of state they must store.

2.3.2 Differentiated Services

The IETF's DIFFSERV (DIFFERENTIATED SERVICES) effort is an alternative approach to providing QOS in the Internet [RFC2475, DiffServ01]. Rather than being based on the idea of per-flow resource reservation, it assumes

that much coarser service differentiation will be satisfactory given the plentiful nature of bandwidth in the future. Using parts of the TOS byte, or DSCP (DIFFERENTIATED SERVICES CODE POINT), as identification, PHBs (PER HOP BEHAVIOURS) are defined which enable routers to give different levels of service to packets sporting different DSCPs.

Standardized PHBs are mapped onto DSCPs by the IETF, and currently consist of *best effort* (standard Internet service with the added requirement that this class must not be starved), *expedited forwarding* [RFC2598], a low latency, low jitter, low loss service, and *assured forwarding* [RFC2597], a low loss, ordered service. Experimental PHBs may also exist, but are not guaranteed to be supported. Different operators may implement the PHBs differently, the only proviso being that the standardized PHBs must be represented by their mandated DSCPs. Traffic between operators will be managed through SLAs, with promotion and demotion of traffic between PHBs allowed in order to meet the specified SLAs[5].

2.3.3 Multi-Protocol Label Switching

MPLS (MULTI-PROTOCOL LABEL SWITCHING) is a framework for forwarding based on a short, fixed-length label in the packet header [RFC3031, MPLS]. By divorcing route determination from the forwarding mechanism, it enables more complex treatment of traffic streams than is possible in current IP networks. This allows packets to follow paths determined by other considerations than the pure hop-by-hop destination-routed model of IP, and specifically by traffic engineering considerations.

Packets entering the network are assigned to a FEC (FORWARDING EQUIVALENCE CLASS) which is then used to assign an appropriate label. A packet may be classified into a FEC by a variety of means, ranging from simple destination-based classification, analogous to the current IP routeing model, to classification based more generally on the packet's headers or content.

The label is either inserted into the link layer header if fields are available, or the packet is encapsulated by a special-purpose *shim* header. This specifies the LSP (LABEL SWITCHED PATH) along which the packet will be forwarded. From this point the network need only perform lightweight label switching operations at each node, until the packet reaches the end of the LSP. Switching points may consist of well-known switch technologies, such as ATM or frame relay, or may be custom built to support MPLS. At a switch the label serves to index into a LIB (LABEL INFORMATION BASE), a table containing the next forwarding hop and a new label. Switches construct their LIBs using

[5] SLAs will be discussed in more detail in Chapter 5; in brief, they are agreements between operators that specify the service and corresponding remuneration to be provided.

an LDP (LABEL DISTRIBUTION PROTOCOL), which may be classified according to how LIB entries are created:

Request-driven LDPs use the messages of a control protocol such as traditional ATM signalling [ATMF-UNI96,ATMF-PNNI96], or RSVP [RFC2205].

Topology-driven LDPs use information derived from network layer routeing protocols, such as BGP [RFC1771], OSPF [RFC2328], or the generic MPLS-LDP [RFC3036].

Traffic-driven LDPs use information gathered by monitoring the traffic streams being switched, as with IP switching [Newman98] for example.

The separation in MPLS of these three network functions – packet classification, packet forwarding and label distribution – simplifies the data path, and allows greater service differentiation between aggregates. Service differentiation at a range of packet aggregation granularities is supported, as is additional functionality such as traffic engineering. MPLS also provides a platform for building VPNs (VIRTUAL PRIVATE NETWORKS) to which resource guarantees may be given [RFC2764, Isaacs00], and hence supporting multiple control systems with resource partitioning [Mortier01].

2.3.4 Discussion

The IETF's INTSERV effort aims to extend the Internet service model to support multimedia and data traffic within the same infrastructure. Although partially successful, in that the RSVP signalling protocol was defined and implemented, this approach has not been widely adopted. There are two principal problems: QOS is signalled on a per-flow basis, and this is believed to be unscalable in the Internet; and due to the need to support the new signalling protocol no benefits can be seen from a partial deployment. If just one router on the desired path does not support RSVP, no guarantee can be given concerning the QOS that traffic on that path will receive.

There is also considerable debate as to whether or not per-flow QOS is suitable for the Internet in any case. Given the widespread use of elastic protocols such as TCP and the proliferation of adaptive real-time protocols such as RTP and associated adaptive applications, it seems that adequate performance does not require per-flow guarantees.

In response to this, the DIFFSERV effort aims to provide a much simpler, more evolutionary solution. The result is that benefits can be seen with only partial deployment, and without requiring end-to-end signalling. However, DIFFSERV is very much work in progress and a number of important questions have yet to be resolved. For example, the precise definitions of the standardized PHBs are still being discussed, and further standardized PHBs

may be required. The definition and implementation of policies to prevent users requesting that all their traffic is given the most resource hungry PHB is a problem yet to be addressed.

In addition, the DIFFSERV architecture [RFC2475] explicitly makes no statement as to how PHBs should be implemented, leading to concern about the strength of guarantee that can be made about the QOS that traffic allocated to a given PHB will receive as it crosses administrative boundaries.

MPLS was originally intended to improve the performance of IP over connection oriented networks such as ATM. However, as IP router performance has improved, it is now principally intended to enable more extensive traffic engineering capabilities for IP traffic. The mapping of IP traffic into FECs and then of FECs onto LSPs allows aggregates of IP traffic to be treated together, avoiding some of the scalability problems associated with INTSERV.

Considerable thought has also been given to efficient support within MPLS for a variety of network types such as ATM and frame relay, as well as support for newer network services, such as VPNs. As such it can be considered a supporting layer for the INTSERV and DIFFSERV efforts discussed above. However, concern remains about how MPLS will support resource management *between* domains as well as within domains [Mortier01].

2.4 Internet routeing

Routeing protocols enable a picture of the state of the network to be built up by participating routers, allowing them to route packets toward destinations. Routeing protocols are generally placed into one of two classes [Perlman00]:

Link State protocols, where each router explicitly advertises information to other routers in the network about the nodes to which it is connected. Routers then use a suitable path finding algorithm, such as Dijkstra's Shortest Path First [Dijkstra59], to work out to where packets destined for a node should be sent.

Distance Vector protocols, where each router advertises the prefixes which it currently knows how to reach along with an associated cost; neighbouring routers receive this information and update their own routeing tables if new routes have been introduced, old routes have been deleted, or routes have changed costs. Such protocols are often considered to implement a distributed shortest path first algorithm [Chandy82].

Distance vector protocols are generally considered to have better memory scalability as each router need only store state about those destinations in the network that it can reach. In contrast, link state protocols require each

router to keep an LSA (LINK STATE ADVERTISEMENT) for every node in the network, where each LSA contains information about the neighbours of that node.

The network bandwidth and computational scalability of the two protocol types in an arbitrary network is less clear, but both are considered to require only a modest amount of bandwidth and have incremental computation versions. However, link state protocols do provide more functionality, and generally converge faster than distance vector protocols [Zaumen91, Zaumen92]. Consequently, link state protocols are usually considered preferable in situations where the increased memory requirements are not prohibitive [Perlman00].

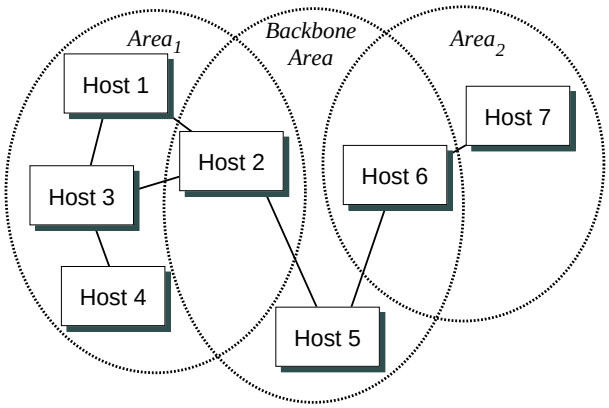
Routeing in the Internet is effectively performed hierarchically, with routeing within an AS, or *intra*-AS routeing, performed by the OSPF (OPEN SHORTEST PATH FIRST) or ISIS (INTERMEDIATE-SYSTEM INTERMEDIATE-SYSTEM) protocols, and routeing between ASs, or *inter*-AS routeing, performed almost exclusively by BGP. Since OSPF and ISIS are very similar, both being link state protocols for intra-AS IP routeing[6], only OSPF will be described. As the only widely deployed inter-AS routeing protocol and the basis for the work described in Chapter 4, BGP is also described. Additionally, as an example of a previous Internet routeing protocol that dynamically calculated route metrics based on congestion, the HELLO protocol is described.

2.4.1 Intra-AS routeing: Open Shortest Path First

OSPF (OPEN SHORTEST PATH FIRST) [RFC2328] is a link state routeing protocol, heavily influenced by the ISIS [RFC1142] protocol. It is intended to operate *within* an AS as an IGP (INTERNAL GATEWAY PROTOCOL). As shown in Figure 2.1, it splits the AS into *areas* formed from contiguous networks and hosts, including the routers that interface between the networks. This splits routeing into two: *intra-area routeing* and *inter-area routeing*. Intra-area routeing refers to routeing within an area and is performed solely on the basis of routeing information obtained within that area.

Inter-area routeing is routeing between areas using the *backbone* of the AS, formed from networks not contained within any area, their routers, and routers contained in multiple areas. Inter-area routeing has three stages: routeing the packet from the source to the router in the source area that connects to the backbone; routeing the packet through the backbone; and then routeing the packet to the destination from the router in the destination area that connects to the backbone. Routers with all interfaces within a

[6] Strictly ISIS itself is an ISO CLNP (CONNECT-LESS NETWORK PROTOCOL) routeing protocol, and the version which routes IP traffic is termed 'integrated ISIS.'



Hosts 2 and 6 are the inter-area routers, routeing packets between Area 1 (containing hosts 1, 2, 3, and 4), and Area 2 (containing hosts 6 and 7) via the backbone area which contains hosts 2, 5, and 6.

Figure 2.1: An OSPF example showing two areas and a backbone area.

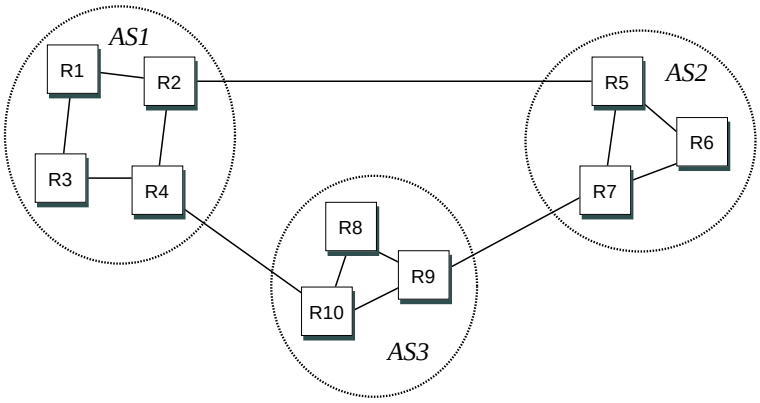
single area or within the backbone run a single copy of the OSPF algorithm. Routers at the boundaries of areas and of the backbone run multiple copies.

Each router builds up a view of the topology of its area using the LSAs from the other routers. The process starts with the routers announcing themselves through the OSPF *hello* protocol[7], allowing the adjacencies and capabilities of routers to be established. Once a router establishes adjacency with another router, it waits for LSAs from that router, whilst sending its own advertisements to other routers. These advertisements allow the current area topology to be distributed to all routers within the area. Border routers summarise the topology of their areas, and distribute these summaries into the backbone and thence to the border routers of other areas.

The routeing process itself is based on running the shortest path first algorithm over the routeing tables constructed from the LSAs from each router. This forms 'shortest paths' from the router running the algorithm to all routers and networks within the area, and to all the border routers for routes outwith the area. The 'length' of a path is calculated based on its metric, of which there are two types.

Type 1 metrics are equivalent to the link state metrics used for internal routes, and are guaranteed to be less than *Type 2* metrics used for routes learnt from sources external to the area. This ensures that internal links are always used in preference to external links, under the assumption that routeing between areas and ASs always costs more than routeing within an area or AS. OSPF also originally allowed routers to provide a separate set of

[7]Not to be confused with the HELLO protocol discussed in Section 2.4.3.



AS1 contains R1, R2, R3, and R4; AS2 contains R5, R6, and R7; and AS3 contains R8, R9, and R10. Inside the ASs, interior BGP is operating as a fully connected mesh of BGP peering arrangments with a session between each pair of ASs.

Figure 2.2: A BGP example showing 3 peering ASs.

routes for each IP TOS, for cases where metrics vary based on the TOS. This ability was removed in a later revision due to lack of implementation experience [RFC2178]. However, the LSAs can still carry the required information for compatibility reasons.

2.4.2 Inter-AS routeing: Border Gateway Protocol

BGP (BORDER GATEWAY PROTOCOL) [RFC1771,Stewart99] is a *path vector* protocol – a distance vector protocol that associates the path to the destination with the destination in the information it distributes. It can operate both within an AS as an IGP, and between ASs as an EGP (EXTERNAL GATEWAY PROTOCOL)[8] as shown in Figure 2.2. Its use as an IGP is typically restricted to distribution of externally learnt prefixes within an AS.

When a BGP speaking router becomes active, it starts sessions with other BGP speakers known as *peers*. These sessions take place over TCP connections, and a given BGP speaker may have many peering sessions active at one time. Each speaker keeps two tables for each peer with which it currently has a session, one storing the information it receives, one storing the information it will re-advertise. Finally, each speaker keeps one further table for the routes it is currently using.

In addition to the `OPEN` message used to start a BGP session between two speakers, there are three other message types: `KEEPALIVE`, `NOTIFICATION`,

[8]Note that EGP without citation in this dissertation always refers to the class of exterior gateway protocols, and never to the specific protocol, EGP [RFC904].

and UPDATE. The first two respectively enable two peers to confirm that the session between them is still alive, and to inform each other of errors during the lifetime of the session. The principal means for communicating route information is the UPDATE message, used to advertise and withdraw prefixes. Additionally, an UPDATE message may contain a number of *path attributes* which apply to all the prefixes being advertised in that message.

Receipt of an UPDATE message can cause the BGP peer to recalculate its routeing table. Assuming that the filtering policies applied to the other peer allow this peer to accept the route information contained in the message, this calculation is performed in two parts. The first is based on longest prefix match, so the most specific route to a prefix is always used. If there is a tie – that is if two ASs are advertising *exactly* the same prefix – then the path attributes associated with the prefix are considered, with the route selection process stopping as soon as the tie is broken and a unique route discovered. The two principal attributes considered are:

- LOCAL-PREF: a locally valid metric (higher is better) associated with a path. If a route is learnt via an external BGP peer or via static configuration, this metric will be recomputed at the learning router, otherwise it is learnt from the advertising router.
- AS-PATH: this field has the AS number prepended as the route is advertised throughout the network. Shorter AS-PATHs are selected over longer ones if there is no unique route remaining after the LOCAL-PREF attribute has been considered.

Other attributes are standardized and can be used if the two above are insufficient. Eventually all ties will be resolved, ultimately by choosing the route learnt from the router with the lowest BGP identifier (typically the highest IP address associated with the router).

2.4.3 Dynamic metric routeing: the HELLO protocol

The HELLO protocol[9] [RFC891] is now deprecated and no longer supported by most routers, but it is of historic interest as a routeing protocol that dynamically computed its link metrics based on congestion. It used a precursor of the Network Time Protocol [RFC1305] algorithm to estimate the RTT from each node to those nodes to which it was adjacent. This estimate was communicated to each adjacent node.

When a routeing table update at a node was triggered, it would use the RTT estimates from itself to its adjacent nodes in conjunction with the RTT estimates associated with each route to decide whether or not to alter the route

[9] More properly, the DCN-Local Network routeing protocol.

for a given destination. Lower RTT estimates signified lower congestion and so were preferred; changes in estimate of less than 100 ms were rounded up to 100 ms. In the case of an update being received for a connected network, the gateway node simply used its own estimate rather than concerning itself with the estimate contained in the route update.

Although the HELLO protocol initially worked well it was found to suffer from a number of problems. As the network grew in size, it did not scale well[10]. Also, as packet timescales within the network altered and the load characteristics of the network changed, it became clear that RTT was not a good basis for a routeing metric. At times of high load, routeing changes can lead to large changes in load, and hence queueing delay, on links. This consequently causes too high a degree of oscillation in the routeing tables.

The fundamental problem is that the RTT varies on too short a timescale to be generally useful as a measurement of load. In particular, if the network becomes heavily loaded, the lengths of queues in the network can become very large, causing similar increase in the RTT. This can cause a positive feedback cycle to occur, where a small increase in traffic in the network causes the routeing protocol to attempt to reroute traffic down unloaded links. This causes those links in turn to become congested, and so the protocol tries again to redistribute traffic. The effect of this is to cause the RTT on these links to begin to oscillate wildly, further increasing the frequency at which the protocol reroutes traffic.

The routeing metric was revised in 1987 [Khanna89] to smooth the measured delay values and limit the relative change in metric between successively reported values; to normalize the reported costs to take into account how the network might react to such a change; and to cause the algorithm to shed load from overloaded links more gradually. However, as the Internet increased in size and competing commercial entities began to take part, use of a common delay metric in this manner was dropped since a suitable trusted metric could not be decided upon [RFC975]. Chapter 4 will discuss use of measures of congestion as drivers for BGP routeing metrics.

2.4.4 Discussion

As stated previously, OSPF is intended as an IGP. Consequently, whilst it has support for metrics enabling choice between multiple routes, its routeing hierarchy – requiring that all packets wishing to travel between two areas must do so via the backbone – is not sufficiently scalable for use as an EGP. Additionally, more recent revisions of the OSPF v2 standard [RFC2178, RFC2328]

[10] Indeed, it also seems to have been a precursor to the EGP [RFC904] protocol, the scaling properties of which led to the initial development of BGP.

recommend that implementations should continue to accept PDUs (PROTOCOL DATA UNITS) with TOS routeing options, but that this information should not be utilized due to 'lack of implementation experience' required by the IETF's standardization procedure. Notwithstanding this, recent research [Fortz00, Wang01, RFC2676] has suggested dynamically setting OSPF weights to achieve traffic engineering objectives. Similar techniques have been proposed for MPLS [Elwalid01].

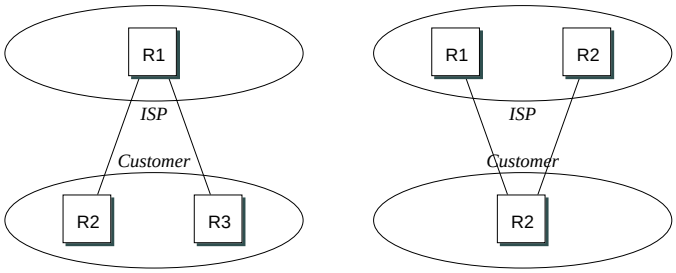
BGP is designed as an EGP and therefore does not impose such a restrictive hierarchy as OSPF. However, due to the peer-to-peer nature of BGP routeing and the desire to offload traffic to another AS as soon as possible, route asymmetries are common [Paxson97]. This can lead to situations where one direction of the path between two end-points has vastly different characteristics to the other.

Furthermore, it lacks a globally useful metric and so provides poor support for situations where a choice between routes must be made. Current deployments rely on artificially increasing the length of the AS-PATH by prepending multiple copies of their own AS number to those routes they wish to discourage – study of a sample BGP database from the KPN-Qwest peering point suggests that approximately 8% of best routes have been so treated. This implies that there does exist a desire amongst operators to be able to influence routes taken by others. Additionally, such techniques appear to have a detrimental effect on the efficiency of the network – interaction with filtering policies appears to cause approximately 20% of paths to be inflated by 50% or more since not all routes are available to all ASs [Tangmnarunkit01].

A globally valid path attribute for BGP, the *destination preference attribute*, appears to have been discussed in the IETF between 1994 and 1996 but seems not to have become sufficiently advanced to be standardised in an RFC[11]. More recent work has suggested use of an *avoidance level* attribute for routes, to enable safe backup routeing [Gao01]. This is similar to the price path attribute proposed in Section 4.2.3, but intended to be used in a more restricted manner.

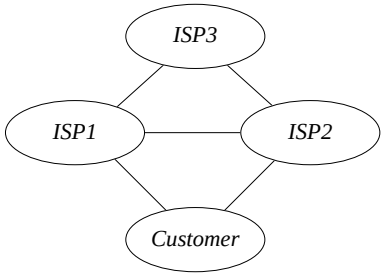
Situations where there is a choice between routes seem likely to arise due to the desirability of multi-homing for reasons of reliability and performance for customers. *Multi-homing* occurs when a customer connects to the Internet in more than one place as depicted in Figure 2.3. This can have strong implications for the aggregatability of addresses, generally considered to be of utmost importance in the Internet. As the network has grown and address space become more fragmented, one of the most significant contributions of BGP is that it enables routeing by address aggregates, meaning that not every

[11] Notwithstanding this, it does appear to be available in at least two deployed BGP implementations.



(a) Customer connects to ISP from two egress routers.

(b) Customer connects to ISP at two points.



(c) Customer is multi-homed to two ISPs which connect via a third ISP.

Figure 2.3: Examples of BGP multi-homing configurations.

router need know about every routeable address. This is true both for those routers within an AS and those that interconnect ASs.

Multi-homing can occur either by the customer connecting to a single ISP in multiple places, or by them connecting to the Internet through more than one ISP. In the former case the ISP can attempt to influence the customer's choice of route in two ways. If the customer has multiple egress routers which connect to the ISP's network as in Figure 2.3(a), the ISP can make use of the `MULTI-EXIT-DISCRIMINATOR` or `LOCAL-PREF` attributes to control traffic distribution. Alternatively, if the customer has one egress router connecting to the ISP in multiple places as in Figure 2.3(b), different prefixes can be advertised as reachable from the different points in the ISP's network, allowing the ISP to control the distribution of inbound customer traffic.

The most problematic case is where a customer wishes to use multiple providers for Internet connectivity as in Figure 2.3(c). In this case care must be taken

over who 'owns' the IP addresses that the customer will use. If the customer uses addresses delegated by one ISP then it is likely that they will be announced by that ISP as part of an aggregated block, but by the other ISPs as addresses specific to that customer (since those ISPs cannot aggregate the addresses delegated by the first ISP). In this case the longest-prefix-match behaviour will take over and potentially cause traffic for the customer to arrive via all *but* the first ISP.

Similar problems can arise if the customer is delegated addresses out of all the ISPs' address spaces – care must be taken to avoid 'magnetic' longest-prefix-match behaviour if the addresses delegated from one ISP are advertised to the others to give increased reliability for the customer. The alternative is for the customer to get its IP addresses from some other registry. However, this decreases the aggregatability of routes in the Internet, since all ISPs are then unlikely to be able to aggregate the customer's addresses into their existing address blocks.

Some of these problems can be solved by using the BGP *community attribute*. This is an optional, non-transitive attribute containing policy information concerning the associated routes. There are three values standardized: `NO-ADVERTISE`, `NO-EXPORT-SUBCONFED`, and `NO-EXPORT`[12]. All control the scope over which a route will be advertised. The first restricts advertisement to the router receiving the advert, the second to the sub-AS that receives the advert for a confederation[13] and the last to the AS receiving the advert.

These community attributes notwithstanding, problems remain with the expression of policy in BGP. Although the standardized community values allow expression of policies that solve a number of the more common problems arising from such situations as multi-homing, they are not flexible. As they can only express whether or not a route should be re-advertised, they cannot easily be used to choose between possible routes, and as a consequence they are not useful for more general traffic engineering. Currently they are also manually configured and configuration cannot easily be automated, leading to extra network management burden.

2.5 The structure of the network

This section describes the model of the network underlying the arguments in this dissertation. It has undergone dramatic change since the original In-

[12] Others are used with semantics assigned on a local basis, as discussed in Section 4.2.1.

[13] AS confederations are not discussed in detail here; they are a mechanism by which the $\mathcal{O}(n^2)$ scaling properties of interior BGP can be remedied. They are briefly described in Section 4.3.5.

ternet with its reliance on the backbone provided by the US NSFnet national network.

2.5.1 The edges

Generally speaking, the network can be divided into a number of parts. At the edges of the network users connect to ISPs to gain access to the network. Access methods range from analogue dial-up services, typically at around 48 kb/s through to cable modem and DSL based technologies at between 512 kb/s and 10 Mb/s. A smaller proportion of users on corporate and academic networks also connect via LAN technologies such as Ethernet or leased lines at speeds of 10 Mb/s upwards. Much of the content available on the network is provided through such connections, in the form of commercial web sites and streaming media distribution.

The other form of user connected to the network is the ASP (APPLICATION SERVICE PROVIDER). These typically provide more active services to users from well-connected locations by collocating with ISPs. Such services range from basic web servers, to more complex proxy and computation services, often with some associated form of charging [Ensim00].

2.5.2 The core

Moving further into the network, the user-facing ISPs typically use larger network providers to provide connectivity to other networks. Depending on the level of service required in terms of bandwidth, reliability, and so on, the ISP may use many such network providers. The network provider will typically inject routes into the ISP's network to provide connectivity for the traffic originating from the ISP, and will advertise routes for the address space owned (or rented) by the ISP.

Finally, at the core of the network the large network providers peer together at exchange points, providing transit services for traffic injected by themselves and their customers. There are two types of such provider. The first operate by installing and maintaining their own physical networks, and thus require a large capital investment leading to significant sunk cost in the infrastructure. For this reason they are often originally telephone operators, e.g. Sprint and AT&T. The second type rent capacity from existing network owners and 'fibre providers' such as Cable&Wireless, allowing them access to the market without such high entry cost but at the cost of some flexibility.

2.5.3 Peering points

ISPs peer together at *peering points*, either *public* or *private*. Public peering points, the norm until relatively recently, generally support a large number of providers. Networks are connected using, for example, 100 Mb/s and 1 Gb/s switched Ethernet, and BGP information is exchanged between providers according to previously negotiated SLAs and the associated imposed policies and filters. Examples of such peering points are the LINX [Linx01] and the MAE peering points [Mae01].

Private peering points are usually formed pairwise between operators, who may choose to make available higher bandwidths than at public peering points. Due to the more restricted participation SLAs are much simpler in these cases, as is implementation of policy to restrict peering. The common case is that once agreement to privately peer is reached, traffic and routes are exchanged without further interference, although *a posteriori* monitoring is still likely to be carried out to enable dramatic changes in traffic characteristics to be dealt with.

2.5.4 Discussion

This decentralized structure, with ISPs connecting at many points, potentially to many other ISPs, allows a great deal of flexibility in the network. However it is not without problems, principally boiling down to accountability.

For valid historical and technological reasons, the Internet does not provide good mechanisms for accountability [Clark88]. Originally designed to provide end-to-end connectivity for co-operating users across a small number of co-operating public networks, there is little inherent support for monitoring, authentication or policing of network use. As described above, protocols such as TCP rely on compliant implementations to ensure that some approximation to fairness for users is achieved.

As the Internet becomes more commercial and provides more socially fundamental services such as telephony, this problem manifests itself in two ways. The first is of a commercial nature: those responsible for the interconnecting networks need to know the quality of the service they are providing and receiving in order that they can effectively manage their networks and the agreements they enter into with other operators. The second is that as the network increases in popularity and use, government agencies and regulators become involved. This involvement typically leads to extra requirements on operators to provide audit trails and information to allow the regulators to ensure that an acceptable service is being provided at an acceptable price.

One aim of the work in this dissertation is to address some of the problems

posed by the way these changes have altered the way the network is structured and consequently operated.

2.6 Summary

This chapter has provided some background to the technologies on which the work described in this dissertation relies. The relevant Internet protocols were described, along with current techniques for providing resource allocation within networks in general and the Internet in particular. In addition, techniques for implementing routeing within the Internet were discussed and the structure of the resulting Internet described.

This chapter illustrates that the requirements of networks have evolved past the simple services provided by IP. Better control of the network and more facilities for resource management, particularly resource management between operators, have become requirements on the network. Although individual users are perhaps more easily satisfied as bandwidth becomes plentiful, there is still a requirement that inter-operator management be performed due to the large volumes of traffic and money involved. SLAs between operators are still specified and must be managed and met.

Having illustrated the continuing requirement for traffic engineering at multiple timescales, the next two chapters present attempts to implement this for the Internet, beginning with control timescales in the following chapter.

Chapter 3

Control timescale traffic engineering

This chapter discusses the application of multi-timescale traffic engineering at control timescales in the network. Existing deployments of congestion control approaches at data timescales are considered, and it is argued that these are insufficient for the smooth operation of the network. Potential problems with and benefits of control timescale traffic engineering are subsequently discussed, and finally simulation and implementation results are considered. Control timescale traffic engineering is shown to improve the operation of the network in terms of the bandwidth achieved by users' flows.

3.1 Internet congestion control

Current models for congestion control in the Internet rely solely on congestion control at data timescales. Congestion is controlled via the individual packets allowed to enter the network. Information is provided to hosts concerning the state of the network, as seen by individual packets. This can be inferred as in TCP where the protocol detects the onset of congestion by attempting to detect if a packet was dropped [Jacobson88, Fall96], or explicit as in ECN [Floyd94].

Alternatively, protocols such as RTP [RFC1889] attempt to use more explicit information about the delay on the path between transmitter and receiver to detect when the network is becoming congested. This information can be aggregated and sent to the transmitter by the receiver, in order that the transmitter can alter its transmission rate or coding scheme appropriately.

Schemes providing more information to the hosts have been suggested, as

described in the ECN modifications to IP [RFC2481] for example. This enables information to be provided to hosts before drops occur and also to be provided more smoothly, i.e. at a finer granularity.

3.1.1 TCP congestion collapse

Although these methods give satisfactory performance in many cases, it is still the case that a TCP flow may observe near-zero goodput when a large number of TCP flows share a bottleneck link in the Internet [Morris97, Morris99]. The consequent competition for resources results in catastrophic collapse of the per-flow performance, even though the link is operating at full utilization.

Congestion collapse occurs as each TCP flow probes for available bandwidth to see if it may increase the amount of data it has 'in-flight' in the network. Since current implementations of TCP have a minimum probe bandwidth of one segment per RTT, or one segment per RTO (RETRANSMISSION TIME OUT) if the probe packet is discarded, then if too many TCP connections are admitted, the total probe bandwidth can itself exceed the capacity of the bottleneck link. This results in a substantial increase in retransmitted data and therefore wasted bandwidth, in addition to woefully inadequate per-flow performance. Given the inability of current TCP implementations to back-off further, the congestion control problem at this point has become network-centric, rather than host-centric, and so it requires appropriate network controls.

The state of congestion collapse has been observed on the UK-US SuperJANet transatlantic link. This link was a major bottleneck for traffic flowing from the US to UK universities, and has historically been under-resourced relative to peak demand. Given the introduction of usage-based charging on this link[1] – and such measures show every indication of becoming more widespread – ensuring reasonable goodput in such cases has become important in order to limit the total cost expended transferring data.

3.1.2 TCP and user utility

Congestion collapse itself is a somewhat extreme scenario. However, even before full congestion collapse occurs, there is often a minimum TCP bandwidth required to achieve a minimal session-level user utility. For example, web users who have to wait too long for the objects within a web page to complete downloading may give up and stop, or worse, restart the download. This wastes already scarce network resources, reducing the number of successfully completed TCP connections, which in turn decreases the number of

[1] Approximately £0.02 per megabyte for UK bound traffic in 1998 [UKERNA01].

successfully downloaded pages – the connection and session level goodputs respectively. When a user aborts a flow due to poor performance, bandwidth has effectively been wasted at the very time it was most scarce, since the data already transferred is of little or no use, and restarting the flow will usually require that this data be retransmitted.

3.1.3 TCP unfairness

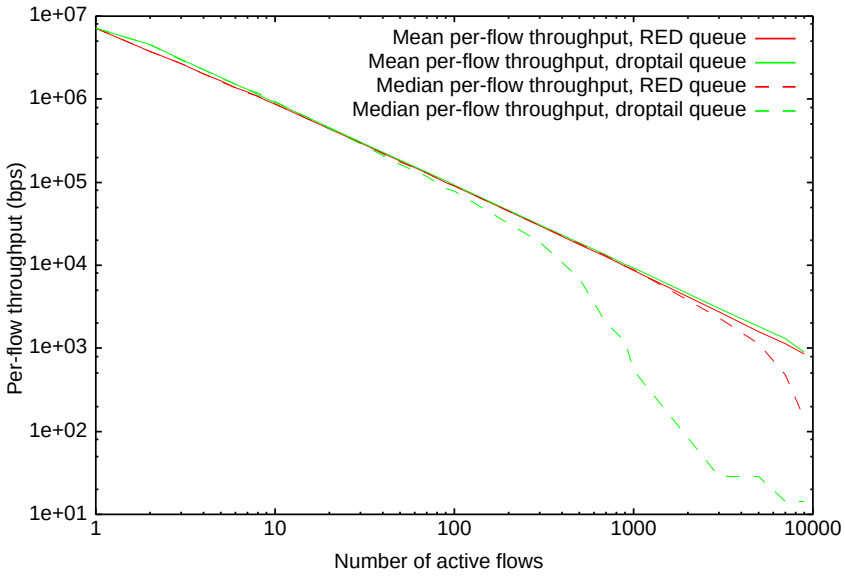
In addition to the wasted bandwidth in those cases where the flow is aborted, TCP does not share bandwidth very fairly in overload situations. As more flows compete for the same resource, the control mechanisms cease to be responsive enough to allow stable bandwidth shares to be achieved. Instead, the bandwidth achieved by a given flow fluctuates quite wildly, over a wide range of timescales. As TCP is an extensively used protocol, preventing the network reaching a state in which this occurs is useful.

3.1.4 Supporting evidence

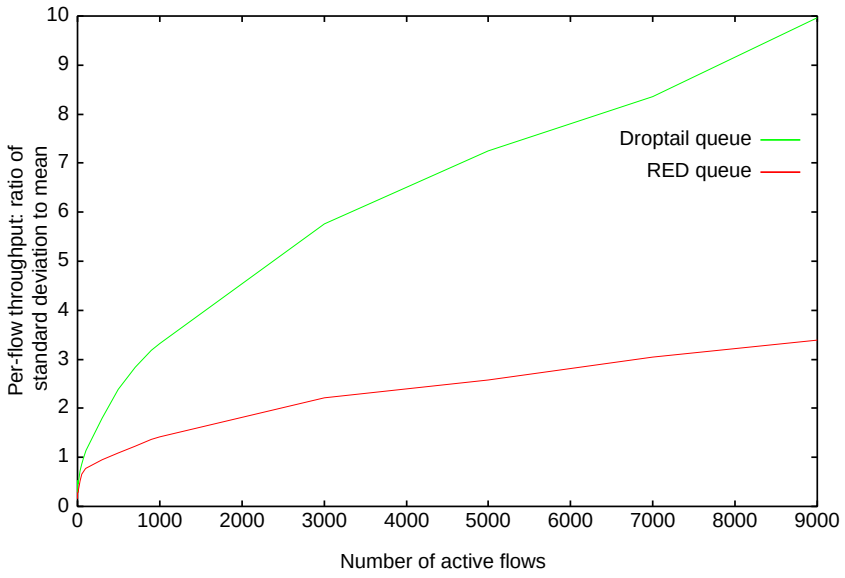
Evidence for the above claims – that TCP can reach a state of 'congestion collapse,' and that even if this state is not reached, per-flow goodput becomes highly variable – is presented in Figures 3.1 and 3.2. These data were generated using the NS (NETWORK SIMULATOR V2) simulator with a simple dumb-bell topology as shown in Figure 3.3. The simulations were run for 300 seconds, with flow start times distributed uniformly between 0 and 10 seconds into the simulation. Bandwidth estimates were taken every 5 seconds.

Two sets of simulations were run using the 'full TCP' model based around code from 4.4BSD and implementing 'Reno' congestion control with SACK. The first provides a base case using a DropTail queue as the bottleneck router; the second uses the more sophisticated RED algorithm [Floyd93] in the bottleneck router.

Figure 3.1(a) shows mean and median per-flow throughput *vs.* number of active flows on log-log scale axes. In the perfect case, one would expect all curves to be identical straight lines with a gradient of -1. However, the figure shows that as the number of active flows increases past 100, the median per-flow throughput achieved through the DropTail router diminishes drastically. The RED router allows the median throughput to be maintained for longer, but also starts to fail in this respect at around 7000 active flows. The fact that the mean throughput values are maintained as expected past 10,000 flows, and are higher than the median values, suggests that the throughput achieved by different flows is varying and is skewed toward lower values.

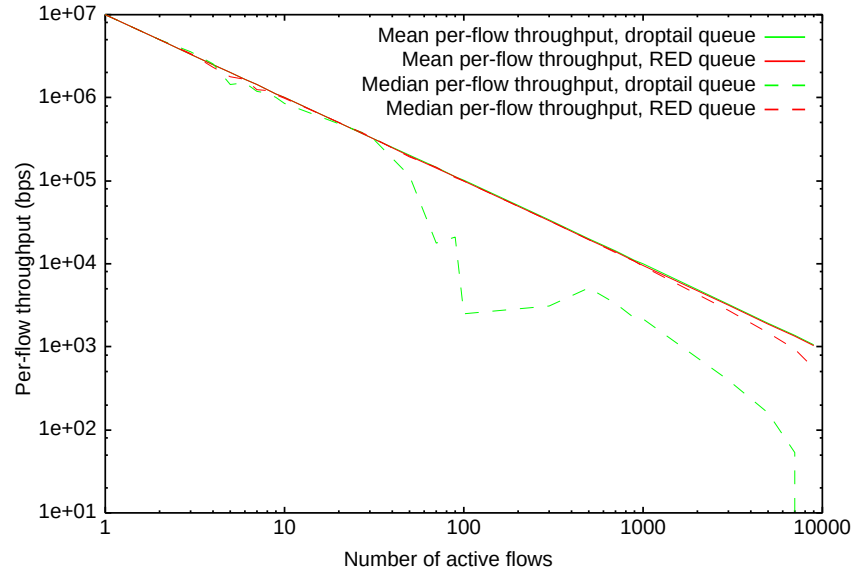


(a) Mean and median throughput *vs.* number of active flows.

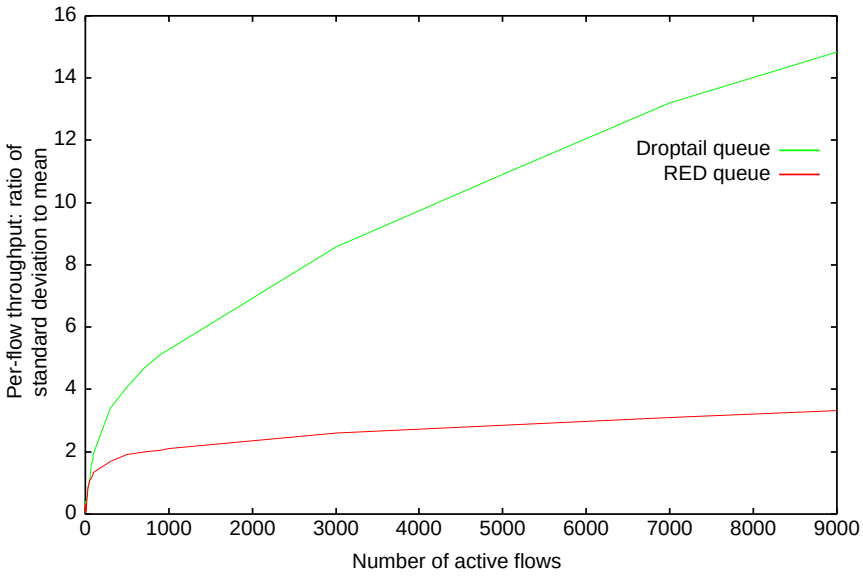


(b) Coefficient of variation (the ratio of standard deviation to mean) of throughput *vs.* number of active flows.

Figure 3.1: Evidence for TCP congestion collapse using 'full' TCP model.



(a) Mean and median throughput *vs.* number of active flows.



(b) Coefficient of variation (ratio of standard deviation to mean) of throughput *vs.* number of active flows.

Figure 3.2: Evidence for TCP congestion collapse using TCP Vegas model.

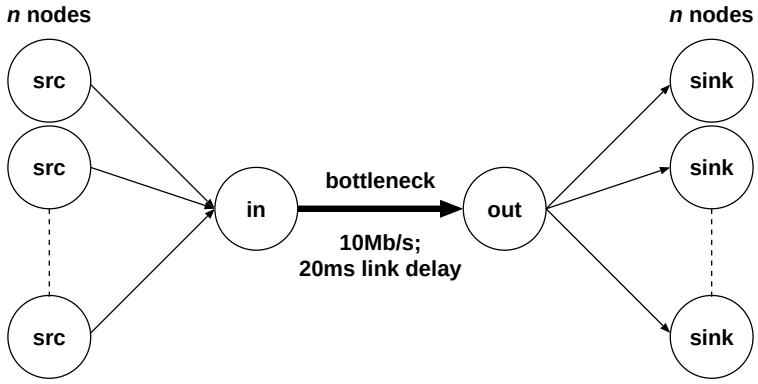


Figure 3.3: The topology used to investigate the overload behaviour of TCP.

This is supported by Figure 3.1(b). This is a plot on linear axes of the coefficient of variation (ratio of the standard deviation to the mean) of the per-flow throughput *vs.* the number of active flows. It shows that this ratio is increasing, again with the effect much more marked for the DropTail case. This suggests that the proportional variability of per-flow throughput is becoming much greater as the number of flows increases.

The same experiments, using both DropTail and RED queues, were then run using the NS re-implementation of TCP Vegas congestion control. This does not model TCP as completely as the 'full TCP' agent, but does use the more sophisticated Vegas congestion control algorithm. Figure 3.2 shows results similar to the 'full TCP' case, indicating that even with the smoother congestion avoidance behaviour of TCP Vegas, congestion collapse due to excess flows is still exhibited.

Using both the 'full TCP' and Vegas TCP models, variability between and within flows over shorter timescales is also apparent. With just 100 flows competing against each other individual flows can achieve very low or even zero goodput whilst others achieve more than their fair share for tens of seconds at a time. Similarly, within an individual flow the goodput achieved can vary substantially over the duration of the simulation. All the above experiments were also performed taking bandwidth estimates over 1 and 10 second periods with no significant differences in the results.

Under both regimes collapse is due to the over-reaction of TCP to congestion. As more flows compete, losses become harder to recover from using the fast-retransmit mechanism. It then becomes more difficult for a flow that reduces its window to recover using slow-start, so flows that reduce their windows are likely to retain small windows for relatively long periods of time, whereas flows that start increasing their windows will increase them very quickly. Consequently, although the link is constantly utilised, individ-

ual flows experience short intense bursts of activity followed by long quiet periods.

Thus, as Massoulié and Roberts [Massoulié99] and Kumar *et al* [Kumar00] argue more abstractly, it makes sense to allow operators to control the admission of traffic at a variety of levels and specifically at the flow level, rather than just at the packet level. This should help to temper the effects of congestion, and ensure that bottlenecks never become so heavily overloaded that real-time services and interactive applications over TCP can make no useful progress. The following section discusses the application and impact of flow admission control in the Internet.

3.2 Internet flow management

This section discusses the requirements for control timescale traffic engineering, and considers different implementation approaches. Mechanisms for flow detection, and flow admission and denial are discussed, followed by consideration of suitable policies to implement using these mechanisms.

3.2.1 Requirements

Protocols such as TCP are considered elastic in their resource demands since they operate relatively satisfactorily within a wide range of resource allocations. Real time protocols are typically inelastic in their resource demands, having a much smaller useful operating range, introducing further complications. They often also place constraints on the amount of delay in the network.

In the case of an elastic protocol such as TCP, the delay constraints are not very stringent – users care about the time for a web page download to complete, not the time for a given packet to arrive. In the case of real time traffic however, the delay of a given packet can noticeably degrade the quality of the media stream. To avoid this applications must use either extensive buffering leading to additional latency, or redundant coding and error correction schemes leading to wasted bandwidth.

To avoid these problems, some means of differentiating between traffic associated with different services is required. Traffic carrying data with real time constraints should not be buffered behind traffic carrying data without real time constraints, but should instead be expedited through the network, or dropped if this is not possible.

As stated in Chapter 1, computer communication is predominantly flow based so it is often the case that dropping entire flows is of more benefit to

the network and the user than allowing the flow to begin and then restricting the bandwidth it can achieve.

The remainder of this section considers the design of suitable admission controllers in more detail.

3.2.2 Estimating the number of flows

As the Internet is a per-hop routed network using a globally valid destination address to route each packet, a *flow* will be defined in terms of the *flow 5-tuple*:

<src.addr, dst.addr, src.port, dst.port, protocol type>

The basic requirement for per-flow control is that the system has reasonable knowledge of the number of flows it is currently carrying. Since the Internet is a hop-by-hop routed packet network, network signalling is not generally performed. Consequently, a network element will not explicitly know the exact number of flows it is carrying at any given moment, so this value must be estimated. Since this must be done in network elements, and since even edge routers may carry many tens of thousands of flows, a good system should not impose significant per-flow calculation or state overheads. Some approaches to estimating the number of flows will now be discussed.

The first is suggested by Massoulié and Roberts [Massoulié99]. The router monitors its output queues and maintains a table of the flow tuples for packets currently traversing the router. This makes flow detection straightforward, but requires some model of the duration of flows in order to effectively age and finally timeout flows. Implementation of this approach is relatively simple, but has quite high state overheads. Choosing optimal parameters for the aging of flows is also not straightforward. This style of flow estimation is likely to remain suitable only at the edges of the network where the number of flows is smaller than in the core. However, flow admission control at only the edges may well be sufficient, leaving management timescale approaches to deal with the aggregates seen in the core as discussed in Chapter 4.

The second approach uses protocol specific knowledge to infer the existence of flows. Most higher level protocols used in the Internet – such as TCP and RTP – perform some form of connection setup. In contrast to connection oriented networks such as ATM, the network elements are oblivious to this connection setup process. However, by adding a small amount of protocol specific code to these network elements, it becomes possible to intercept the packets associated with connection setup. This enables flow counting to be performed, which can then be used to drive flow admission and denial as

discussed in the following section. This technique was used in the RTP-ECN-proxy discussed in Section 3.5.

In certain circumstances the second approach has benefits over the first, due to its more explicit knowledge of flow initiation and completion. This may be of use where the network element performing the estimation is known to be the single point of connection to the network – for example a firewall connecting a corporate LAN to the Internet. In more general topologies however, some timeout mechanism is still required, since traffic belonging to a flow may travel through different sequences of network elements. Ensuring that the network element implementing the admission control function has up-to-date knowledge of all the required protocols could also be a problem, unless an additional complementary approach were used to deal with currently unsupported protocols. However, it does have significant benefits in terms of the state required compared to the first approach.

The third approach is based around measurement techniques and is similar in spirit to MBAC. Estimation of the number of active flows in a router is performed using statistical information provided by the router, such as packet drops or queue dynamics. More generally, the load on the router is estimated through such statistics, and then assumptions about the traffic mix are applied to generate an estimate of the number of active flows.

This approach has a number of advantages over the two previously discussed and was used for the implicit admission control work presented in Section 3.3. It has very low state overheads, and these overheads do not usually scale with the number of flows. It also does not in general require per-flow or per-packet calculation, as information is typically aggregated in blocks over time, and calculation performed after a block has been gathered. The challenge with such methods is making good assumptions about the traffic mix, and finding relationships between the measured statistics and the number of flows.

3.2.3 Flow admission and denial

If one wishes to control the number of flows in the network, it is necessary to have some mechanism to deny access to flows. There are two principal approaches. The most straightforward and most generally applicable, is simply to drop all packets associated with a flow that has not been accepted. Although simple, this does require that a list of accepted flows must be kept; this may not be a problem, depending on the style of flow estimation in use. It also may cause network bandwidth to be wasted, since packets may successfully enter the network and traverse one or more network elements, or even ASs, before being dropped.

A more efficient solution is to signal to the endpoint generating the flow that the flow has been denied. One mechanism would be to introduce a new protocol or to extend an existing protocol, allowing a router within an AS that wished to deny a flow to signal to the flow's ingress router that packets for that flow should be dropped. While effective, such protocols introduce yet more control traffic to the Internet, and require implementation on all routers within an AS (if not throughout the entire Internet) to be useful.

A compromise between dropping the packets of a flow, and explicitly signalling denial of admission to a flow, is termed *Implicit Admission Control* [Mortier00]. This uses protocol specific knowledge to notice packets associated with flow setup, such as SYN packets in TCP, and then either drop these packets, or generate the correct protocol specific message, such as a RST packet in TCP, to prevent setup of the connection.

This approach has the advantage that it denies access to a flow early in its lifetime, preventing bandwidth being wasted, but without introducing further complexity in the network in the form of new control protocols. However, as with the second approach to flow detection discussed in the previous section, it does require that admission controllers be kept up-to-date with new or updated protocols.

3.2.4 Flow admission policy

The previous two sections have considered how to estimate the number of flows traversing a router, and how to implement admission control for flows in the Internet. These give the basic mechanisms for flow-based congestion control in the Internet. However, mechanisms must be considered in conjunction with policy to be useful; as described in Section 2.2.4 pricing is a useful way to express the required policies.

The sort of policy desirable to express here is concerned with per-flow resource management. This is principally of concern near the edges of the network, where end-systems connect to user-facing ISPs. Once traffic has been injected into an ISP and thence to the wider Internet, per-flow resource management is probably too costly to implement due to the huge number of flows and high speeds involved. Since the core is expected to have bandwidth to spare however [Sprint00], it seems that bottlenecks are most likely to occur at or near the edges of the network either where networks interconnect or where users connect to the network.

Users place different valuations on traffic associated with different services, and hence need to be able to differentiate between their flows. The network needs to be able to ensure that this differentiation is maintained to ensure

users receive the QOS they desire. At the same time, the network must attempt to efficiently and fairly share the available resource among different users. Current trends suggest that packet marking according to congestion experienced is a useful mechanism for achieving fair sharing at the same time as allowing service differentiation.

WTP [Key99a] is a mechanism that enables users to express the relative importance of their traffic by causing them to pay for marked packets received, and enabling them to express different per-flow packet mark rates to the end-system. This allows the customer to pay more for the flows they believe to be important and less for those they don't. By also controlling the number of flows entering the router, elastic protocols – principally TCP at this time – are able to remain in an operating region which provides some assurance of progress for users.

Alternatively, by using mark-proxies such as the one described in Section 3.5, the ISP can exert control over customers' resource use. By translating received marks (i.e. congestion signals) into harder congestion signals (e.g. either dropping the packet or clearing the mark for TCP, or rewriting the RR loss field for RTP), the ISP can influence users' behaviour. In conjunction with the admission control mechanisms previously described, this allows ISPs greater control over users' resource use.

In a network supporting packet marking, WTP and mark-proxies are two schemes at opposite ends of the spectrum of control. The first allows users fine grained control over their resource use, enabling them to express on a per-flow basis their desires to the network. The cost of such control is that they may have to deal with rapidly varying prices, and the possibility of being starved of access to the network. The second allows the ISP to use the congestion information received from the network to influence user behaviour, to ensure that the network can maintain a given level of service, both per-flow *and* per-user. This allows them to offer simpler pricing schemes to those users unwilling to deal with the complexity of a completely unregulated WTP scheme.

In the latter case, it might make sense to offer a 'flat rate per TCP flow' pricing scheme, or to cap the number of flows a user is allowed. It is then the task of existing protocols such as TCP, or more expressive approaches such as WTP, to share bandwidth between a given user's flows. This could also have consequences for the complexity of the ISP's billing system – rather than having to bill a user for the number of marks generated, with each mark potentially costing a different amount, users can be billed for their contracted number of flows, a much simpler problem.

This section first discussed the design of the two mechanisms involved in implementing admission control: estimation of the number of flows, and the

admission and denial of individual flows. It then discussed policies that could be implemented over these mechanisms, and the expression of these policies through pricing. The following sections now describe the simulation and implementation work carried out to validate the proposed mechanisms.

3.3 Implicit admission control

This section describes an implementation of implicit admission control carried out as a feasibility study. It addresses the questions of how existing TCP implementations and applications behave in the face of the approaches to flow denial suggested in Section 3.2.3. It also gives some tentative performance figures for an implementation; it should be noted however, that a real implementation would likely be carried out over significantly different hardware, so the system presented here is very much a prototype. Detailed performance evaluation of implicit admission control itself is carried out in Section 3.4.

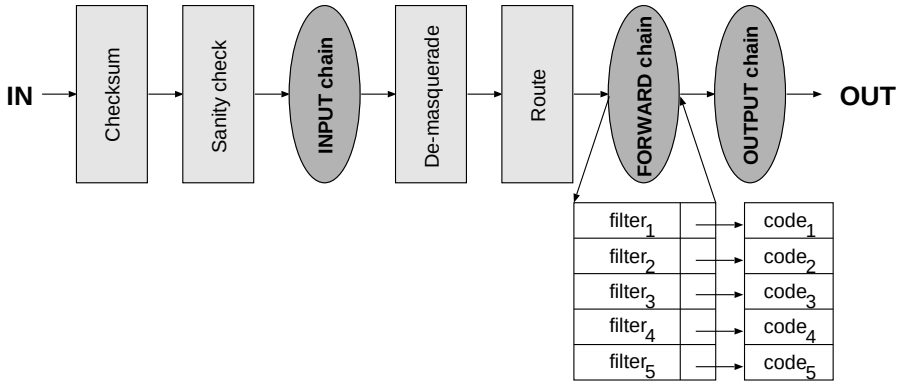
3.3.1 Linux and IPChains

Implementation was carried out using Pentium III PCs with 100 Mb/s 3Com 3c905 Ethernet cards, running the Linux 2.2.9 operating system.

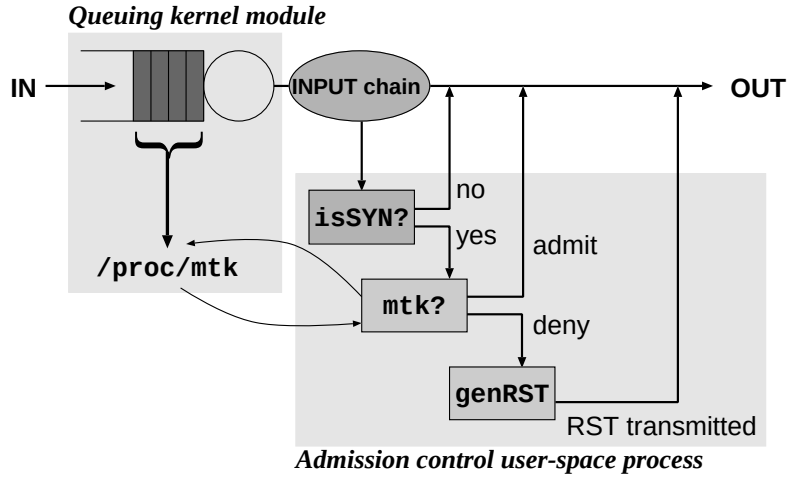
The Linux 2.2.9 operating system has a flexible mechanism known as *IPChains* for intercepting incoming packets for data gathering and modification before forwarding them on to their next hop. Incoming packets can be intercepted at three points as the kernel routes them, depicted in Figure 3.4(a). At any of these points each packet matching an installed filter can be passed to code running in user space. This code can modify the packet, or simply record its passing, and then return the packet to its path through the kernel. Eventually the (possibly modified) packet will be transmitted or dropped, according to the standard routeing process.

The implementation of implicit admission control for Linux takes the form of a lightweight kernel module and a simple user-space process, interacting as shown in Figure 3.4(b). The kernel module implements a simple 'drop-from-tail' queue along the packet forwarding path to enable measurement of queue dynamics, the results of which are made available through the `proc` file system. The user-space process imposes a filter on the packet forwarding path to enable it to receive TCP control traffic.

Upon receipt of suitable control traffic (in this case SYN or SYN/ACK packets), it calls into the *Measure* [Measure98] MTK estimator to ascertain the current admission decision. The MTK estimator, having been monitoring the queue



(a) Architecture, showing five installed filters and associated code for the FORWARD chain only.



(b) MTK/TCP implementation.

Figure 3.4: Linux IPChains.

Operating System	SYN RTO interval sequence (s) Data RTO interval sequence (s)	Total (s) Total (s)
FreeBSD 2.2.7	2.8, 6.0, 12.0, 24.0 1.4, 2.0, 4.0, 8.0, 16.0, 32.0, 64.0*7	44.8 511.4
HPUX 9.05	3.7, 10.1, 24.0 0.5, 0.5, 4.0, 8.0, 16.0, 32.0, 64.0*7	37.8 509.0
Linux 2.2.9	3.0, 6.0, 12.0, 24.0, 48.0, 96.0, 120.0*5 0.2, 0.4, 0.8, 1.6, 3.2, 6.4, 12.8, 25.6, 51.2, 102.4, 120 .0*6	789.0 924.6
NetBSD 1.3	6.0, 12.0, 24.0 1.0*11	42.0 11.0
OSF/1 3.2D	0.7, 3.0, 6.0, 12.0, 24.0 1.4, 3.0, 6.0, 12.0, 24.0, 48.0, 64.0*8	45.7 606.6
SunOS 5.5.1	1.7, 5.1, 11.8, 25.3, 52.3, 106.3, 162.6 0.9, 0.8, 1.5, 3.0, 6.0, 12.0, 24.0, 48.0, 56.3*6	365.1 434.0
SunOS 5.6	3.5, 6.4, 12.8, 25.6, 51.2 0.2, 0.5, 1.0, 3.8, 7.6, 15.3, 30.6, 61.2, 122.4	99.5 242.6
Windows 98	2.9, 6.0, 12.0 0.3, 0.6, 1.2, 2.4, 4.8	20.9 9.3
Windows NT4.0	3.2, 6.6, 13.1 0.6, 0.9, 1.8, 3.5, 7.0	23.0 13.8

x, y means that packet p_y was retransmitted y seconds after packet p_x. x*n means that n packets were retransmitted at intervals of x seconds.

Table 3.1: Measurements of packet retransmission intervals for some TCP implementations following SYN and data loss.

behaviour to keep a running estimate of the current probability of a packet being dropped, returns its decision. If the decision is to allow the flow to enter the network, the control packet is returned to the usual forwarding path and continues further into the network. If the decision is to deny access to the flow, then the packet is either dropped, or rewritten as a valid TCP RST packet, its source and destination addresses and ports swapped, and then returned to the forwarding path. This results either in the originating host detecting the loss of the connection setup packet and backing off, or in it receiving an explicit reset for that connection setup attempt.

3.3.2 TCP stack behaviour in response to flow denial

The back-off sequence when a SYN is dropped should be exponential as for normal traffic loss, but *must* last for at least 3 minutes instead of 100 seconds, the recommended value for data traffic [RFC1122]. This ensures that retransmitted SYN packets do not themselves overload the link. Table 3.1 is the result of measuring the RTO intervals for a number of TCP implementa-

tions; it demonstrates that although the mandated behaviour is not always followed, no implementation would cause the link to be overloaded with SYN packets. The decision to retry may also be taken by the application or user, rather than the protocol.

Other suggested methods of denying admission to a connection include using ICMP *source quench* and ICMP *reject: unknown protocol* messages [Kumar00]. The former has the advantage that it also allows the operator to control the throughput of active connections since it reduces the receiver's congestion window to one. The latter has the disadvantage that in addition to denying the requesting flow access, it can also cause existing flows between the same endpoints to break.

3.3.3 Application behaviour in response to TCP flow denial

Since the Web is currently the most popular use of the Internet, Netscape v4.5 on both Linux and Microsoft Windows NT, and Internet Explorer v4 on Microsoft Windows NT were used as examples of user applications dependent on TCP. In all these situations, when a TCP connection is rejected by the admission controller, the application will silently accept that it could not retrieve an object on the page unless it is the base page itself, in which case a dialog box is popped up informing the user that the page cannot be retrieved.

Further, it appears that Netscape has a timeout of approximately 30 seconds before it gives up on TCP retry attempts, whereas Internet Explorer attempts to connect 4 times for a given source port, and then repeats this for a further 4 different source ports, incrementing the source port by one each time. Consequently it does not seem that flow denial is overly intrusive from a user point-of-view.

3.3.4 Conclusions

Even a relatively heavy-weight flow estimator such as MTK uses less than 10% CPU when the machine is forwarding at line rate (100 Mb/s). As previously stated, the MTK estimator does not keep per-flow state. The results concerning TCP stack behaviour demonstrate that deployment of implicit admission control would not cause a significant increase in the amount of TCP control traffic in the network. The results concerning application behaviour demonstrate that existing applications respond satisfactorily to the denial of their flows. In all, the results from this section demonstrate that implicit admission control is feasible from the points of view of router, end-system, and user behaviour.

The following section considers the performance impact on the network and users' flows of implicit admission control using the MTK estimator.

3.4 Implicit admission control simulation

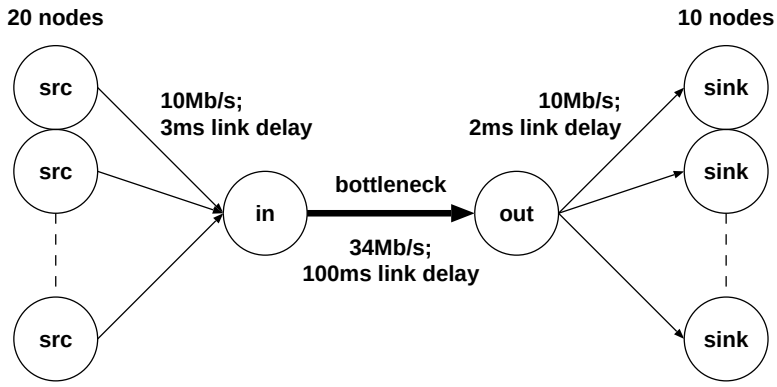
This section presents results from simulation of implicit admission control and demonstrates that it can have a significant positive performance impact, even in relatively unfavourable cases. With admission control in place, more flows complete in comparison to the no admission control case. Additionally flows complete on average faster, and a higher proportion achieve a consistent minimum goodput (the value of 10 packets per second was used) over their lifetime.

3.4.1 The Network Simulator, NSv2

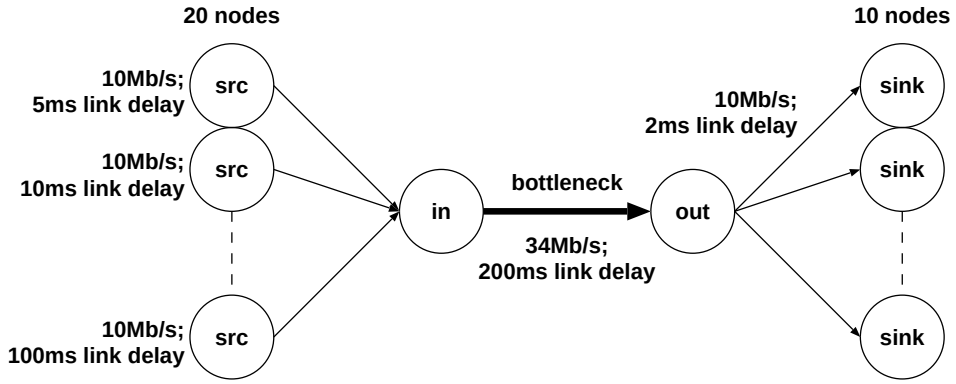
NS (NETWORK SIMULATOR V2) [NSv2] is a discrete event simulator, built specifically to simulate Internet protocols. It simulates at a per-packet level, using a C++ back-end to manage common, per-packet operations, and an OTcl [Wetherall00] front-end to construct the simulations and initial event schedule. It has been extensively used for simulating TCP variants [Fall96], and is the *de facto* standard Internet protocol simulator.

NS describes the network in terms of *nodes*, *agents*, and *links*. Nodes represent multiplexing points in the network and are connected by links. Nodes allow packets to be classified based on their addresses for transmission down links to other nodes, or to be classified based on address and port for delivery to agents. Agents represent endpoints in the network where packets are produced or consumed. They can be layered to produce layered protocols, and can have *applications* attached to cause them to generate packets based on different probabilistic distributions or other criteria. Links enable nodes to be connected and represent both the links in the network, with varying bandwidths and delays, and the queueing and scheduling policies in the network elements. They can be uni- or bi-directional.

Implicit admission control for NS was implemented by extension of the DropTail queue class (itself an implementation of a finite queue with 'drop-from-tail' policy). The DropTail class was extended to allow it to track the number of flows currently active, and to enable it to refuse new flows access to the network. The extended class, DropTailMtk, bases its admission decision on information from the MTK estimator. This monitors the queue behaviour to form an estimate of the probability that a packet will be dropped. MTK is then instructed to deny admission to flows should this drop probability estimate become too high.



(a) Constant link delays.



(b) Variable link delays.

Figure 3.5: Basic dumb-bell simulation topologies.

Two topologies were studied initially, shown in Figure 3.5. Both are 'dumb-bell' topologies, one with constant delay links, the other with links of varying delay to simulate flows with different RTTs. Although topologies such as these are too simple to satisfactorily model a network such as the Internet, it is also the case that the natural place to position systems such as implicit admission control, or the RTP-ECN-proxy discussed in Section 3.5, is at the ingress to an ISP's network, or at the egress from a stub AS. Such places are likely to be the principal bottlenecks that user traffic will see, since the core network is dimensioned to keep ahead of demand. Similarly, LAN technologies are such that users are unlikely to experience congestion within their own network. Consequently modelling the network as a 'dumb-bell' is not unrealistic from the users' perspective.

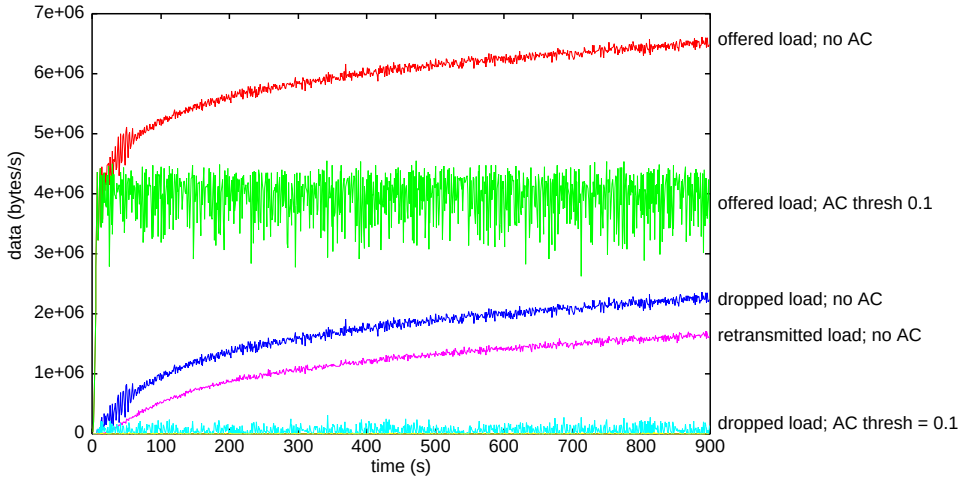
To complement the two topologies, two traffic models were also used. The first is simple constant size bulk data transfer with Poisson arrivals process, with each flow transferring 1 MB of data. This is configured explicitly to heavily overload the link. The second is a more complex model with flow lengths generated from a distribution constructed from data obtained by analysis of web server logs from a variety of sources; this mixes many short flows with a few much longer flows, leading to the heavy-tailed flow length distributions commonly reported [Paxson94a, Paxson94b].

3.4.2 Results using the simple traffic model

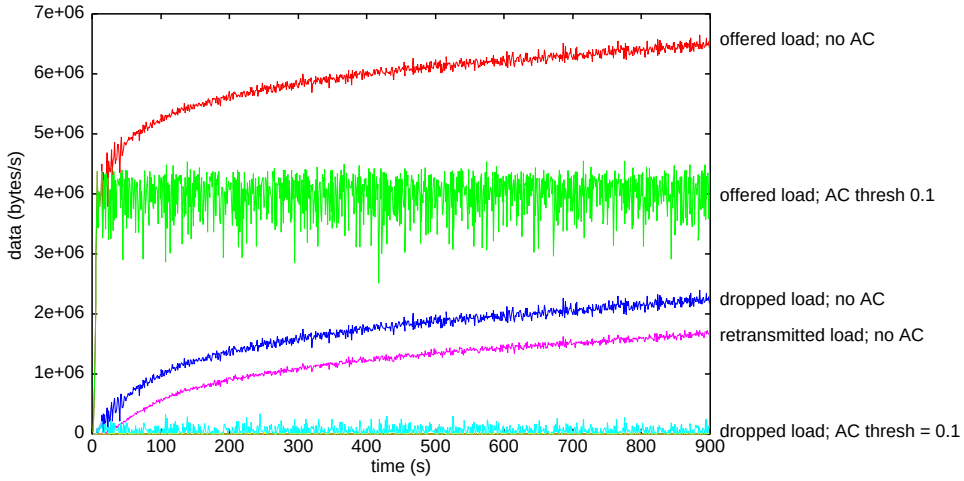
The first set of results is shown in Figure 3.6 for the simple traffic model. Figure 3.6(a) shows results for the identical link topology of Figure 3.5(a), and Figure 3.6(b) shows results for the differing link topology of Figure 3.5(b).

In the first case without admission control, the aggregate offered load is approximately 30% higher than the link capacity, leading to approximately 30% of the traffic on the link being retransmissions, due to the large volume of packets being discarded. The offered load is determined by the (congestion) window size and RTT of the TCP flows involved. Due to limitations in the way that TCP measures the RTT, the network is effectively unable to advertise the correct window size given the current load. This forces the TCP flows to continue assuming the congestion window is 30% higher than is actually the case which leads to approximately 30% offered load being discarded.

Conversely, when admission control is turned on, the offered load is kept slightly below the link's capacity, ensuring that drops and consequent retransmissions are tightly constrained. In the second case with differing link delays, the results are not significantly different. Varying the packet size between simulations has similarly negligible effect.

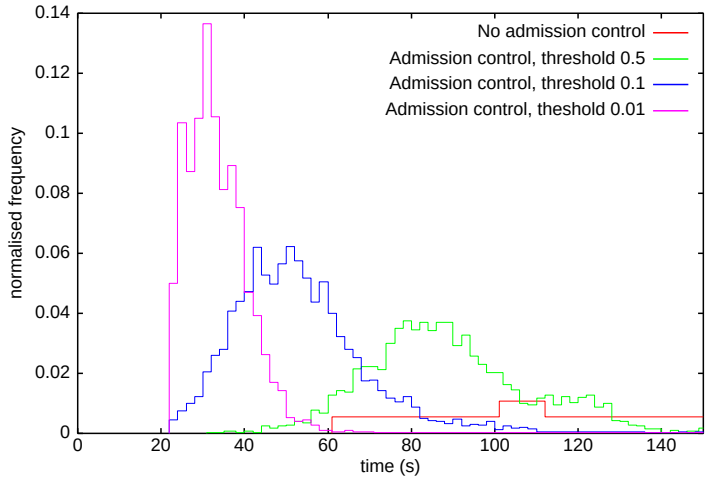


(a) Constant link delays as in Figure 3.5(a).

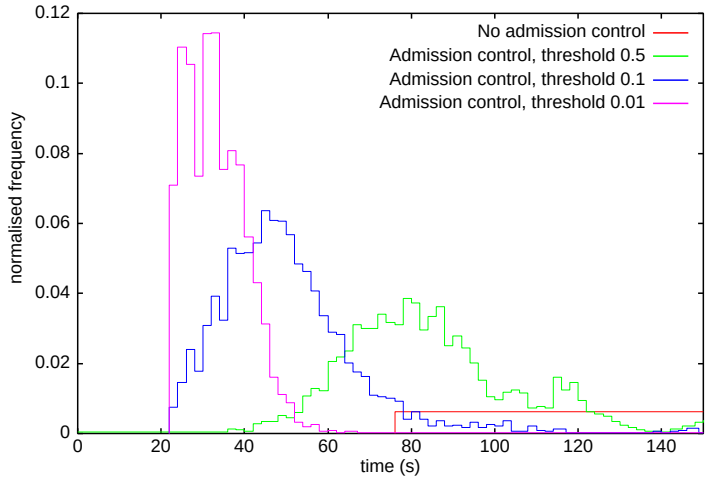


(b) Variable link delays as in Figure 3.5(b).

Figure 3.6: Offered, dropped and retransmitted load, with and without admission control. In both graphs, the retransmitted load is negligible in the admission control with threshold 0.1 case.



(a) Constant link delays as in Figure 3.5(a).



(b) Variable link delays as in Figure 3.5(b).

Frequency counts were made over buckets of 2 seconds, and normalized to the total number of flows which complete. Note that the x-axis has been truncated for clarity; due to the 'no admission control' case, it actually extends to 894 seconds.

Figure 3.7: Histograms of flow durations in the simple traffic model.

Threshold	Completed (flows)	Received (packets)	Retransmitted (packets)	Mean (seconds)	Std. dev. (seconds)
None	186	186000	687504	509.06	228.03
1.0	2493	2493000	20052	135.08	84.951
0.5	2831	2831000	9457	130.62	94.948
0.1	3323	3323000	1372	54.583	25.543
0.05	3349	3349000	562	42.563	16.146
0.01	3413	3413000	255	34.073	16.950

(a) Flow durations: identical link delays as in Figure 3.5(a), simple traffic model.

Threshold	Completed (flows)	Received (packets)	Retransmitted (packets)	Mean (seconds)	Std. dev. (seconds)
None	162	162000	699275	482.33	226.67
1.0	2476	2476000	18444	130.29	85.856
0.5	2874	2874000	9082	123.73	92.613
0.1	3365	3365000	1307	52.168	29.405
0.05	3394	3394000	617	42.747	31.098
0.01	3443	3443000	231	33.638	15.046

(b) Flow durations: differing link delays as in Figure 3.5(b), simple traffic model.

Table 3.2: The number of flows completed, packets transferred by completed flows, the total number of packets retransmitted, and the duration means and standard deviations for the completed flows. Simulations were run for 900 seconds.

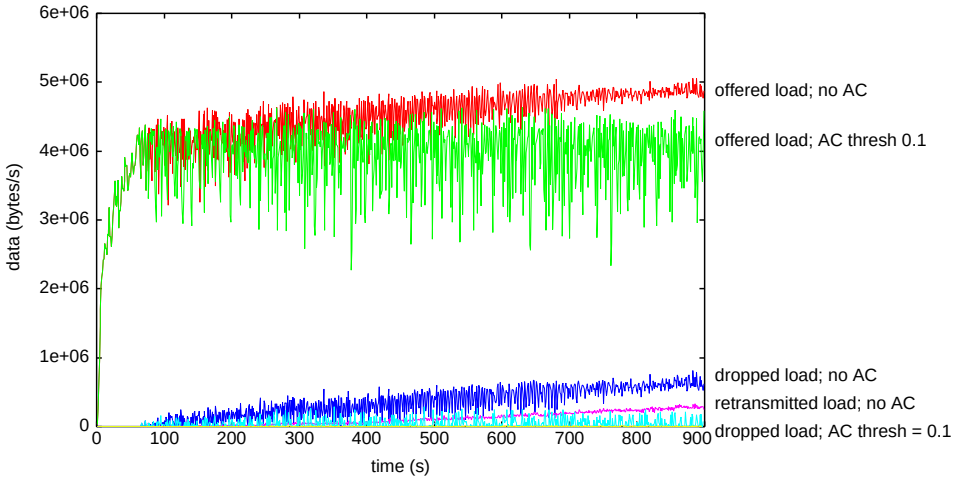
Based on these results, Figure 3.7 shows histograms of the time to successful completion for flows and Table 3.2 shows the means and standard deviations of their durations. These demonstrate that employing admission control can greatly increase the number of flows that successfully complete in a given time interval by allowing flows to complete substantially faster. Without admission control most flows do not complete, and those that do have a mean duration of 509 seconds and a standard deviation of approximately half the mean. Conversely, completion times when admission control is applied as leniently as the current estimator allows have a mean duration of 135 seconds, and a correspondingly lower standard deviation, and nearly 20 times more flows complete.

Since TCP is 'greedy,' admitted flows will attempt to use the available bandwidth in the bottleneck and the link remains at near full utilisation even with admission control in place. This is shown by the offered load results in Figure 3.6. In conjunction with those, the results shown in Figure 3.7 and Table 3.2 demonstrate that many applications will achieve higher utility if admission control is applied. Users may be prepared to wait for 1 minute for a large download to complete; they are less likely to be prepared to wait for 15 minutes. In effect the results demonstrate that it is possible for the network operator to tune the network based on users applications' requirements, in order that users receive higher utility.

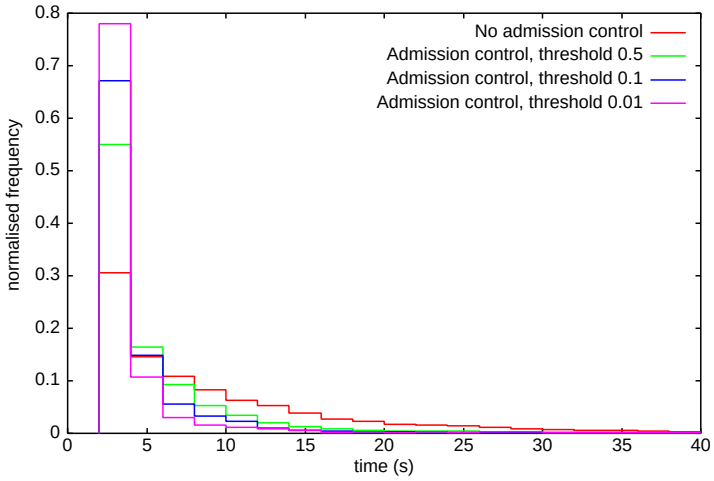
3.4.3 Results using the complex traffic model

The next set of results are for the web-log based complex traffic model. As can be seen in Figure 3.8 admission control has a similar effect as with the simple traffic model: the offered load is kept at or slightly below the link capacity when admission control is applied, but continues to rise when no admission control is in place. The drops and retransmissions also exhibit similar behaviour as with the simple traffic model. However, the flow duration histogram in Figure 3.8(b) and the table of the mean and standard deviation of flow durations in Table 3.3 show that fewer flows complete successfully with admission control in place.

Table 3.3 shows that flows are completing faster and with more tightly controlled durations when admission control is applied. However, fewer flows complete successfully which appears discouraging. Examination of the number of packets received reveals an explanation. When admission control is applied, approximately the same number of packets are successfully received, suggesting that link utilization remains the same. Without admission control the proportion of retransmissions, for the longer flows in particular, rises as the existing longer flows lose out to the shorter flows in slow-start. When admission control is applied, the longer flows are able to complete since the



(a) Offered load, drops and retransmissions, without admission control and with an admission threshold on the target loss probability of 0.1. Again, retransmitted load is negligible in the admission control with threshold 0.1 case.



(b) Flow durations, where the frequency count has been made over buckets of 2 seconds, and normalized to the total number of flows to complete.

Figure 3.8: Results for the complex traffic model with the topology shown in Figure 3.5(a).

Threshold	Completed (flows)	Received (packets)	Retransmitted (packets)	Mean (seconds)	Std. dev. (seconds)
None	17180	1842083	63269	15.350	35.273
1.0	14027	1869053	10325	10.712	31.099
0.5	13389	1855134	5738	9.5144	28.359
0.1	12070	1860911	1146	7.2418	21.878
0.05	11778	1825970	859	6.7397	20.311
0.01	10915	1771488	313	5.8935	17.947

Flow durations: identical link delays, complex traffic model. The simulations were run for 900 seconds, using the complex web-cache log based traffic model. The topology shown in Figure 3.5(a) was used.

Table 3.3: Number of flows completed, packets transferred by completed flows, total number of packets retransmitted, and mean and standard deviations of the durations of the completed flows.

Threshold	Completed Flows	'Good' Flows	(%)	'Bad' Flows	(%)
None	15219	4595	(30%)	10624	(70%)
1.0	12065	7255	(60%)	4810	(40%)
0.5	11427	7968	(70%)	3459	(30%)
0.1	10192	8591	(84%)	1601	(16%)
0.05	9900	8634	(87%)	1266	(13%)
0.01	9157	8618	(94%)	539	(6%)

Only flows that started after the first 100 seconds had passed were counted, in order to remove initial transient behaviour.

Table 3.4: The number of completed flows with the number that met a target of 10 packets per second over their lifetime ('good' flows), and the number that failed to meet this target ('bad' flows).

excess short flows are unable to enter the link and cause the long flows to experience excessive loss.

Examining Table 3.4 provides further insight. Picking a target of 10 packets per second per flow as a measure of 'useful' goodput, the application of admission control nearly doubles the number of 'good' flows that complete. This suggests that a large number of the extra flows that manage to complete with no admission control are receiving very low transfer rates, and are hence of less use.

This table also suggests a manner in which the operator could set the threshold. One might choose a target throughput and then adjust the admission threshold to achieve it, the particular value depending on the traffic mix and on the level of service the operator wishes to provide for its customers. Better estimators might give one a more controllable parameter with greater dynamic range – a weakness of MTK in these circumstances is that its maximum

threshold is 1.0, which leaves quite a large gap in system behaviour between no admission control and admission control at its most lenient. However, MTK does appear to give a reasonable range of values for the operator to tune, which is encouraging given that the link is only experiencing overload of approximately 20%.

Additionally, appropriate use of dynamic pricing schemes as suggested in Section 3.2.4 might give the operator some indication of how to set the admission parameters. Alternatively, more static pricing schemes simply require that the number of flows be limited. This could be achieved by adjusting the threshold in response to an estimator for the number of flows.

3.4.4 Conclusions

This section presented results from an implementation of implicit admission control for the NS simulator. The results show that with admission control in place flows experience lower completion times when the link is overloaded. The data presented demonstrate that even for relatively elastic protocols such as TCP there is benefit to be had from limiting the number of flows competing for a congested resource. There is also evidence to suggest that admission control may be of benefit even where the resource is not continuously overloaded, in that doing so may provide greater fairness in resource allocation to flows and users.

The following section discusses implementation of a different approach to control timescale traffic engineering in the Internet: an RTP-ECN-proxy.

3.5 An RTP-ECN-proxy

This section presents the design of, and results from, an implementation of an RTP-ECN-proxy for Linux. This enables applications that are not ECN-aware to be made aware of the congestion information being given to them as ECN marks. The proxy was implemented for the RTP protocol and utilises IPChains as shown in Figure 3.9, more dynamically than the implicit admission control implementation.

The proxy watches for the initial port negotiation procedure of the RTSP protocol to enable it to see which ports will be used for the RTP conversation. Having discovered this information, it imposes a filter on the forwarding path to enable it to capture incoming ECN-marked packets, and outgoing RRs (RECEIVER REPORTS). The proxy can then account incoming ECN-marked packets to the relevant flow.

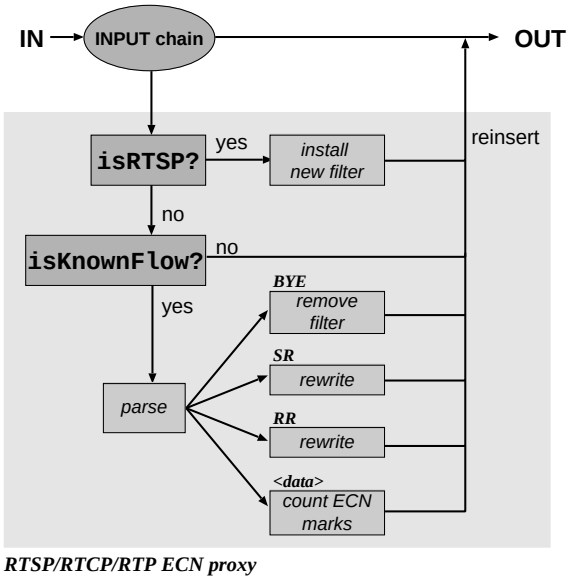


Figure 3.9: Linux IPChains RTP-ECN-proxy implementation.

When the receiver sends an RR (containing packet loss statistics) back to the sender, the proxy captures the RR, and rewrites the packet loss field to take account of the ECN-marked packets it has seen on that flow. This then causes the sender to adjust its rate as if the marked packets had been dropped. A more complete implementation would extend the RTP protocol to allow the RR to separate the 'packet lost' and 'packet marked' information, enabling the sender to make a more intelligent rate adjustment decision.

3.5.1 Application behaviour in response to the proxy

The RTP-ECN-proxy described in Figure 3.9 was tested using the VIC application [Vic01], video-conferencing software that uses RTP. VIC was extended to respond to the congestion signals from the network in the form of the loss information contained within RRs. It responds according to a simple additive-increase, multiplicative-decrease scheme to attempt to maintain the frame rate at the expense of image resolution, whilst altering the target transmission bandwidth in line with the congestion signals from the network contained in the RR.

The proxy counts the marked packets received by the receiver, and rewrites outgoing RRs from the receiver to the transmitter. The transmitting VIC application can then modify its behaviour based on congestion in the network without either the transmitter or receiver requiring modification to deal with ECN marks.

When used in conjunction with TCP admission control schemes, this might be considered one way in which QOS could be maintained for real-time traffic and existing TCP flows during times of congestion. Rather than allow even more best-effort TCP traffic to enter the network, the threshold for the admission control system could be modified to deny access to more flows. Existing flows would not be harmed by the onset of congestion collapse, and real-time traffic would be able to maintain a reasonable frame rate at the expense of resolution (or other suitable trade-off, dependent on the application and user preferences).

3.5.2 Conclusions

As with the implicit admission control implementation, CPU use for the RTP-ECN-proxy is low, so it is to be hoped that edge routers where such schemes might be deployed have sufficient spare computing power. Memory use for the proxy is approximately 100 bytes per flow, so such a system should scale to a reasonable number of flows.

3.6 Summary

This chapter discussed traffic engineering at control timescales within the Internet. It began by demonstrating the need for control timescale traffic engineering caused by TCP's unfairness and eventual congestion collapse due to too many flows contesting a restricted resource. It then discussed the implications of performing this sort of traffic engineering, and went on to consider design issues and possibilities. Finally, results to support the claims made were presented, and demonstrated that per-flow admission control and mark-proxies can be used to implement control timescale traffic engineering in the Internet, and that doing so has benefits for users.

The following chapter now discusses management timescale traffic engineering in the Internet.

Chapter 4

Management timescale traffic engineering

This chapter discusses management timescale approaches to traffic engineering. It describes the various components involved in controlling the network at management timescales, concentrating on the protocol used for inter-AS routeing, BGP. Current uses of BGP and how it may be extended through the use of path and community attributes are then described. This is followed by the design of a new path attribute for BGP to enable advertisement of price with route updates. Finally, results from an implementation of this within a new BGP simulator are discussed.

The following chapter will discuss the integration and deployment of the techniques presented in this and the previous chapter.

4.1 Scope

This section discusses the scope of this chapter. It describes the various components of routeing in the Internet, notes their application to management timescale traffic engineering, and their relation to data timescale traffic engineering as presented in Section 2.2.4. The following section then concentrates on inter-AS routeing and BGP, and modifications to BGP for implementing management timescale traffic engineering.

4.1.1 The structure of Internet routeing

There are three principal components to routeing in the Internet. All disseminate information concerning IP *prefixes* and the next-hop to which traffic

matching a particular prefix should be sent. An IP prefix consists of an IP address and a prefix length which states how many bits of the address are significant. The routeing components can be split based on the origination of the prefixes advertised and the receivers of the adverts.

As previously noted in Section 2.4, the initial split can be made based on whether or not the prefixes advertised originate within the AS. Prefixes originating within an AS are advertised via the IGP (INTERNAL GATEWAY PROTOCOL). These protocols are typically restricted to the local area and examples include OSPF and ISIS. Conversely, EGPs (EXTERNAL GATEWAY PROTOCOLS) are used to advertise prefixes originating outwith the AS, and the only widely deployed EGP is BGP.

A further separation can be made based on the use to which BGP is put. If it is used to advertise prefixes across administrative boundaries (i.e. *between* ASs), it is referred to as EBGP (EXTERNAL-BGP). Conversely, if it is used to re-advertise externally learnt prefixes within an AS, it is referred to as IBGP (INTERIOR-BGP).

This chapter concentrates principally on BGP in its EBGP form, although the application of IGPs and IBGP to management timescale traffic engineering is briefly discussed in the following subsection.

4.1.2 Routeing and traffic engineering

Traffic engineering at management timescales can be implemented via routeing protocols. Routeing protocols currently aim to provide connectivity through the Internet, and pay little attention to the load. Load balancing policy can be applied via the routeing protocol by configuring different routes for particular prefixes, but this is managed manually by the operator. A contribution of this dissertation is to enable routeing protocols to distribute traffic throughout the Internet, and between ASs in particular, so that congestion is avoided. The end result should be that traffic is smoothly distributed throughout the Internet, resulting in higher performance for users.

As in the previous chapter, the unifying approach to traffic engineering taken throughout this dissertation is that of pricing. In the context of management timescale traffic engineering this is applied by using the currently advertised price for a network link as the routeing metric for that link. As will be discussed further in Section 6.2, such calculations could well be integrated with the dynamic calculation of parameters for the admission control module discussed in Chapter 3.

Intra-AS pricing refers to the use of pricing within an AS to control the management of traffic within that AS. This involves the IGP such as OSPF or ISIS

measuring the load of the links over which it runs and using these measurements to calculate a price. This price can then be used to calculate values for the protocol's routeing metrics. The intent of doing so is to cause traffic to be distributed efficiently and automatically throughout the AS, much as was attempted by the HELLO protocol.

The HELLO protocol and its use of RTT as a metric was discussed in Section 2.4.3; its fundamental problem is that the RTT is not very suitable as a measure of congestion for this purpose. A better alternative is available through packet marking schemes as discussed in Sections 2.1.2 and 2.2.4. Marking strategies and pricing schemes that damp the oscillations experienced with RTT can be implemented. Also, since the majority of intra-AS routeing protocols are link-state unlike the HELLO protocol, more explicit information is available for the shortest path computation.

The price in this context could equally be viewed simply as the calculated load metric [Fortz00] as it is unlikely to be transformed into a charge since settlement within the AS is generally unnecessary. In conjunction with the generally better convergence properties of link-state protocols over path- and distance-vector protocols, it thus seems reasonable to suppose that intra-AS pricing could be implemented so as to avoid oscillatory behaviour. Consequently, the work described in the remainder of this chapter concentrates on the use of pricing for management timescale traffic engineering within BGP, a path vector protocol.

4.1.3 Relation to existing traffic engineering schemes

Packet marking is usually proposed as a mechanism for data timescale traffic engineering, generally by performing congestion control based on received marks. The problem it addresses is that of communicating to end-systems the current state of the network so that each end-system may decide whether or not it should inject a packet into the network at this time. As discussed in Section 2.2.4, with suitable functions for marking and for translation of received marks into prices, the network and end-systems can attempt to optimize network usage.

Optimization from the operator point of view is concerned with the maximisation of revenue achieved through payment by end-users for marks. From the user or end-system point of view, the optimization problem aims to ensure that users receive maximal utility from the network. Kelly *et al* [Kelly98] show that for certain utility functions these problems are strongly related: they can be simultaneously solved by setting prices such that users achieve a share of the contended resource (i.e. the network) proportional to the price that they are willing to pay.

The aims of management timescale traffic engineering differ in a number of important ways. Management timescale traffic engineering aims to make the distribution of traffic through the network more efficient given the traffic that has already entered the network. Management timescale traffic engineering does not deal with transient overload in the network; this is dealt with through data timescale traffic engineering approaches and prevented through control timescale traffic engineering approaches. In this sense the problem of calculating suitable prices is much simpler: there is no 'optimum' price to be reached. Rather, the concern is with the stability of the calculated routes and the smoothness of the traffic distribution.

Furthermore, prices in the context of this chapter are only directly distributed between BGP peers and not to end-systems. The following section discusses the use of BGP for management timescale traffic engineering in more detail.

4.2 Inter-AS routeing and pricing

Inter-AS routeing is concerned with the prefix-based routeing of large aggregates of traffic through the network, subject to policies imposed by ISPs. This section describes use of BGP to achieve this, and proposes a new path attribute for advertising a per-AS price.

The principal aim of inter-AS pricing is to give greater control over traffic distribution to ISPs offering and receiving transit services. Those who desire higher quality service, both for transmitted and for received traffic, should have some mechanism to express this. Conversely, those ISPs providing transit services should have some mechanism that allows them to encourage or discourage customers (i.e. other ISPs) from routeing traffic toward them. Given an operational network, the major cause of service quality degradation is congestion, so it seems reasonable that these prices should be based ultimately on the congestion in the network.

4.2.1 Path and community attributes

BGP is used to communicate external connectivity information throughout the Internet. As the only available mechanism, it is also used by ISPs to express policies they desire. In the simplest case this involves filtering BGP UPDATEs so that routes are only advertised to the correct peers, and so that adverts are only believed from peers with whom the ISP has a suitable SLA.

More complex policies are then expressed through the use of BGP *path attributes*. These are attributes associated the advertising router associates with the set of prefixes contained in a given advertisement. Where multiple routes

for a prefix are available, the receiving router then bases its choice of 'best route' on the values of these path attributes. A commonly seen example is the use of the length of the AS-PATH attribute, the number of ASs listed in the AS-PATH attribute associated with an advertised prefix. Given a choice, a router will generally prefer the route with the smallest number of such hops.

Further control is available through the *community* attribute. This is a path attribute consisting of four octets, the first two of which are the AS number of the advertising router by convention. The values of the other two octets then encode either IETF standardised values, or have semantics defined bilaterally between the advertising and receiving ASs. Using this mechanism, an advertising AS can instruct a co-operating receiving AS to prefer one route to another, or to perform some operation on a route before re-advertising it. In this way more complex policies such as multi-homing are currently implemented [RFC1998].

4.2.2 Current use of BGP

The previous subsection described the features BGP provides to implement policies desired by co-operating ISPs. Unfortunately, since it was not designed for doing so it is not well suited to this task, leading to ad hoc solutions. For example, one common technique is AS *pre-pending*, where the router prepends its AS number multiple times to the AS-PATH for a route. Thus the router attempts to dissuade the receiving AS from using that route. Examination of a sample BGP routeing table from KPN-Qwest suggests that this technique is used on approximately 5% of valid routes (approximately 17,000 out of 306,000), or 8% of 'best routes' (approximately 8,000 of 100,000) implying that a more explicit mechanism for achieving this effect would be useful. Furthermore, an AS might wish to instruct a neighbouring AS to perform AS pre-pending based on the source or destination of an advert and can only currently do so via community attributes.

Since there is no means by which to co-ordinate the implementation of such policies, conflicts might be introduced that could lead to permanent route oscillations [Griffin99, Labovitz01]. There have been attempts to codify such policies through the use of RPSL (ROUTEING POLICY SPECIFICATION LANGUAGE) [RFC2622]. It is intended that policies encoded in RPSL may then be statically checked to reduce the possibility of policy conflicts leading to route oscillation. However, even were such codifications complete, they does not make it any easier to express desirable policies such as multi-homing, and in any case it is not possible to statically detect all such policy conflicts. Additionally, there is no way to express more dynamic policies, such as 'use route R_1 unless load is higher than l, in which case use route R_2' as might be desirable for traffic engineering purposes.

4.2.3 The *price* path attribute

Consequently, a more dynamic mechanism by which to express such policies is required. One such mechanism is to associate a load-based price with prefixes advertised to peers. Where a choice between routes for a prefix exists, peers can then make the decision based on more dynamic policies such as 'pick the cheapest route.' This implements management timescale traffic engineering within the routeing protocol.

The natural mechanism to implement inter-AS pricing for management timescale traffic engineering is as a new path attribute for BGP. This dissertation proposes such a path attribute and suggests that it be *optional* and *non-transitive* (i.e. not all BGP implementations need support it, and it need not always be communicated to peers). This preserves compatibility with prior versions of BGP, whilst enabling incremental deployment. Noting the conceptual separation in Section 1.1 between *pricing* and *charging*, inter-AS pricing splits into three parts:

Measuring congestion. This will be done by the routers in the AS, and the information made available through mechanisms such as SNMP (SIMPLE NETWORK MANAGEMENT PROTOCOL). Congestion can be measured in a variety of ways such as RTT, packet drop rate, and packet mark rate.

Calculating the price. Based on the congestion measured per-link, a price for the node can be calculated. This represents some application of policy by the owning ISP to the congestion measured on its links, for instance in terms of the QOS they wish to be able to provide.

Charging for use. Finally, based on the calculated price the ISP can calculate a charging rate for the link. Different rates can be calculated for different customers and advertised in the relevant UPDATE messages. This enables the owning ISP to apply policy based on the customer so that they can influence the customer's choice of route for traffic.

The division of inter-AS pricing in this way achieves a number of goals. The most important from a technical point of view is that ISPs now have some rational mechanism to choose between possible routes when such choice is available. More nebulous effects include the easing of network management since usage based charging in this way gives a basis for automated settlement of bilateral peering arrangements; potential structural changes to the network, both technical and economic (discussed in more detail in Section 5.5); and the possibility for ISPs to influence the route taken by traffic destined for them. In this way ISPs can begin to offer differentiated service in the Internet which can include some form of statement about the treatment of traffic in networks other than their own.

Although there are a number of possible network metrics on which to base the price, the work described in this dissertation concentrates on the use of packet marks for this purpose: routers will monitor the congestion that they are experiencing, and calculate a price based on this. It is believed that the mark rate is a useful measure of congestion since it takes into account both the remaining capacity and queueing delay on the link, but should not (given a sensible mark scheme) induce oscillatory behaviour. The price will thus be based on, and usually proportional to, the congestion that the router is experiencing.

The price is calculated per-node based on the per-link load, and not per-link, since a price will be advertised on both IBGP and EBGP sessions. Although it may seem more natural to have a per-link price, there are a number of reasons why this is not appropriate.

First, BGP does not have a natural notion of a 'link.' Rather, BGP sessions are transported over TCP and so use the underlying IP network and IGP routeing information to effectively create 'virtual links' between every pair of peering routers. Consequently, two apparently different links (i.e. two prefixes with different `NEXTHOP` attributes) may overlap. Furthermore, depending on lower layer configuration, traffic apparently destined for the same `NEXTHOP` may actually traverse different links.

Second, the price is used in both IBGP and EBGP peering sessions. When advertised in IBGP sessions, it is not mapped to a charge, but is essentially used to generate a per-AS price. When advertised in an EBGP session, it is mapped into a charge, but is attempting to advertise a per-AS charge, rather than a per-link charge. This makes settlement in situations where two ASs connect in multiple locations simpler, since the same charge will be advertised to both.

Finally, the AS receiving the traffic remains at liberty to use other mechanisms such as `MULTI-EXIT-DISCRIMINATOR` to attempt to influence the transmitting ASs choice of egress link, and to distribute traffic within its AS as it sees fit.

Before the calculated price is advertised to the peer through the `UPDATE` message, it is transformed according to local policy into a *charge*. As previously mentioned, this enables the operator to apply policy to influence the influx of traffic to their network. Finally, the charge is advertised to the BGP peers through the `UPDATE` message. Charges received from peers are then used to calculate `LOCAL-PREF` values, allowing policy to influence the efflux of traffic from their networks[1].

[1] An alternative available in most BGP implementations is to map the received charges into the *administrative weight*, a value associated with a route according to some per-router policy. This value is not advertised by the router, and indeed, is not documented in any current RFC.

4.2.4 Settlement

Having received price path attributes associated with prefixes, operators can choose their best routes on the basis of the charges they will incur. Hence there must be a basis for *settlement*, where the charges associated with neighbouring prefixes are transformed into bills. Although there are a variety of metrics on which settlement might be based, this dissertation proposes use of the traffic volume exchanged between peers.

Traffic volume has a number of advantages: it is straightforward to understand and to measure; it is generally slowly varying between ASs, allowing operators to make relatively accurate predictions about future bills; and many operators already have to collect such information in order to police the SLAs into which they have entered.

Of course, scope exists for more complex settlement schemes. For example, if suitable feedback could be arranged, settlement might be performed based on the number of packets marked. Although this links the final bill more closely to congestion (since charges will not be levied unless congestion is occurring and hence packets being marked), such a scheme is more complex to understand and predict, and requires more infrastructure to support.

The following section considers the detailed design of the price path attribute.

4.3 Design of the price path attribute

This section discusses design issues surrounding the price path attribute. It considers bases for calculation of the price, the properties and use of the price path attribute and potential problems with its introduction and use. The following subsection considers the calculation of prices to be advertised.

4.3.1 Price calculation

As noted in Section 4.1.3, pricing in this context has different aims to pricing in other contexts, particularly data timescales. Approaches to data timescale pricing typically address end-to-end congestion issues, and express prices to users and user applications so that they may make informed decisions about their use of the network. As a consequence, the price is being used by the network to solve the optimization problem of maximising user utility.

Since charges are advertised by an optional non-transitive path attribute, it is immaterial whether they are mapped into weight or `LOCAL-PREF`.

Pricing in the context of management timescales aims to reduce the manual intervention required to manage SLAs, enabling greater automation and to allow operators to make informed routeing choices for aggregates of traffic, where such choices exist. These choices may be driven by the type of network service the operator wishes to offer, and hence driven indirectly by end-user desires, but end-users do not directly influence such routeing decisions.

As a consequence, prices calculated for traffic engineering at management timescales have different requirements to prices calculated for data timescales: the price itself should stabilise, but the key point is that the BGP routeing tables should converge. Operators may be able to deal with oscillating prices since actual settlement will not be performed continuously, but network engineering considerations do require that BGP has no worse stability properties than at present.

The first consideration is the information available at a node for calculating the price. As described so far, a node has knowledge of the load its links are experiencing and the charges advertised to it from its peers. The price, p_i, at a node, N_i, may thus be viewed as a function, $p_i = p(l_i^j, c_j^i) \quad \forall j \neq i$ where l_i^j is the load between nodes N_i and N_j, and c_j^i is the charge node N_j advertises to node N_i. Note that although these parameters are all time dependent, this is not made explicit in the notation. Denoting the load at a node N_i by $l_i = \Sigma_j l_i^j$, there are then a number of reasonable constraints that exist concerning the price:

1. $p_i > 0$, since a negative price is nonsensical;
2. $\frac{d}{dl_i}(p_i) > 0$, since the price should rise as load rises;
3. $\frac{d^2}{dl_i^2}(p_i) \leq 0$, to make the price less sensitive to load changes as load, and hence the price, increases.

The first two constraints are trivial, but the third bears some explanation. As the network becomes excessively congested, changing the selected best route will have a progressively more disruptive effect, and so should become harder to do.

The reason is that route stability is likely to be a more important constraint than maximising the revenue generated by these price-based mechanisms: operators have other means to generate revenue, and there is not much that the routeing protocol can do to deal with a network which is simply overloaded. At this point, measures such as admission control and the end-to-end congestion control mechanisms of the transport protocols must play their part to reduce congestion. Although this may be implemented through pricing visible to end users, it is not within the remit of the routeing protocol to calculate or advertise these prices.

BGP_PRICE_CALC(running at router R_i)
1 record number of marked packets l_i
2 calculate price p_i, based on load and received charges, $c_j^i, j \neq i$
3 calculate LOCAL-PREF values
4 calculate best routes
5 FOR-EACH IBGP session:
6 advertise routes with price p_i to internal peers $R_j, j \neq i$
7 FOR-EACH EBGP session:
8 transform price p_i into a (session-specific) charge $c_i^j, j \neq i$
9 advertise routes with charge c_i^j to external peers $R_j, j \neq i$

Figure 4.1: Basic pricing algorithm.

A further constraint not considered here is that the revenue that the operator generates from the network should be positive. This may not require that traffic based charges discussed here actually render a net positive result, since operators may generate revenue through other means, but is certainly something that should be considered in a real deployment. Similarly, if traffic based revenue is to be used to recover sunk costs, this factor might be taken into account when calculating the charge.

In summary, the price should be positive, increase as the load on the router increases, and be related to the charges advertised by external peers. Both the price and the policies that might be implemented using it should not damage BGP stability, and at a given level of load on the network the price should itself converge to a stable value.

4.3.2 The algorithm

The algorithm used is given in Figure 4.1 in more detail. This is run at each BGP peering router, both ingress and egress. Section 4.3.4 discusses IBGP behaviour, and interaction with the IGP is considered in Section 6.2. In step 1 the router in question monitors the number of packets it marks, and the price is calculated in step 2 based on the measured load and on the received charges from peers. The LOCAL-PREF values are calculated in step 3 and then the best routes are calculated in step 4.

Depending on whether the peering session is an IBGP or EBGP session, step 8 then uses the price, p_i, to calculate the charges, c_i^j. Finally the routes are advertised to peers, either with unmodified prices, p_i in step 6, or with the charges, c_i^j in step 9. In this way the routers within the AS are instructed how they should distribute traffic within the AS, and routers in adjacent ASs can make decisions about whether to use this AS for transit. Automated settlement between peers and hence between ASs can thus be achieved by suitable

monitoring of traffic between routers and settlement of any difference in the load-charge product.

4.3.3 Expression of policy

Mechanisms are of little use without some understanding of the policies that may be expressed. With the price path attribute discussed above, an AS may express policy for outgoing traffic through 'best' route choice based on received adverts; and for incoming traffic through the value of the charges it advertises to its neighbours. Such policies may be separated into two categories: static and dynamic.

Static policies include the sorts of BGP configurations commonly found today. The standard example would be 'always choose AS_i over AS_j' for a given prefix. For such policies the price path attribute acts purely as an accounting mechanism, simplifying the construction, parameterization and settlement of SLAs; it plays no direct part in the distribution of traffic through the network, and hence should not affect routeing stability.

Dynamic policies are more interesting and enable more expressive semantics but are harder to understand. Perhaps the most obvious such policy would be 'pick the cheapest route.' More complex policies could be implemented if the ISP could measure the load neighbouring ASs were experiencing.

Using such load measurements, the ISP could implement policies such as 'pick the highest quality route,' 'pick the cheapest route j such that $c_j^i l_i^j < C$' or 'pick the highest quality route j such that $c_j^i l_i^j < C$.' The implementation of such remote monitoring facilities is not covered here; route server *looking glasses* already allow queries of the routes available to particular destinations, and these might be extended if the facility was considered desirable[2].

Given suitable further extension to BGP, it would be possible to be even more flexible in the expression of policy. For example, it might become desirable to be able to chose a route based not on the destination address of the packet, but on a combination of the destination and source addresses; this is something currently explicitly forbidden by the BGP standard. However, such extra flexibility is likely to have complex interactions, and could significantly affect routeing stability. Consequently, it is not discussed further in this dissertation, but rather left as an area for future study.

[2]In fact, companies offering such Internet performance measurement services now exist [Keynote01, Matrix01].

4.3.4 IBGP behaviour

As previously stated, BGP is used to perform two functions: as EBGP, to receive and advertise prefixes outwith the AS; and as IBGP, to readvertise externally learnt prefixes within the AS. The protocol behaves identically in both cases with one exception: the `LOCAL-PREF` path attribute is trusted in the IBGP case, but not in the EBGP case.

By similar reasoning, routers need not apply local policy to change the price into a charge when it is to be advertised on an IBGP session. They merely advertise the price as measured, and the receiving router can apply suitable policy through calculation of its own price. By treating separately the prices learnt through EBGP and IBGP sessions, routers can attempt to arrive at a 'per-AS' price using the information gleaned from IBGP sessions about the load on the entire AS. Additionally they may use the information learnt concerning the load in neighbouring ASs, which may have been transformed according to SLAs or other inter-AS policy requirements.

The separation between IBGP and EBGP information and between the price and the charge may also be used to improve the stability and resource usage of the protocol. Fluctuations within an AS due to IGP oscillations or fluctuating traffic demand may be hidden from the rest of the Internet by using the price to charge mapping as a damping factor.

As described the algorithm assumes the standard configuration of a full-mesh of IBGP connections within the AS. This does not scale well as the size of the AS increases, so two mechanisms have been developed to remedy this scalability problem: AS confederations and route reflection.

AS *confederations* [RFC3065] effectively divide a large AS into a set of smaller sub-ASs. A sub-AS contains the standard full-mesh of IBGP connections; between sub-ASs BGP now behaves as EBGP, although such sessions retain the properties of IBGP sessions in that attributes such as `LOCAL-PREF` can be trusted. *Route reflection* [RFC2796] involves having some routers designated as *route reflectors* and some as *route reflector clients*. Route reflectors are allowed to readvertise routes within the AS whereas the clients can only advertise routes to the reflectors. The proposed price path attribute is affected by neither of these modifications.

4.3.5 Discussion

There are a number of potential issues with the price path attribute that bear some discussion. They can be split into four: route disaggregation; route oscillation; price oscillation; and interaction with pricing applied at other layers. These will be dealt with in turn.

Route disaggregation is a problem affecting the scalability and resource usage of BGP. It occurs since operators desire greater control over the traffic they carry. Since routeing is currently performed on a longest prefix match basis, the only way for providers to exercise finer grained control over traffic aggregates is to disaggregate prefixes in order to separate traffic aggregates. This allows them (and their neighbouring ASs) to choose different best routes for these smaller aggregates. Such disaggregation has traditionally been avoided at all costs, as it can increase both the number and size of UPDATE messages, and perhaps more importantly, it increases the size of routeing tables.

However, methods to automatically create equivalent forwarding tables containing the (provably) minimum number of prefixes exist [Draves99] and can yield a 45% reduction in the number of prefixes in the forwarding table. Simultaneously, the computational power and memory capabilities of routers have dramatically increased. These facts coupled with the fact that disaggregation of this nature is occurring in any case due to multi-homing suggest it is not as serious a problem as it might at first appear.

Route oscillation is undesirable for a variety of reasons: it increases the routeing protocol's resource usage; it can cause substantial variation in the path and hence network characteristics that end-to-end traffic experiences; and under heavy load conditions it makes network management more difficult since large quantities of traffic may be moved between links.

Section 4.5 will discuss how dynamic routeing based on advertised prices may be implemented so as not to increase the potential for oscillatory behaviour of the protocol. In fact, it can also be speculated that it might actually cause a decrease.

It is already known that BGP can suffer from oscillatory behaviour [Labovitz97, Labovitz98, Griffin99, Labovitz00]. Previous work on the stability of BGP suggests that instability results due to incompatible routeing policies [Griffin99,Varadhan00,Labovitz01]. Incompatibilities arise due to the application of different LOCAL-PREF values to routes from different providers, removing the monotonically increasing metric required to guarantee stability and provided by the AS-PATH length.

The addition of a globally valid metric (i.e. the price) that will be monotonically increasing on the majority of paths should reduce the likelihood of instability due to policy incompatibility by restricting the set of implemented policies. A further benefit of pricing in such situations is that any conflicts arising due to prices (for example, two multi-homed ASs disagreeing over which of the two transit ASs should be used as depicted in Figure 4.4(a)) should be resolved precisely in the direction of the party placing the most value in favourable resolution.

Finally, it should be noted that the price is being used only as a metric to

decide between multiple available routes to a destination. As a consequence, reachability should be maintained even when routes are oscillating due to prices changing: oscillating prices only affect the choice of route (from many available routes) to a destination, not the reachability of the destination.

It is noted above that price oscillation might also cause a decrease in the stability of the routeing tables. However, a more serious problem from a financial standpoint is how the operator should deal with rapidly fluctuating charges when these are being used as the basis of inter-operator settlement. Since the volume of traffic involved may be large, so might the amounts of money. An ISP could end up significantly over-spending in the interval between a neighbouring AS increasing its price and the ISP being able correspondingly to increase and advertise its price, particularly if UPDATE messages are being rate limited for route stability reasons. Such a situation could arise naturally, or as a result of a (distributed) denial of service attack by customers of the ISP receiving the advertised price; in either case network administrators should be informed through some mechanism.

This situation allows a number of possible remedies. To cover all such risks providers might wish to buy futures to protect themselves against such situations. Less heavyweight solutions include capping the size of price increments to prevent one ISP getting too far out of step with others. This could have further beneficial impact with respect to the stability of routes, since such capping could help reduce the frequency at which routes change due to price changes.

The final point to note, but one which is not discussed in detail here, is the complexity of potential interaction between application level and routeing level reaction to marking. This might be of particular interest where the application is also charging or being charged for marks, or where the application needs to reflect marks it generates back to the user. A standard example of this situation is a web server which may cause many marks to be generated by transmitting requested data toward a user's web-browser. There must be some way that the value the user receives from the marks generated by the web-server can be recovered by the operator of the web-server.

Subsequent sections discuss the implementation and simulation framework, and describe results of some simple simulations.

4.4 Implementation

This section discusses the implementation work carried out to validate the designs presented in this chapter. A BGP simulator is presented with an associated simulation description language. The following section presents the

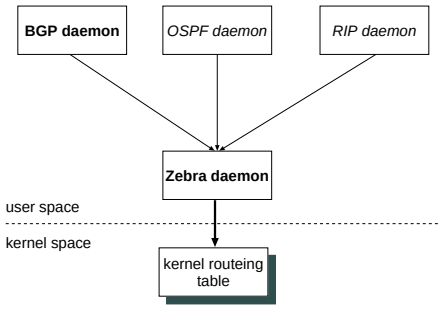


Figure 4.2: The ZEBRA routeing protocol suite.

results from the simulations.

4.4.1 The simulator

Although many simulators exist for network protocols, such as the NS [NSv2] simulator used for much of the research done on TCP and for the work described in Chapter 3, these were considered inappropriate for BGP simulation.

In general such simulators work at a packet level, simulating each packet in the network as a discrete event and then treating network nodes and links as objects which modify the behaviour of these events by introducing delay, forwarding the packet to one or many links, and so on. It is the macroscopic properties of the protocol that are of interest when investigating routeing; typically such properties include stability and scalability. Fully investigating such behaviour requires large simulations to be run, difficult when simulating every packet as with packet level simulators.

Furthermore, the ability to simulate using a deployed implementation of the routeing protocol is attractive. Routeing protocols (and BGP in particular) are notoriously difficult to implement correctly, and writing a simulated version that accurately modelled the protocol would have been difficult.

Consequently, the decision was taken to implement a new routeing protocol simulator, based around a deployed implementation of routeing protocol code. The basis for this simulator is the GNU ZEBRA [Zebra00] protocol suite. This provides a number of routeing dæmons (BGPv4, OSPF, OSPFv6, RIP, RIPng, with others being added), all of which update kernel forwarding tables via the ZEBRA dæmon, as shown in Figure 4.2. This provides a modular system able to run multiple routeing dæmons on a single machine, with each feeding information into the kernel's forwarding tables.

The simulator was implemented over this by modifying the initialization code

in the BGP dæmon so that it would explicitly bind to a local address. By then using the facility in Linux for 'virtual IP interfaces,' multiple copies of the ZEBRA and BGP dæmons could be instantiated on a single machine, and could communicate with other instances, each instance believing itself to be running on an independent router. A discrete event simulator harness was written to enable the instantiation of numerous BGP dæmons within a single Unix process. These instances run in the same way as the standard dæmons, using the BSD sockets API to communicate and executing the standard BGP route management and preference code; the discrete event harness deals with scheduling the BGP instances. Finally, the ZEBRA dæmon was modified to log rather than modify the kernel forwarding tables.

4.4.2 Simulation of load

To study stability behaviour of the network it is necessary to simulate behaviour of traffic in the network under the influence of the routeing protocols. It is infeasible to use real traffic when running multiple routeing dæmon instances on a node for variety of reasons: cross-talk of supposedly independent traffic being multiplexed onto a single physical link; the constrained link bandwidth and packet forwarding bandwidth resource available in a single machine; and the problem of enabling Linux to support multiple independent forwarding tables for the virtual interfaces.

Consequently, the BGP dæmon was also extended to support some notion of 'load' and a simple discrete-event style monolithic simulator written. Topology information is used to calculate an equivalent load for each router: each prefix for which a router chooses a given neighbour as next-hop is considered an equivalent unit load for that neighbour.

This makes the following simplifying assumptions:

All prefixes are assumed to be equally likely destinations. In practice this is untrue, but greatly simplifies the setup and control of the simulator.

Equal numbers of packets are marked per route. Load at a router is directly proportional to the number of best routes pointing at that router. Not only does this make the assumption that all routes are equally likely, but also that the mark probability along all paths is equal.

The network is homogeneous. The simulator currently assumes that all links and routers are of equal capacity and latency. This is clearly untrue in general; however, much of the Internet, especially the core, now uses technologies with commensurate, if not identical, performance characteristics. Furthermore, by causing different numbers of prefixes to originate from different ASs, the simulator does allow the load to differ

<SimSpec>	::=	<BaseConfig>+ [<ZebraConfig>+] [<BgpdConfig>+]
<BaseConfig>	::=	`base` [`configfile` | `logfile`] = *FileName*
	|	`base port` = *PortNumber*
	|	`base` [`presleep` | `postsleep`] = *SleepTime*
<ZebraConfig>	::=	`zebra debugging` = <ZebraDebug>
<ZebraDebug>	::=	`events` | `packets`
<BgpdConfig>	::=	`bgpd debugging` = <BgpdDebug>+
	|	[`bgpd` | *IPAddress*] <BgpdTimer> = *TimerValue*
	|	`AS`*n* `contains` *IPAddress*+
	|	`AS`*n* `advertises` *IPSubnet*+
	|	*IPAddress* `peers` *IPAddress*
	|	*IPAddress* `time` <BgpdTimeEvt>
<BgpdDebug>	::=	`events` | `filter` | `fsm` | `load`
<BgpdTimer>	::=	`holdtime` | `checkload` | `keepalive` | `connect`
<BgpdTimeEvt>	::=	*TimerValue* [`withdraws` | `advertises`] *IPSubnet*

Figure 4.3: The simulation description language grammar.

between ASs.

The result is the simulation of the distribution of load throughout the network by a real implementation of the routeing protocol. Although the simulation of load makes these simplifying assumptions about the behaviour of traffic in the network, the state machine and routeing protocol behaviour are not simplified in any way.

4.4.3 Describing simulations

Each copy of a routeing dæmon needs to be given a configuration file to inform it where it should put logging information, to which addresses it should bind, and other protocol specific information. In the case of BGP this consists of its AS number, its peers in other ASs, and the network prefixes it can reach. To assist with the generation of these configurations, a simple simulation description language was defined. This allows simulations to be described in a single file, and then the relevant configuration files to be generated from this file.

The simulation language currently concentrates on describing simulations for BGP. A description of the grammar is given in Figure 4.3. As can be seen it is very simple, and could easily be extended to provide for simulations including other routeing protocols such as OSPF. The simulation description is parsed to generate the desired output.

4.5 Results

Results for three simple scenarios are now presented and discussed. In each case the stub ASs do not carry transit traffic, and so carry a total load equal to the number of other ASs in the topology (the load that they sink) plus the total number of ASs in the topology (the load that they source). In all simulations, each AS contains only one router; the effect of pricing on IBGP operation is not addressed here.

Pricing was applied using the same price and charge mappings for all nodes: $p_i = l_i$, $c_i^j = p_i + c_{j'}^i$ where $c_{j'}^i$ is the charge advertised to node N_i by the node to be used as best route, $N_{j'}$. The charge for a route was mapped linearly into the LOCAL-PREF, causing the route selection policy to be the most obvious dynamic policy, 'prefer the cheapest route.'

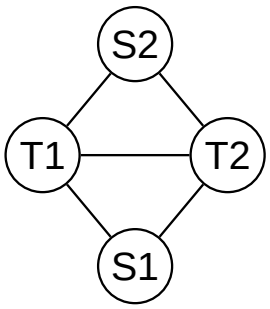
It should be noted that ZEBRA version 0.91a, on which the simulator was based, contains a modification to the standard BGP route selection process to reduce route flap: rather than always breaking ties using the BGP identifier with the lowest value winning, it prefers the first received route. The effects of this are discussed in the presentation of results for each scenario.

The control results presented in Figures 4.5(a), 4.6(a), and 4.7(a) have this modification removed so as to follow the published BGP specification. The 'modified' results presented in Figures 4.5(b), 4.6(b), and 4.7(b) have both this ZEBRA modification and pricing applied. Additional experiments were carried out for the cases where pricing is applied without the ZEBRA modification, and where the ZEBRA modification is applied without pricing. Results from these experiments are discussed in the text, but the results themselves are not presented.

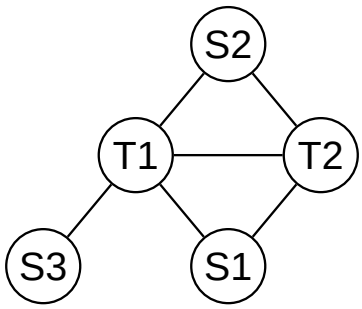
4.5.1 Scenario 1: simple multi-homing

This scenario depicts a multi-homing situation with only one router per AS. Its purpose is to demonstrate the effects of pricing in a basic multi-homing scenario as might occur between two customers both multi-homed via the same transit providers. Pricing is observed to operate as expected, causing load to be evenly distributed between the two transit providers.

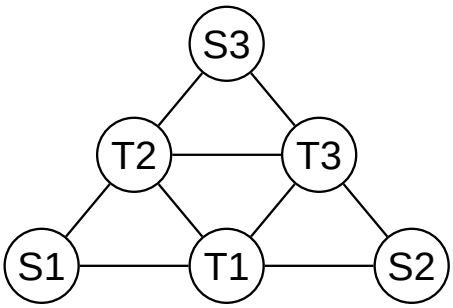
Standard unmodified BGP results are shown in Figure 4.5(a): T_1 deterministically becomes more heavily loaded with 9 units, compared to 7 units for all the other nodes. This is due to the BGP decision process preferring the lowest BGP identifier in situations where the AS-PATH lengths are equal. With the ZEBRA modification but without pricing this determinism is lost: it also becomes possible for T_1 to carry 8 or 7 units and correspondingly for T_2 to carry 8 or 9 units.



(a) Scenario 1: basic multi-homing.

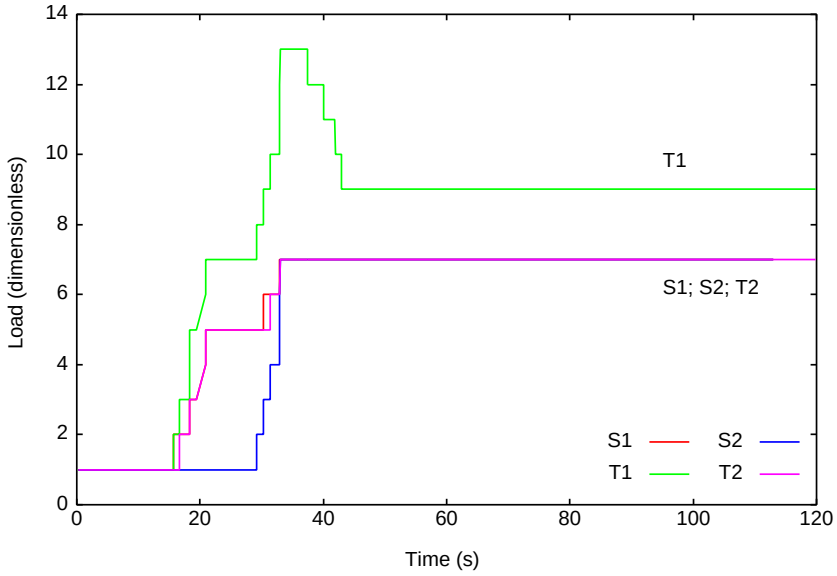


(b) Scenario 2: multi-homing with contended transit.

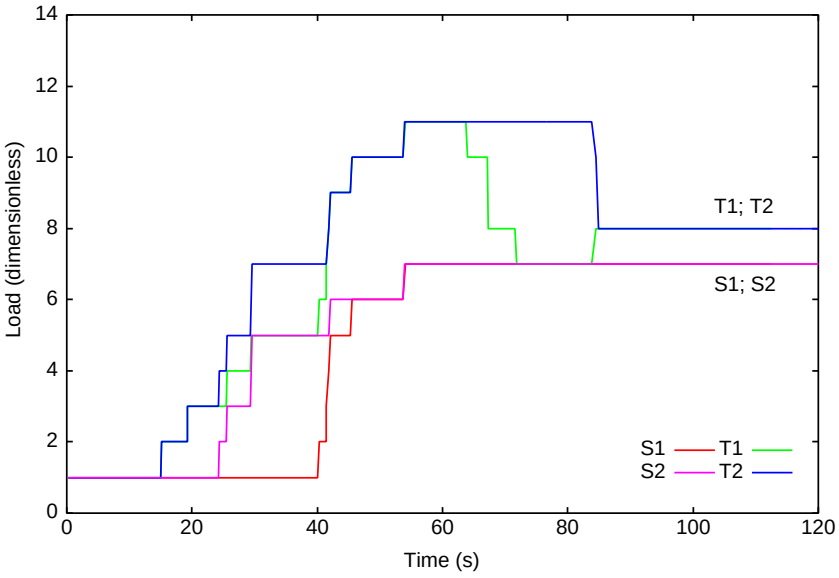


(c) Scenario 3: multi-homing with a complex core.

Figure 4.4: Simulation topologies.



(a) Unmodified BGP.



(b) Modified BGP.

Figure 4.5: Per-node load distributions for Scenario 1 shown in Figure 4.4(a).

The large spike to 13 units in the initial portion of the graph is effectively an artifact of the message ordering imposed by the scheduling of the discrete event simulator harness. Due to the ordering of BGP messages, S_1, S_2, and T_2 discover T_1 in advance of the direct routes to each other. Consequently, they all initially use T_1 to reach each other. As the simulation progresses, BGP information propagates through the network and the routers make more reasonable routeing choices, leading to the final load distribution shown.

With pricing and no ZEBRA modification, the system persistently oscillates. Application of pricing does cause the balanced distribution to be reached. However, since both the topology and load distributions are symmetric in this simulation, the prices are also equal and hence so are the LOCAL-PREF values. As a consequence the BGP decision process passes over the LOCAL-PREF attribute, and the tie is broken in favour of the router with the lowest BGP identifier, T_1. This causes the price on T_1 to increase, leading to one or both of S_1 and S_2 ceasing to prefer it for transit to the other.

This results in one of two cases: either T_2 is preferred by both S_1 and S_2, or T_2 is preferred by only one of S_1 and S_2. In the first case, the price advertised by T_2 becomes higher than that advertised by T_1; in the second case, the prices advertised by T_1 and T_2 become equal. Consequently, the first case leads to the price advertised by T_2 being higher than T_1, and the second case leads again to the symmetric situation where the LOCAL-PREF values are equal, and so the tie is broken in favour of T_1; in either case, it can be seen that the system will continue to oscillate.

With pricing and the ZEBRA modification applied, the system always stabilises to a balanced distribution of load, with both T_1 and T_2 carrying 8 units as shown in Figure 4.5(b). However, the convergence time is approximately double that in the control case, and the process involves approximately 4 times as many BGP messages. Additionally, the choice of route between S_1 and S_2 is non-deterministic: the route from S_1 to S_2 may involve either T_1 or T_2; this may be made deterministic by application of policy through the price-to-charge mapping.

For example, if S_2 wished to encourage traffic to travel towards it via T_2, it could do so by making the charge it advertised to T_1 higher than that advertised to T_2 for a given price. Assume that both T_1 and T_2 implement some rational policy such as 'choose cheapest route' and are taking account of charges advertised to them in setting their own prices, as in these simulations. The effect will be that S_1 will prefer to use T_2 to reach S_2, since the charge advertised by T_2 to S_1 for the prefixes associated with S_2 will be lower than that advertised by T_1.

4.5.2 Scenario 2: multi-homing with contended transit

This scenario depicts a less symmetric multi-homing scenario with one transit AS more heavily loaded than the other. In this sense it is more realistic than Scenario 1, which is unreasonably symmetric. Again, pricing is observed to operate as expected, causing load to be redistributed from the more heavily loaded AS to the more lightly loaded AS, as far as possible. In this case, due to the greater imbalance and smaller possibility of redistributing load, the result is less prone to oscillation than Scenario 1, allowing load-shedding to be more aggressive.

Unmodified BGP causes T_1 to become very heavily loaded with 17 units *vs.* 9 units for all other ASs, as shown in Figure 4.6(a). This occurs for the same reason as the imbalance in Scenario 1 – the default BGP tie-breaker. Again, with the ZEBRA modification but without pricing, it is possible for other stable points to be reached dependent on the ordering of the BGP messages.

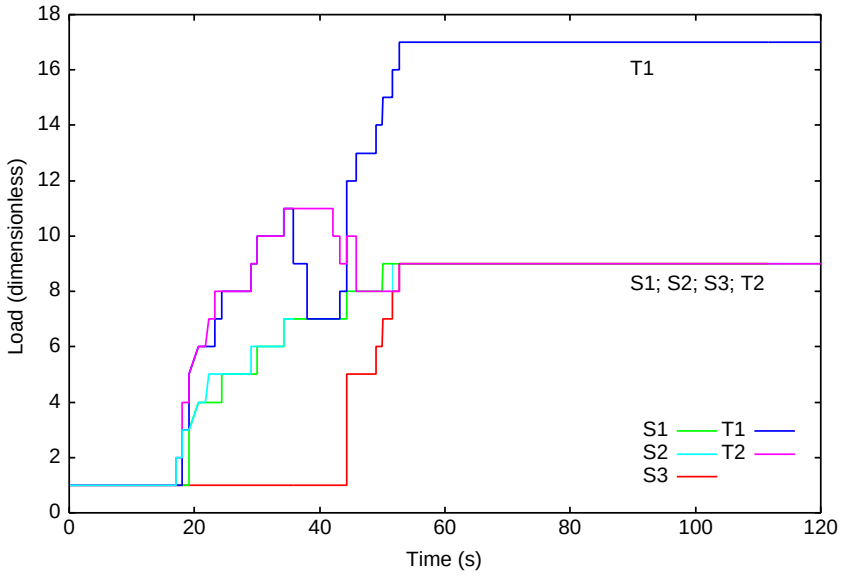
With pricing in place T_1 advertises a higher charge than T_2. Since both S_1 and S_2 have a choice of routes to reach each other, they both choose to use T_2 as transit between them. This increases the load on T_2 to 11 units from 9 units, and decreases the load on T_1 to 15 units from 17 units, shown in Figure 4.6(b). At this point, the charge advertised by T_1 is still higher than that advertised by T_2 and hence this configuration is stable. This is the case with and without the ZEBRA modification since even after T_1 has shed all the load it can, it is still more heavily loaded and hence more expensive than T_2.

However, the convergence time approximately doubles again, and the number of BGP messages also increases, but by a factor of 8 from approximately 200 to just over 1600.

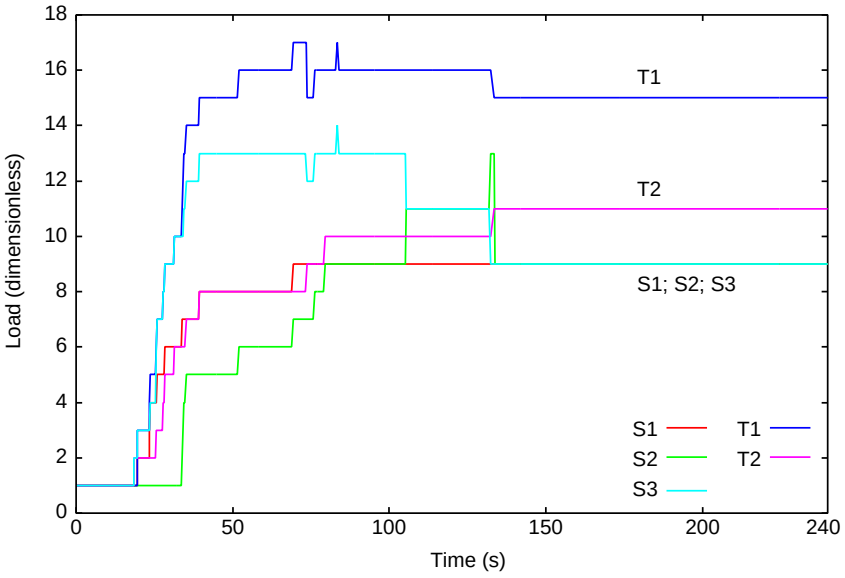
4.5.3 Scenario 3: complex symmetric topology

Finally, this scenario depicts a multi-homing situation but with a more complex core than Scenario 1, such that not all source-destination pairs use the same transit ASs. Once more, pricing operates as expected, causing all transit nodes to experience the same load. However, in this scenario, the more complex core causes the modifications discussed in Section 4.5.4 to come into play, resulting in smoother trajectories as the system approaches stability.

Results for this scenario with unmodified BGP are shown in Figure 4.7(a). Once again, the default tie-breaker process causes imbalance in the final load distribution, with T_2 most heavily loaded at 17 units, followed by T_3 at 15 units, and T_1 at 13 units. S_1, S_2, and S_3 all take 11 units. As before, with the ZEBRA modification but without pricing, it is possible for other stable



(a) Unmodified BGP.



(b) Modified BGP. Note that the x-axis extends to 240 s.

Figure 4.6: Per-node load distributions for Scenario 2 shown in Figure 4.4(b).

points to be achieved, dependent on the ordering of the BGP messages. Similarly, with pricing and no ZEBRA modification, the default BGP tie-breaker causes any stable, balanced allocation achieved to be destroyed, inducing persistent oscillation as described for Scenario 1.

Results when pricing is applied are shown in Figure 4.7(b). Here, the imbalance in loads experienced by T_1, T_2, and T_3 cause different prices to be advertised. This results in all three of these transit nodes finishing with 15 units apiece. Again, the convergence time approximately doubles, and the number of BGP messages increases by a factor of 13 from approximately 240 to 3200.

A more interesting feature of this result is that the approach to the stable point is much smoother. This is caused by the modifications to the pricing algorithm discussed in more detail in Section 4.5.4. Essentially, the increase in the number of nodes (and hence routes in the network) and the more conservative load-shedding policy cause each node to react less violently to an alteration in the distribution of load around the network. This causes the system to behave more smoothly.

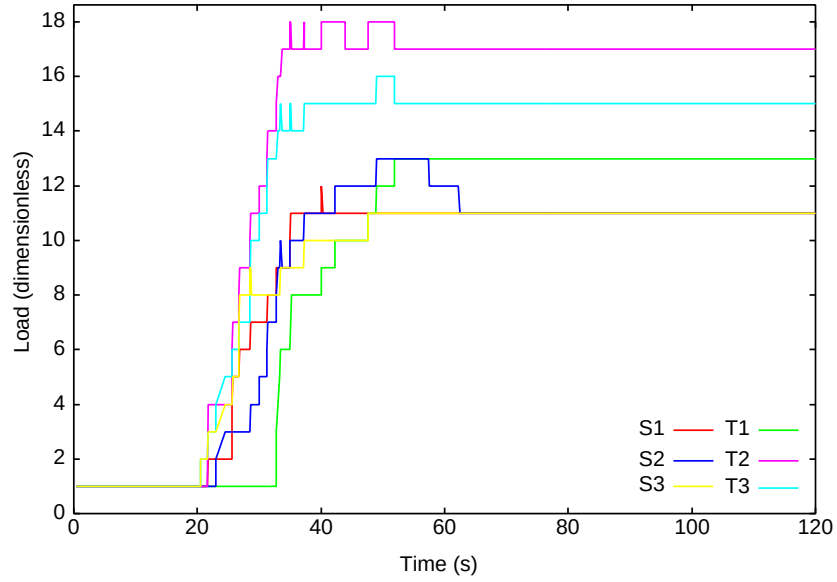
4.5.4 Discussion

The results above have demonstrated that it is possible to achieve route stability and a more efficient distribution of load using BGP with pricing and the ZEBRA modification. However, a number of issues became clear in the course of testing the simulator and running these experiments. These principally affect the traffic model used and the policies applied when redistributing load and are discussed below.

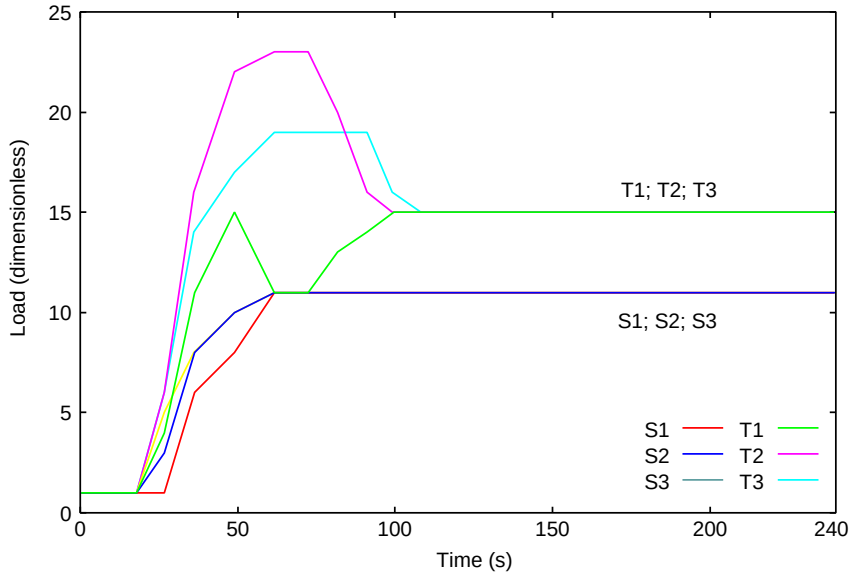
First, even in the stable cases presented above, the number of BGP messages increases as change in the load causes change in the price. These changed prices must then be advertised to peers, requiring BGP messages. With more realistic sizes of network and routeing tables this might become a problem and so deserves further investigation.

Second, correct choice of which routes to move to the cheaper AS can be difficult. If an AS advertises a reduction in its charge, the natural reaction is to cause as many routes as possible to use that AS as transit. However, doing so can increase the load on that AS to the extent that the price reduction is destroyed, and replaced by a price increase. This can cause the AS receiving the advert to now choose to move its routes back, resulting in needless route flap.

This problem can be addressed in two ways. Firstly, the assumption that each AS sources traffic from only one prefix means that BGP has no flexibility over how much traffic to shift: it must move all or nothing. In a real deployment, a



(a) Unmodified BGP.



(b) Modified BGP. Note that the x-axis extends to 240 s.

Figure 4.7: Per-node load distributions for Scenario 3 shown in Figure 4.4(c).

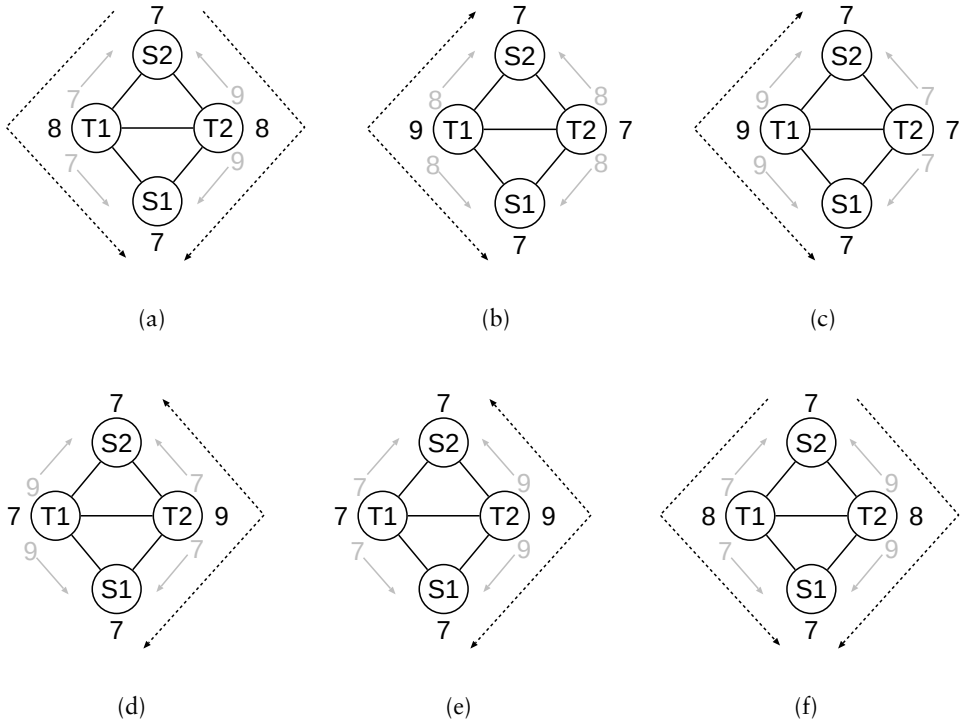


Figure 4.8: Example of persistent oscillation.

single AS is unlikely to both source sufficient traffic and do so toward a single prefix to cause this effect – where such a situation occurs, dynamic SLAs can be considered inappropriate without application of other techniques such as disaggregation. Secondly, route flap damping [RFC2439] can be used to rate limit adverts in such situations.

Finally, simulation of Scenarios 1 and 3 demonstrated an issue with the algorithm as presented in Figure 4.1. It is possible for the routers to persistently oscillate particularly in the more symmetric topologies. An example using Scenario 1 is shown in Figure 4.8. Although the optimum distribution has been reached in Figure 4.8(a) with 8 units through each transit node, old UPDATE messages still propagating through the network cause this distribution to be unstable.

The current price at a node is shown in black, and old price information still propagating through the network is shown in grey. Figure 4.8(a) shows that the balanced state has been reached, with a price of 8 units advertised by each transit node. However, due to out-of-date information still propagating through the network, S_1 and S_2 come to believe that the current prices are 7 for T_1 and 9 for T_2. This causes them to change their preferred route

choice as shown in Figure 4.8(b). Due to the ZEBRA modification, this is unaffected by receipt of the now out-of-date prices of 8 units for T_1 and T_2 in Figure 4.8(b).

In Figure 4.8(c) S_1 and S_2 receive the now correct prices of 9 for T_1 and 7 for T_2. In this example, this causes both S_1 and S_2 to change their preferred routes to use T_2 rather than T_1. This leads to the situation shown in Figure 4.8(d). Subsequently, Figure 4.8(e) shows S_1 and S_2 detecting the new prices of 7 units for T_1 and 9 units for T_2. This time S_2 decides to change its preferred route, and does so such that the effects of this change reach S_1 before S_1 next makes its route preference choices. This causes the prices at T_1 and T_2 to become equal at 8 units, shown in Figure 4.8(f), and the oscillation may repeat.

Although this synchronization can be destroyed simply due to the timing of BGP messages, a mechanism that guarantees to break up this synchronization is required. This is achieved by making two modifications to step 5 of the algorithm. Firstly, the number of routes that may have their LOCAL-PREF altered on the basis of a change in price is limited, in this case to one. This implements a more conservative load shedding policy; changes in price are still re-advertised as soon as they are processed. Secondly, modification of the LOCAL-PREF is only allowed to take place after a delay proportional to the maximum AS-PATH length in the network.

These changes attempt to ensure that changes in price have a chance to propagate throughout the network so that routes are not changed on the basis of out-of-date prices, to prevent situations such as shown in Figure 4.8 occurring. An alternative, less pessimistic, scheme would be to choose the delay randomly from $[0, n]$ where n is proportional to the diameter of the network; this should decrease convergence times while still preventing synchronisation.

Notwithstanding these issues, the results presented do demonstrate that it is possible to implement pricing in BGP such that the protocol converges to a more even distribution of traffic through the network. The resulting distribution can be controlled according to policies considered desirable by the network operator.

4.6 Summary

This chapter discussed management timescale approaches to traffic engineering in the Internet. It began by considering the scope of management timescale traffic engineering, noting that user utility maximisation and network congestion control are more appropriately achieved using data and control timescale approaches to traffic engineering. It continued by consid-

ering inter-AS routeing and pricing, and how current practise relates them to traffic engineering. Subsequently, a new path attribute was proposed and its detailed design presented.

Finally, the implementation of a BGP simulator was described, and initial evaluation of the new path attribute performed in three simple scenarios. Although detailed evaluation is beyond the scope of this thesis, the simulations gave some insight into the possible behaviour of the protocol when extended with the new path attribute. This insight was used to successfully modify the algorithm; however, it is clear that further investigation is required here.

Detailed investigation of different pricing and charging regimes is necessary before deployment could be considered. Similarly, the interactions when different ASs use different pricing and charging functions are unknown and require investigation. The final area for further investigation concerns more operational details BGP of: the behaviour of IBGP with pricing, and algorithms for combining IBGP advertised prices to achieve a price for the AS should be studied. Implementing more complex dynamic policies involving 'quality' estimates of neighbouring ASs, interaction between ASs applying different policies, and interactions between static and dynamic policies should also all be studied further.

This chapter and the preceding chapter have presented two mechanisms for performing traffic engineering at control and management timescales in the Internet. The following chapter now considers how and why these mechanisms, along with data timescale traffic engineering mechanisms, could be deployed in the Internet, and the effects of such deployment.

Chapter 5

Internet traffic engineering

Preceding chapters have discussed mechanisms for the application of pricing to traffic engineering at multiple timescales. The benefits of such approaches have been demonstrated for three mechanisms: admission control and proxying for transport protocols in the Internet, and modifications to BGP, the Internet's inter-AS routeing protocol. However, in a system such as the Internet a key consideration is that of deployment. Such large-scale modifications require a straightforward deployment path lest network operators and users decide that the costs outweigh the benefits and consequently maintain the status quo.

This chapter discusses some of the issues concerning use of the ideas presented thus far for traffic engineering. It begins by noting the requirements of inter-AS traffic engineering and describing the state of the art in this area. It then describes the disincentives and incentives toward deploying modifications such as those presented in preceding chapters in a system such as the Internet. It concludes with a discussion of the affects of such deployment on the structure of the network, its economics, and the services it can offer.

5.1 Traffic engineering requirements

This section considers the requirements that the various participants in the network have with respect to traffic engineering.

5.1.1 User requirements

Users of the network require that traffic engineering be performed to make efficient use of resources. Traffic should be distributed throughout the network

in a manner that maximises the amount of traffic carried whilst attaining the levels of service users desire. This requires that protocols receive timely usage information so that correct traffic distribution decisions may be taken subject to the constraints imposed by users. Without accurate and timely usage information, no routeing protocol is able to make routeing decisions that correctly balance traffic through the network.

Furthermore, in order for traffic to be efficiently distributed through the network, it seems clear that rich peering between network operators should be encouraged, as should the accurate expression of users' desires. Given the problems of effectively managing networks using existing protocols, the incentives for rich peering are not strong. As discussed in Chapter 1, there are also few mechanisms for customers to express their desires clearly to the network.

5.1.2 Operator requirements

Operators desire the ability to distribute traffic efficiently through their networks, and to have some control over from where they allow traffic to enter their networks, and to which networks they allow traffic to exit. They wish to be able to provide differentiated service within their networks for traffic of different types and from different customers. It would also be useful if the network could inform the operator when particular SLAs are becoming inappropriate, either because they have targets that have become unattainable, or because they are being priced too high or low.

In addition to these more technical concerns, a major concern for most network operators is the cost of billing. Various figures, some of them substantial, are quoted as the proportion of the final bill to the customer as taken up with the billing process. In general, operators desire that the billing process is made less complex, more automated, and more accurate, so that the cost of billing is reduced. This includes not only billing to users, but also settlement of SLAs between operators. In general, reduction of management costs is considered highly desirable by operators.

5.2 The state of the art

This section considers the state of the art of Internet traffic engineering. It describes the types of network interconnection and the content of the SLAs into which ISPs enter. It also discusses the implementation of SLAs and the mechanisms available for Internet management.

5.2.1 Network interconnection

There are two accepted ways for network operators to interconnect their IP networks.

The first is through *transit agreements*. In this case the smaller operator becomes a customer of the larger, with the larger operator agreeing to advertise routes to and from the smaller so that traffic can be routed to and from the smaller operator's IP addresses. The second way that operators interconnect is *peering*, controlled by the SLAs (SERVICE LEVEL AGREEMENTS) into which the operators enter.

In both cases arrangements are managed via SLAs. These are legal, rather than technical, agreements and have two forms: SLAs for *bilateral* private peering arrangements between two operators who wish to exchange traffic; and *multilateral* peering arrangements between groups of operators all peering together at some exchange point. They specify the requirements that each party places on the other, the service that each party will provide to the other, any costs a party may incur, and the grounds on which a party may terminate the SLA. A number of SLAs are publicly available [SLA-SPRINT00, SLA-GIGABELL01, SLA-GENUITY01, SLA-LEVEL3, SLA-UUNET00, SLA-UUNET01, SLA-MAE01], and the requirements stated in these agreements can be roughly classified as follows:

Operational support

Operational support covers the more mundane details of interconnecting networks, such as suitable access by the respective operators to the peering point and machines, 24×7 staff support at the network operations centre, rack space for installation of equipment, and power supply.

Network size

Network size covers specification of the geographic diversity of the network, often in terms of a minimum number of *peering points* in the region covered by the peer, and interconnection bandwidth available at those peering points.

Network capacity

This concerns the network's capacity in the region under consideration, in terms of the network's bandwidth (as opposed to the interconnection bandwidth referred to above), and the maximum allowed average busy hour load.

For example, the Worldcom-UUnet [SLA-UUNET00, SLA-UUNET01] agreement specifies connectivity with at least 50% of the peering points in the relevant region (at least 15 states in the US, at least 8 countries in Europe, or at least 2 countries in Asia-Pacific); fully redundant backbone at speed dependent on the region (622 Mb/s in the US, 45 Mb/s in Europe, and 12 Mb/s for Asia-Pacific); and maximum utilization of not more than 50% during the average busy hour.

Total ingress/egress traffic

Total ingress/egress traffic refers to the amount of traffic to be exchanged under the agreement. Since these are peering agreements this usually limits the ratio of ingress to egress traffic so that the imbalance is not too high. It also often includes some statement about the minimum rate of traffic to be exchanged.

For example, the same Worldcom-UUnet agreement specifies 40 Mb/s minimum traffic exchange, and that the ratio of traffic exchanged not exceed 1:1.5 in either direction.

Route control

Route control concerns the manner in which route information to and from the two peers will be treated. Route exchange between peers is via BGP, although different operators may choose to use different protocols as their IGP. Some operators do place constraints on the policy to be expressed through the IGP, such as 'shortest exit policy,' requiring that the transmitting AS route traffic to the exit closest to the receiver. Also specified are policies concerning redistribution and use of routes, and other routeing support. For example, Genuity [SLA-GENUITY01] mandate that their peers support IP's *loose source record route* option at the edges of the network.

5.2.2 Implementation of SLAs

SLAs are agreed between the two parties after high level discussion, and consequently change slowly and are specified for periods of months. Each has high manual overhead leading to a high associated cost, making it less desirable for operators to enter many SLAs. Since the utility of the network increases with the number of participants, this is in tension with the desire for network operators to peer with many other networks. The result is that the network is less well connected than might otherwise be the case.

As described in Section 2.4, routeing in the Internet is currently performed principally by three protocols: OSPF or ISIS for routeing within an AS, and BGP for routeing between ASs. The principal mechanism available to operators to allow them to implement the SLAs they enter with other operators is therefore BGP. Correspondingly, OSPF and ISIS are the principal mechanisms by which operators can manage traffic within their networks to ensure both that whilst it is under their control and at the point that it exits their control it is being treated in a manner which meets applicable SLAs.

Although the original specification of OSPF [RFC1583] included support for calculation of separate routes based on the IP TOS byte, this has since been removed [RFC2178], due to a lack of requisite implementation experience. The current specification uses assignment of metrics to paths to compute shortest paths, but a given path is only allowed a single metric. This prevents separate treatment for different traffic types. When entering into multiple varied SLAs with many other operators, it is likely to be desirable for operators to have the ability to apply different treatment to traffic from different operators.

The DIFFSERV proposals enable operators to treat traffic differently, and to use these different treatments in the specification of SLAs. However, DIFFSERV is only intended to allow the use of forwarding and queueing behaviour at nodes to differentiate between traffic; routeing treatment of traffic is intended to be unaffected.

BGP allows different prefixes to have different preferences within an AS, but provides no way for a receiving AS to advertise a cost to a transmitting AS for carrying its traffic. The `MULTI-EXIT-DISCRIMINATOR` path attribute can be used by the receiver to influence the transmitting AS's choice of entry-point into the receiving AS; however this can be, and often is, ignored by the transmitting AS since they have no incentive to trust it. As described in Section 4.2.1, the commonest way of achieving the desired effect is by the receiving AS prepending multiple copies of its own AS number to the `AS-PATH` when it advertises the prefix to its peers.

5.2.3 Discussion

Operators specify a number of requirements in SLAs when entering peering arrangements. As the preceding discussion notes, there are currently few mechanisms available for the implementation of such agreements. Although BGP allows operators some ability to implement policy between ISPs and the DIFFSERV proposals enable individual ISPs to differentiate between traffic at individual nodes, these mechanisms are unsatisfactory.

It is difficult to automate the implementation of policy within BGP, and manual implementation is prone to error and to potential conflicts between ISPs

leading to persistent oscillation of routeing tables. Attempts to provide policy repositories where operators register the policies they wish to implement using a recognised policy specification language have proved only partially successful, and do not address the problems of automation of policy implementation.

Correspondingly, although DIFFSERV addresses the problem of enabling individual nodes to implement differential packet forwarding, it explicitly does not specify how such differentiation should be implemented. Furthermore, it does not address the translation of DSCPs (DIFFERENTIATED SERVICES CODE POINTS) at network boundaries, leaving it to bilateral agreements between operators as to how traffic sporting a particular DSCP should be treated by the receiving network. The DIFFSERV proposals also do not consider how such agreements are to be implemented and managed.

Finally, MPLS (MULTI-PROTOCOL LABEL SWITCHING) grants greater control over traffic aggregates to operators, allowing the implementation of finer grained SLAs. Consequently, a mechanism to automate the parameterization and settlement of such SLAs is valuable; use of BGP as an LDP for MPLS, enables the price path attribute to be used in MPLS networks.

5.3 Deployment

This section discusses the use of the mechanisms presented in Chapters 3 and 4 to implement SLAs. These mechanisms not only allow more flexible SLAs to be specified, but also allow the automation of their management.

Before any such implementation might be undertaken, aspects such as engineering the code, interfacing with network management tools, and so on would have to be dealt with. Although important, such details are not considered further here as they are not relevant to the thesis being presented.

5.3.1 Disincentives

Objections to the deployment of the mechanisms discussed in Chapter 3 focus on the end-to-end nature of the Internet. It is generally believed that the Internet should only operate at the packet level, and as such, interior nodes should not consider traffic at any other granularity[1]. Flow admission schemes are considered inappropriate since they place extra computation and state within the network, and it is assumed that doing so will violate the Internet's scalability. Furthermore, since flow admission control must involve the

[1]It should be pointed out that the restrictions this places on the functionality of the Internet, particularly in terms of accountability, have been noted for some time [Clark88].

denial of access to the network to some flows, such schemes also violate the assumption of connectivity through the network. Finally, it is assumed that flow admission mechanisms will increase the management effort required by the network operator, making the network more expensive to run.

Implementation of the mechanisms discussed in Chapter 4 requires two things: first, the deployment of the technology presented; and second, co-operation at a management level between providers. The technological issues consist of design and implementation of suitable pricing and charging functions. This is a relatively large problem, but it is hoped that elements of Chapter 4 go some way towards a solution. Potential disincentives toward deployment of these mechanisms can be divided into two categories: *technological* and *managerial*.

The basis for the technological disincentives is that routeing in the Internet is an extremely complex, ill-understood system. It is implemented over a wide variety of platforms both in terms of the hardware and software used in end-systems and routers, and in terms of the support systems in place to deal with issues such as billing and traffic monitoring. This makes controlled deployment of alterations to the structure of the Internet difficult and costly, as evinced by the problems faced in the largely abortive deployment of RSVP, and the continuing deployment problems faced by IPv6 and multicast. In particular, any modification to BGP that may increase fragmentation of the IP address space is seen as unreasonable.

These problems lead in turn to the managerial disincentives. Many people do not see the need for the capabilities offered by improved traffic engineering. They claim that over-provisioning of the network is sufficient for its foreseeable future uses, and point to the failure of modifications like RSVP to provide sufficient benefit to outweigh the associated costs. As well as the obvious costs associated with any such upgrade, there are hidden costs such as the retraining of support staff, the modification of support systems and so on. It also seems to be the case that many people have more emotional reasons for avoiding pricing and charging for the Internet, believing that it should be a free service for all.

Finally, Metcalfe's Law observes that the utility of a network tends to increase as the square of the number of participants[2]. Consequently, deployment of a network-wide change struggles: its utility is not obvious while used by only a small number of the network participants. In particular, if there is a high entry cost associated with starting to use the new network, there is a 'catch-22' situation. Whilst there are only a small number of users the cost of joining is high, and the benefits of joining are perceived to be low. This applies especially to the Internet given its current structure where traffic may

[2]More generally, communication networks have *positive externalities*.

cross many administrative domains as it travels to its destination.

5.3.2 Incentives

The incentives for deployment of the mechanisms presented in Chapter 3 focus on network performance and service differentiation. There are a number of performance benefits to be gained by flow management in the Internet, from the points of view of the user and the network operator. Most fundamentally, allowing the network to deny access to flows gives it another mechanism to deal with congestion. This can help to prevent congestion collapse situations where 'elastic' connection oriented protocols – principally TCP – are not elastic enough and would be forced into a bandwidth region in which they cannot usefully operate.

Furthermore, by restricting the number of flows in the network the operator can provide what might be termed 'soft bandwidth partitioning.' Since users are generally expected to run compliant implementations of protocols, in many cases simply restricting the number of active flows of a given protocol can be enough to provide a soft 'guarantee' of the service each flow will receive. This allows users more freedom to specify the value they are placing on a particular use of the network.

Finally, as previously stated, by allowing users to be more explicit about their requirements from the network, and by allowing network operators to better control the flow of traffic through their networks, billing and management should become more straightforward. As data transfer is predominantly flow based, giving operators easy access to the value and duration of a flow gives them the information required to provide more flexible billing.

Incentives for the deployment of the mechanisms presented in Chapter 4 split into two parts: those applicable to small ISPs who typically do not currently enter into peering arrangements; and those applicable to large ISPs, who typically already enter into peering arrangements.

The principal incentive for a smaller ISP to deploy the mechanisms proposed in this dissertation is to enable them to provide increased service differentiation. Since communication networks generally grow in utility with size, for a small ISP to be successful it should be more aggressive in the services it offers to counter-balance the problem of its small size. By deploying mechanisms such as those discussed in Chapters 3 and 4, greater service differentiation may be offered to users.

Furthermore, in cases where the ISP really only exists to provide connectivity for a single content provider, the ability to ensure the quality of the content distribution channel is valuable and provided by mechanisms such as those

presented in Chapter 4. Finally, by making peering more automated and manageable, it becomes feasible for smaller ISPs to form co-operative groups able to leverage their aggregate size to satisfy the requirements discussed in Section 5.2.1.

The case for larger ISPs to deploy such mechanisms is more subtle. Although those who also act as user facing ISPs may benefit from the admission control techniques suggested in Chapter 3, the techniques of Chapter 4 are of equal relevance. Whilst these techniques do allow more effective competition from the smaller ISPs, they should provide a reduction in management costs, both in terms of administration of SLAs and in terms of the operational management costs of running the network. Furthermore, should co-operatives of smaller ISPs form, it becomes beneficial for the larger ISPs to peer with them; this increases the benefit of reducing the costs associated with peering. Similar incentives concerning control over traffic leaving the ISP's network also apply.

5.3.3 Discussion

In response to the perceived problems with per-flow admission control, it should be noted that whilst it is true that the Internet provides end-to-end connectivity, as soon as traffic crosses administrative boundaries this is *all* that it provides. Any provision for QOS in the Internet therefore requires a process of discussion and agreement between the administrators of the networks over which the service is to be provided.

Per-flow control need have neither excessive state nor computation requirements, as demonstrated in Sections 3.3 and 3.5. Furthermore, admission control should only cause a flow to be dropped where it was likely that the flow would achieve such low bandwidth as to be of no use to the user. Finally admission controllers such as the MTK controller evaluated in Section 3.4 have simple parameterizations that can be easily understood and tuned by operators.

Deployment of these mechanisms is straightforward. It simply requires that the ISP identify any bottleneck links they may have, and install the admission control device at the relevant points. The only difficult aspect of this deployment is the choice of estimator and parameterization for the admission controller. As discussed in Section 3.4.3 the MTK estimator is probably not the most suitable estimator to use here; investigation of more suitable estimators and feedback schemes to dynamically parameterize them is left as an area for further research.

Fundamentally, the response to the disincentives related to the mechanisms presented in Chapter 4 is simply that many of the problems posed in the

previous section are *already* being faced by operators, and must therefore be dealt with. Problems associated with operational system support are becoming more important as more people make use of the Internet, and particularly as more people and businesses start to rely on it as a core infrastructure service. Increased support within the Internet infrastructure to deal with such problems is becoming a necessity.

As discussed in Chapter 1, simple over-provisioning of the network is not a satisfactory solution. Consumer demand and expectations rise in line with the increased capabilities of the technology, and similarly show no signs of slowing down. Furthermore, over-provisioning is not possible everywhere. Although network technologies such as dense wave division multiplexing do allow for huge bandwidths in the core of the network and thus over-provisioning may be reasonable for certain core network providers, bottlenecks will still exist elsewhere.

Additionally, not only is the available bandwidth not homogeneous throughout the network, but neither is the traffic load. Even with future technologies, it may not be possible, and certainly not financially reasonable, to ensure global over-provisioning. Since traffic demand can change dramatically from day to day, and even from hour to hour, having mechanisms to deal with such fluctuations seems valuable.

The technical problems of controlled and incremental deployment can be addressed through techniques and tools such as the BGP simulator presented in Section 4.4.1. This allows for such modifications to be tested, at least in part, before deployment need begin. Similarly, pre-deployment testing of modifications such as presented in Chapter 3 is well-established through the use of tools like NS. Using such tools, operators can attempt to gain some confidence in their proposed policies before deployment.

All of the techniques discussed in Chapters 3 and 4 – implicit admission control, the RTP-ECN-proxy, and the price path attribute – allow for straightforward incremental deployment. Chapter 3 demonstrated that implicit admission control need not adversely affect user applications. Such techniques are also most naturally implemented at particular bottlenecks at the edges of the network, where other 'middle boxes' [Carpenter01] such as firewalls are currently deployed. These places are generally under the control of a single administrative domain, and thus the deployment of implicit admission control need not require co-operation between operators. The extensions to BGP are naturally incremental in that path attributes are intended precisely for extending the protocol whilst retaining interoperability with prior versions.

Consequently, none of the objections to the mechanisms presented in Chapters 3 and 4 are so strong as to make deployment unreasonable. Furthermore, the incentives presented give positive reasons why deployment of these

mechanisms is desirable, and the process of deployment is itself feasible.

The following section discusses the application of the presented mechanisms to the provision of services by ISPs to users.

5.4 Service provision

This section discusses how the mechanisms presented in Chapters 3 and 4 might be used to provide services. It considers service provision from the two principal points of view: users and operators. Finally, it presents a concrete example of service provision using the previously described mechanisms.

5.4.1 User perception

There are two key components to user perception of the Internet: the ISP service to which they subscribe, and any separate content services to which they subscribe.

The mechanisms most relevant to users in terms of the ISP service to which they subscribe are the control mechanisms presented in Chapter 3. Current developments intend users to achieve service differentiation by subscribing to different DIFFSERV levels of service for their traffic. This would allow the routers of the ISP receiving the traffic and transmitting it into the rest of the Internet to apply different treatment to traffic marked for different PHBs. Users would pay extra dependent on the amount of traffic marked with an 'expensive' PHB such as expedited forwarding as compared to traffic marked with a 'cheap' PHB such as best effort.

This style of resource allocation falls under the technology oriented approach described in Chapter 1. It requires that the user (or their applications) be aware of the mechanisms used to implement service differentiation, and that they (or their applications) understand which DSCPs are being used to implement particular PHBs for this ISP. Although reasonable for the standardised PHBs and DSCPs, this does not give operators a great deal of flexibility in utilising locally specified PHBs to provide new or modified services.

A more intuitive mechanism for exposing service differentiation to users would be to provide different subscriptions with associated service levels. Utilising mechanisms such as those presented in Chapter 3, these service levels could be specified in terms of the number of connections or sessions a user was allowed to have running concurrently. Although still somewhat technology dependent (for example, Netscape commonly runs with multiple open HTTP sessions, and software is available that opens multiple TCP connections for a single HTTP download), being allowed a connection to a server

seems more easy to understand than whether or not packets for this application require expedited or assured forwarding, or some other ISP-specific PHB.

For users that desired the extra flexibility provided by DIFFSERV, services could be provided which limited the number of high quality PHB streams. Alternatively, the protocol type or port number could be used to assign flows transparently to service classes. This allows traffic associated with different uses to be given an appropriate QOS. Section 5.4.3 discusses an example in more detail.

The content services to which the user subscribes also have a significant impact on their perception of the Internet. Such subscriptions could be provided as part of the service offered by the ISP; alternatively, the user could subscribe to them directly. Interaction of such content subscription with the mechanisms presented in Chapter 4 is discussed in the following subsection.

5.4.2 Operator perception

The fundamental idea behind the use by ISPs of the presented mechanisms is that they should retain complete control over the traffic entering their network, whilst trying to influence the treatment by other ISPs of traffic exiting their network.

Interaction with other operators would be managed, as currently, through SLAs. However, using the mechanisms presented in Chapter 4, operators should find such SLAs easier to manage, since prices and traffic volumes no longer have to be specified in advance but settlement can be performed based on dynamically calculated prices and measured traffic volumes. The SLAs would simply specify some quality level that the other ISP was offering for traffic it carried, or in more complex cases, specify the semantics of the 'load' value advertised with the price.

The information BGP gleans concerning the global state of the Internet, along with information from per-flow mechanisms as presented in Chapter 3 could then be used to set path weights for IGPs such as OSPF. This could then allow traffic connected with different services, or sporting different DSCPs to be routed to different exits and consequently via different transit providers. Different neighbouring providers have different prices advertised to them for prefixes, and thus the ISP attempts to influence the treatment of traffic it injects into the rest of the network, and the treatment of traffic that it receives from the rest of the network.

Using the mechanisms presented in Chapter 3 intuitive services can be offered to users. Using load information from routers in conjunction with that

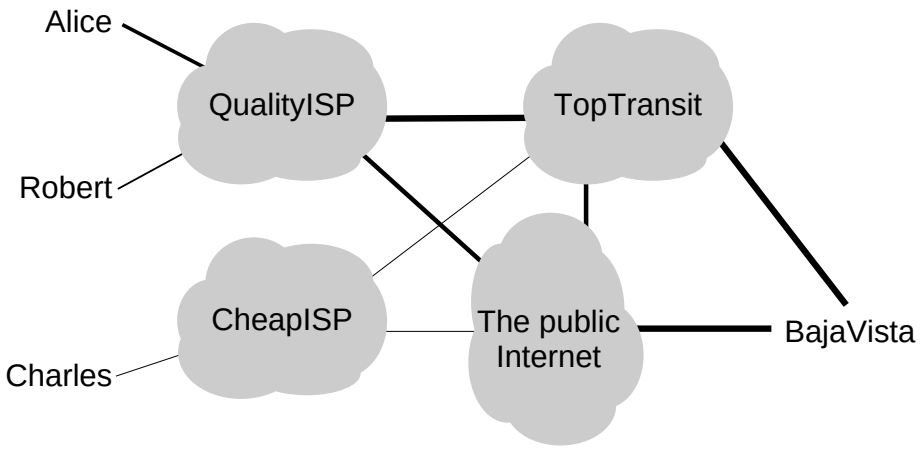


Figure 5.1: A concrete example.

being presented through the modified BGP protocol would allow operators to automate parameter tuning. Content providers might wish to subscribe to a service that enabled them to improve the transport of traffic to and from their site for all customers.

This can be achieved by the content provider effectively paying for the policy the ISP will apply when advertising the content provider's prefixes; by offering to carry traffic cheaply for those prefixes, the ISP can attempt to make itself and its chosen neighbours the preferred route for reaching that content provider. Conversely, traffic from the content provider would be assured that it would take the highest quality path currently available by using DSCPs and control of IGP link weights to choose the route it takes, where possible.

It is worth noting that all of these proposed mechanisms are backward compatible with currently deployed systems. Furthermore, they all offer visible incremental improvements, unlike solutions such as INTSERV with RSVP, which provide no guaranteed improvement as soon as there is a single path element that doesn't support the new capability.

5.4.3 A concrete example

Consider two edge ISPs, *CheapISP* and *QualityISP*, three users, Alice, Robert and Charles, and a content provider, *BajaVista*. The core of the network is made up of a group of core transit providers including *TopTransit* to which the ISPs and content provider connect directly. Alice subscribes to *QualityISP*'s platinum service, and Robert subscribes to their gold service. Charles chooses to subscribe to *CheapISP*. The content provider, *BajaVista*, wishes to ensure that all its users see a high quality service but requires that this be

paid for.

Since Alice subscribes to such a high quality service, she is allowed as many premium quality flows as she likes, subject to some total limit applied by *QualityISP* to ensure that all flows can still make good progress. On the other hand, Robert has a per-user limit applied to the number of high quality flows he can introduce; assuming *QualityISP* was providing a higher quality PHB for real time media streams, this might translate to limiting the number of such streams Robert could achieve, or to utilising a mechanism such as the RTP-ECN-proxy presented in Section 3.5 to limit the quality Robert's streams could achieve. Finally, since *CheapISP* offers no limit on the number of streams its subscribers can have, Charles can use as many streams as he wishes, but may see extremely poor service at times of high load.

Settlement for the service *BajaVista* provides will be provided by *QualityISP* for Alice and Robert as part of their standard service. Charles would have to subscribe to *BajaVista* directly. In all cases, the actual cost *BajaVista* incurs by attempting to appear to all users as if they were well-connected to it could be monitored in terms of the number of marks arriving at the destination ISP, either *QualityISP* or *CheapISP*. The ISP could then either settle itself in the case of *QualityISP*, Alice and Robert, or it could pass the bill on to the user in the case of *CheapISP* and Charles.

BajaVista desires that all users see reasonable service. It subscribes to *TopTransit* specifying its desires. In turn, *TopTransit* advertises *BajaVista*'s prefixes with a low associated cost, and furthermore, it advertises them to other transit ISPs providing a high quality service. This should result in traffic for those prefixes typically following a high quality path to *BajaVista*. Conversely, *BajaVista* also subscribes to a high quality service from *TopTransit*, so that traffic *from BajaVista* will be transmitted efficiently toward the requester, be they Alice, Robert or Charles.

5.5 Consequences

This section discusses some of the consequences of deployment of the work presented in this dissertation. It divides into two parts: how the network's structure might be affected, and how the economic structures associated with network operation and management might be affected.

5.5.1 Network structure

Perhaps the most obvious likely consequence in terms of network structure is that the network should become much more richly connected at an AS level.

By removing many of the barriers to network interconnection, the natural incentive for ISPs to richly peer should come to the fore. This has benefits both for the robustness of the network since the higher degree of connectivity makes routeing around failure easier, and in terms of performance since the higher degree of connectivity at an AS level should lead to a lower diameter network.

It is also reasonable that improved traffic engineering will lead to more efficient network use in terms of the distribution of traffic, further improving performance for users. It seems likely that widespread deployment of admission control could lead to a greater number of long lived flows, which are generally easier to manage and route, and which map more efficiently onto newer network technologies such as MPLS and pure optical networks.

Additionally, new capabilities are made available to network customers in terms of their ability to specify desired levels of service. In the past, the addition of such capabilities has often led to the development of new services and applications able to make use of the richer network semantics now available.

5.5.2 Economic structure

As alluded to in Section 2.5, network operators can currently be divided into two: those large enough to be able to enter into peering arrangements with other providers; and those too small to do so, and who thus only enter into customer/provider arrangements. In general, the richer the peering structure of the network, the better as customers' traffic sees fewer hops to reach its destinations, usually resulting in higher quality of service.

The work presented in this dissertation has the potential to alter this two-tier structure. By simplifying the problems associated with management of peering arrangements, it becomes feasible for operators, particularly smaller operators, to enter into many more such arrangements. As the market matures, economics suggests that the larger operators will endeavour to take over the smaller operators, or otherwise remove the competition they present. It is therefore in the interests of the smaller operators to grow, and this usually occurs by merger.

Consequently, it is likely that the number of market players will reduce, either through takeover or merger[3]. Automatic mechanisms by which such a reduction can be managed, initially in terms of the peering arrangements operators wish to enter into, but potentially also in terms of better quality statistics concerning the value of such businesses, should provide a relatively low-cost way for this to occur.

[3]In addition to bankruptcy, given the current economic climate.

5.6 Summary

This chapter has discussed issues related to the deployment of the ideas and mechanisms presented in Chapters 3 and 4 of this dissertation. It began by considering the requirements placed upon Internet traffic engineering along with the state of the art of Internet traffic engineering. It continued with discussion of the disincentives and incentives toward deployment of the presented work. This was followed by a discussion of how user and operator perceptions of services offered over the Internet might change, and of the network and economic consequences of deployment of the proposed mechanisms.

In conclusion, this chapter has argued that the ideas and mechanisms presented in this dissertation are both useful and deployable, with benefits for both small and large network operators, and network users. The final chapter now concludes the dissertation and considers areas where further work would be useful.

Chapter 6

Conclusions

This chapter concludes the dissertation by summarising the work it described, and noting areas in which further work is required.

6.1 Summary

This dissertation has addressed issues of traffic engineering in the Internet at multiple timescales. *Chapter 1* began by motivating the continuing need for traffic engineering in the Internet. It argued that current approaches are unsatisfactory, and proposed that successful traffic engineering requires consideration of network behaviour at both control and management timescales in addition to data timescales. It concluded by proposing *pricing* as a useful mechanism for implementing and unifying traffic engineering across all timescales.

Chapter 2 then considered background and related work to the problem of Internet traffic engineering. The relevant Internet protocols were reviewed and it was argued that they do not provide sufficient information to enable efficient traffic engineering. The chapter went on to consider resource allocation mechanisms for networks, introducing pricing in particular as such a mechanism. The specific case of resource control in the Internet was then discussed and current proposals were shown to be unsatisfactory. Both intra- and inter-AS Internet routeing protocols were then considered and the principal inter-AS protocol, BGP, argued to be too restrictive in its operation to enable automated inter-AS traffic engineering. Finally this chapter noted the context of the work described in this dissertation, in terms of the structure of the network and the assumptions made about it.

The bulk of the contribution of this dissertation was reported in the following three chapters. *Chapter 3* considered control timescale traffic engineering,

i.e. dealing with connections, concentrating on the TCP and RTP protocols. It began by demonstrating that current approaches to congestion control in TCP can fail in extreme cases to ensure that all users achieve reasonable goodput through the network. It also showed that even if such failure does not occur, TCP allocates resource in a highly variable and potentially unfair manner.

To alleviate these problems, admission control and specifically *implicit admission control* was proposed. The potential impact of this was discussed, followed by design considerations for such a system. Implementation of such a system in the Linux operating system was then presented, demonstrating the feasibility of this approach. Simulation work reporting an implementation of implicit admission control based on measured traffic statistics in the NS simulator followed, and showed that implicit admission control for TCP substantially improves the performance of the network at times of overload. Finally, implementation of an RTP-ECN-proxy demonstrated the feasibility of an alternative to admission control. The presented mechanisms were shown to improve the performance of the network for users, and the controllability of traffic within the network for operators.

Chapter 4 discussed issues related to management timescale traffic engineering, i.e. dealing with aggregates of traffic between ISPs. It described current mechanisms within the Internet for performing this and discussed the relation with data timescale traffic engineering. It then looked in more detail at inter-AS traffic engineering using the BGP protocol, and proposed the *price path attribute* as a mechanism that improves the facility for management timescale traffic engineering. Design considerations for the price path attribute were then detailed and implementation within a BGP simulator described. Finally, results of simulations using this simulator were presented and discussed.

Chapter 5 presented the case for evolving from the state of the current Internet toward that presented in this dissertation. It began by describing the requirements users and operators have for Internet traffic engineering, and the state of the art of their implementation. Having previously demonstrated the benefits of the mechanisms presented in Chapters 3 and 4, arguments for and against the deployment process were presented. The deployment process itself was shown to be desirable; this was followed by a discussion of user and operator perceptions of service provision and a concrete example of the services that deployment of these mechanisms would allow. Finally, consequences from the point of view of the network and associated economic structures were presented.

6.2 Further work

This section notes areas where further work is required, and future directions related work could take. Leaving aside the issues of engineering the prototypes described in this dissertation before deployment could occur, there are a number of areas where further work is required.

The first such area, and one which applies across both Chapters 3 and 4 is the design of suitable marking schemes. Mark rate was used as a congestion measure as it is believed to be a suitably smooth and accurate congestion indicator; however, this deserves further investigation, particularly investigation of whether or not current marking schemes are satisfactory, and how different marking schemes interact. Application to other current network technologies such as wireless networks, IPv6, and MPLS all bear further investigation.

There are three principal pieces of work arising from Chapter 3. The first is the need for accurate and timely estimation of the number of flows a router carries. Each of the three suggested approaches requires further understanding of Internet traffic mixes and behaviour, to a greater or lesser degree. Connected with this is the second area: development of pricing functions suitable for per-flow pricing, their interfacing and interaction with end-system operating systems, and the presentation of the generated information to the user. The latter two areas are already being studied in the context of data timescale packet marking, discussed in Section 2.2.4.

Finally, more flexible mechanisms for performing flow deferral, rather than denial could also be investigated – possibilities include the 'splicing' of TCP connections to allow the admission controller to truly defer the end-to-end connection setup without requiring the end-system to retry.

There a number of areas for further work suggested by Chapter 4. The largest is intra-AS pricing, only briefly touched upon in this dissertation. Modification of IGPs, design of suitable pricing functions, and integration with BGP are key areas for further work. More detailed study of IBGP interaction, along with study of pricing functions, policies to be expressed through the price-to-charge mapping, and routeing and price stability for BGP in general are also required. In general, issues concerning the control and management of resources between operators needs more study.

More radical modifications to certain of the Internet protocols could enable much greater control over traffic distribution within the Internet. This includes mechanisms for efficiently managing prefix disaggregation, and the corresponding increase in routeing table size. Possibilities here rely on more extensive modification to BGP and the way in which prefixes are advertised; by more efficiently encoding of prefixes in the protocol, it becomes possi-

ble to refer to groups of prefixes in routeing tables, restricting routeing table size. Additionally, tighter integration with end-systems and per-packet marking schemes could greatly increase the ability of users to specify their requirements to the network.

6.3 Conclusion

To conclude, it is the thesis of this dissertation that traffic engineering is required at multiple timescales within the Internet, and that current provision for it is unsatisfactory. Users are unable to express their desires to the network, and in any case operators do not have sufficient control over traffic within the network to meet these desires. Furthermore, given mechanisms for multi-timescale traffic engineering, a suitable unifying framework for the policies to be expressed is required.

This dissertation has presented and evaluated mechanisms to achieve this by enabling operators to control access to the Internet on a per-flow as well as a per-packet basis; by providing mechanisms to allow for automated settlement between operators; and by discussing structures within which these mechanisms can be used to increase service differentiation throughout the network, enabling a better match to be achieved between the desires of users and the capabilities of the network.

Pricing has been presented as a unifying framework in which users and operators can express desired policies. It was argued that pricing is well-suited to this task as it is both flexible and intuitive, and it provides both parties with incentives for appropriate behaviour.

In summary, this dissertation has argued that deployment of the presented mechanisms with pricing as a policy framework would help satisfy both user and operator requirements for Internet traffic engineering.

Bibliography

[Ahn95] J. Ahn, P. Danzig, Z. Liu, and L. Yan. *Evaluation of TCP Vegas: Emulation and Experiment*. Computer Communication Review, 25(4):185–195, August 1995. Proceedings of ACM SIGCOMM 1995. (p 16)

[ATMF-PNNI96] The ATM Forum Technical Committee. *Private Network-Network Interface Specification 1.0*, March 1996. af-pnni-0055.000; see also addendum 'Addendum to PNNI v1.0 for ABR parameter negotiation,' af-pnni-0075.000. (p 25)

[ATMF-TM99] The ATM Forum Technical Committee. *Traffic Management Specification 4.1*, March 1999. af-tm-0121.000; see also addendum 'Differentiated UBR,' af-tm-0149.000. (p 20)

[ATMF-UNI96] The ATM Forum Technical Committee. ATM *User-Network Interface Signalling Specification 4.0*, July 1996. af-sig-0061.000; see also addendum 'Signalling ABR addendum,' af-sig-0076.000. (pp 19, 20, 25)

[Bodin00] U. Bodin, O. Schelen, and S. Pink. *Load-tolerant Differentiation with Active Queue Management*. Computer Communication Review, 30(3):4–16, July 2000. (p 19)

[Bouch99] A. Bouch and M.A. Sasse. *It Ain't What You Charge It's The Way That You Do It: A User Persepctive of Network QoS and Pricing*. In M. Sloman, S. Mazumdar, and E. Lupu, editors, *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management (IM'99)*, pages 639–655, May 1999. (p 23)

[Bouch00] A. Bouch, M.A. Sasse, and H.G. DeMeer. *Of Packets and People: A User-Centred Approach to Quality of Service*. In Proceedings of 8th International Workshop

on Quality of Service (IWQoS'00), Pittsburgh, PA, USA, June 2000. (p 23)

[Brakmo95] L.S. Brakmo and L.L. Peterson. *TCP Vegas: End to End Congestion Avoidance on a Global Internet*. IEEE Journal on Selected Areas in Communications, 13(8):1465–1480, October 1995. (p 16)

[Breslau00] L. Breslau, E.W. Knightly, S. Shenker, I. Stoica, and H. Zhang. *Endpoint Admission Control: Architectural Issues and Performance*. Computer Communication Review, 30(4):57–69, October 2000. Proceedings of ACM SIGCOMM 2000. (p 21)

[Carpenter01] B. Carpenter and S. Brim. *Middle boxes: Taxonomy and Issues*. Internet Draft, July 2001. <draft-carpenter-midtax-02.txt>. (p 104)

[Chandy82] K.M. Chandy and J. Misra. *Distributed Computation on Graphs: Shortest Path Algorithms*. Communications of the ACM, 25(11):833–837, November 1982. (p 26)

[Chu99] K. Chu. *Demand for Different Qualities of Service for Internet Access: INDEX Findings*. In Network Modelling in the 21st Century: Royal Society Discussion Meeting. Royal Society, December 1999. Available from `http://www.statslab.cam.ac.uk/~richard/research/topics/royalsoc1999/index.html`. (p 23)

[Clark88] D.D. Clark. *The Design Philosophy of the DARPA Internet Protocols*. Computer Communication Review, 18(4):106–114, August 1988. Proceedings of ACM SIGCOMM 1988. (pp 36, 100)

[Clearwater96] Scott Clearwater, editor. *Market Based Control: A Paradigm for Distributed Resource Allocation*. World Scientific, 1996. (p 20)

[Cocchi91] R. Cocchi, D. Estrin, S. Shenker, and L. Zhang. *A Study of Priority Pricing in Multiple Service Class Networks*. Computer Communication Review, 21(4):123–132, September 1991. Proceedings of ACM SIGCOMM 1991. (p 20)

[Cocchi93] R. Cocchi, D. Estrin, S. Shenker, and L. Zhang. *Pricing in Computer Networks: Motivation, Formulation*

and Example. IEEE/ACM Transactions on Networking, 1(6):614–627, December 1993. (p 20)

[Courcoubetis97] C. Courcoubetis, F.P. Kelly, and R.R. Weber. *Measurement-based Charging in Communication Networks*. Technical Report 19, Statistical Laboratory, University of Cambridge, 1997. (p 21)

[Courcoubetis98a] C. Courcoubetis, F. P. Kelly, V. A. Siris, and R. Weber. *A Study of Simple Usage-based Charging Schemes for Broadband Networks*. In Proceedings of IFIP TC6 International Conference on Broadband Communications (BC'98), Stuttgart, Germany, April 1998. (p 21)

[Courcoubetis98b] C. Courcoubetis, C. Manolakis, and G.D. Stamoulis. *An Intelligent Agent for Negotiating QoS in Priced ABR Connections*. In Proceedings of International Conference on Telecommunications (ICT'98), Halkidiki, Greece, June 1998. (p 22)

[Courcoubetis98c] C. Courcoubetis and V.A. Siris. *An Evaluation of Pricing Schemes that are based on Effective Usage*. Technical Report 214, Institute of Computer Science, Foundation for Research and Technology, Hellas (ICS FORTH), February 1998. (p 21)

[Courcoubetis98d] C. Courcoubetis, G.D. Stamoulis, C. Manolakis, and F.P. Kelly. *An Intelligent Agent for Optimizing QoS-for-Money in Priced ABR Connections*. Telecommunication Systems, Special Issue on Network Economics 1998. (p 22)

[DiffServ01] *IETF: Differentiated Services Working Group*. `http://www.ietf.org/html.charters/diffserv-charter.html`, January 2001. (p 23)

[Dijkstra59] E.W. Dijkstra. *A Note on Two Problems in Connexion with Graphs*. Numerische Mathematik, 1:269–271, 1959. (p 26)

[Draves99] R.P. Draves, C. King, S. Venkatachary, and B.N. Zill. *Constructing Optimal IP Routing Tables*. In Proceedings of IEEE Infocom 1999, New York, March 1999. Also available as Microsoft Technical Report MSR-TR-98-59. (p 79)

[Edell95] R. Edell, P. Varaiya, and N. McKeown. *Billing Users and Pricing for TCP*. IEEE Journal on Selected Areas in

Communications, 13(7):1162–1175, September 1995. (pp 21, 23)

[Elwalid01] A. Elwalid, C. Jin, S. Low, and I. Widjaja. *MATE: MPLS Adaptive Traffic Engineering*. In Proceedings of IEEE Infocom 2001, pages 1300–1309, Anchorage, Alaska, April 2001. (p 32)

[Ensim00] Ensim Corp. *Ensim Corporation*. `http://www.ensim.com/`, 2000. (p 35)

[Falkner00] M. Falkner, M. Devetsikiotis, and I. Lambadaris. *An Overview of Pricing Concepts for Broadband IP Networks*. IEEE Communications Surveys, Q2 2000. Available from `http://www.comsoc.org/pubs/surveys/`. (p 20)

[Fall96] K. Fall and S. Floyd. *Simulation-based Comparisons of Tahoe, Reno, and SACK TCP*. Computer Communication Review, 26(3):5–21, July 1996. (pp 15, 39, 54)

[Feng99] W. Feng, D. Kandlur, D. Saha, and K. Shin. *Blue: A New Class of Active Queue Management Algorithms*. Technical Report CSE-TR-387-99, University of Michigan, April 1999. Available from `http://www.eecs.umich.edu/~wuchang/blue/`. (p 19)

[Floyd93] S. Floyd and V. Jacobson. *Random Early Detection Gateways for Congestion Avoidance*. IEEE/ACM Transactions on Networking, 1(4):397–413, August 1993. (pp 19, 41)

[Floyd94] S. Floyd. *TCP and Explicit Congestion Notification*. Computer Communication Review, 24(5):10–23, October 1994. (pp 14, 15, 39)

[Floyd96] S. Floyd. *Comments on Measurement-based Admission Control for Controlled-Load Services*. Technical Report, Lawrence Berkeley National Laboratory, July 1996. (p 20)

[Floyd00] S. Floyd, M. Handley, J. Padhye, and J. Widmer. *Equation-based Congestion Control for Unicast Applications*. Computer Communication Review, 30(4):34–56, October 2000. Proceedings of ACM SIGCOMM 2000. (p 19)

[Fortz00] B. Fortz and M. Thorup. *Internet Traffic Engineering by Optimizing OSPF Weights*. In Proceedings of IEEE Infocom 2000, Tel Aviv, Israel, March 2000. (pp 32, 69)

[Gao01] L. Gao, T. Griffin, and J. Rexford. *Inherently Safe Backup Routing with BGP*. In Proceedings of IEEE Infocom 2001, pages 547–556, Anchorage, Alaska, April 2001. (p 32)

[Gibbens95] R. Gibbens, F. Kelly, and P. Key. *A Decision-theoretic Approach to Call Admission Control in ATM Networks*. IEEE Journal on Selected Areas in Communications, 13(6):1101–1114, 1995. Special issue on Advances in the Fundamentals of Networking. (p 20)

[Gibbens97] R.J. Gibbens and F.P. Kelly. *Measurement-based Connection Admission Control*. In V. Ramaswami and P.E. Wirth, editors, *Teletraffic Contributions for the Information Age: Proceedings of the 15th International Teletraffic Congress, Washington, DC*, pages 879–888, 1997. (p 20)

[Gibbens99a] R.J. Gibbens and F.P. Kelly. *Distributed Connection Acceptance Control for a Connectionless Network*. In Key and Smith [Key99b], pages 941–952. (p 21)

[Gibbens99b] R.J. Gibbens and F.P. Kelly. *Resource Pricing and the Evolution of Congestion Control*. Automatica, 35:1969–1985, 1999. (p 14)

[Griffin99] T. Griffin and G.T. Wilfong. *An Analysis of BGP Convergence Properties*. Computer Communication Review, 29(4):277–288, October 1999. Proceedings of ACM SIGCOMM 1999. (pp 71, 79, 79)

[Hui88] J.Y. Hui. *Resource Allocation for Broadband Networks*. IEEE Journal on Selected Areas in Communications, 6(9):1598–1608, December 1988. (p 7)

[IntServ00] *IETF: Integrated Services Working Group*. `http://www.ietf.org/html.charters/intserv-charter.html`, September 2000. (p 23)

[Isaacs00] R. Isaacs. *Dynamic Provisioning of Resource-Assured and Programmable Virtual Private Networks*. PhD thesis, University of Cambridge Computer Laboratory, December 2000. (p 25)

[Jacobson88] V. Jacobson and M. Karels. *Congestion Avoidance and Control*. Computer Communication Review, 18(4):314–329, 1988. Proceedings of ACM SIGCOMM 1988. (pp 15, 17, 39)

[Jamin97a] S. Jamin, S.J. Shenker, and P.B. Danzig. *Comparison of Measurement-based Admission Control Algorithms for Controlled-Load Service*. In Proceedings of INFOCOM'97, April 1997. (p 20)

[Jamin97b] S. Jamin, S.J. Shenker, and P.B. Danzig. *Measurement-based Admission Control Algorithms for Controlled-Load Service: A Structural Examination*. Technical Report CSE-TR-333-97, University of Michigan, April 1997. (p 20)

[Kelly97a] F. Kelly. *Charging and Rate Control for Elastic Traffic*. European Transactions on Telecommunications, 8:33–37, 1997. (p 21)

[Kelly97b] F. Kelly. *Internet Economics*, chapter 'Charging and Accounting for Bursty Connections', pages 253–278. MIT Press, 1997. (p 21)

[Kelly98] F. Kelly, A. Maulloo, and D. Tan. *Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability*. Journal of the Operational Research Society, 49:237–252, 1998. (pp 21, 69)

[Kelly00] F.P. Kelly, P.B. Key, and S. Zachary. *Distributed Admission Control*. IEEE Journal on Selected Areas in Communications, 18(12):2617–2628, 2000. (pp 14, 21, 21, 21)

[Key99a] P. Key, D. McAuley, P. Barham, and K. Laevens. *Congestion Pricing for Congestion Avoidance*. Technical Report MSR-TR-99-15, Microsoft Research, February 1999. `http://www.research.microsoft.com/research/network/disgame.htm`. (pp 14, 21, 21, 49)

[Key99b] P. Key and D. Smith, editors. *Teletraffic Engineering in a Competitive World: Proceedings of ITC-16*, volume 3b of *Teletraffic Science and Engineering*. Elsevier Science B.V., June 1999. (pp 119, 127)

[Keynote01] Keynote.com. *Keynote.com*. `http://www.keynote.com/`, 2001. (p 77)

[Khanna89] A. Khanna and J. Zinky. *The Revised ARPANET Routing Metric*. Computer Communication Review, 19(4):45–56, September 1989. Proceedings of ACM SIGCOMM 1989. (p 31)

[Kumar00] A. Kumar, M. Hegde, S.V.R. Anand, B.N. Bindu, D. Thirumurthy, and A.A. Kherani. *Nonintrusive TCP Connection Admission Control for Bandwidth Management of an Internet Access Link*. IEEE Communications Magazine, pages 160–167, May 2000. (pp 45, 53)

[Labovitz97] C. Labovitz, G.R. Malan, and F. Jahanian. *Internet Routing Instability*. Computer Communication Review, 27(4):115–126, October 1997. Proceedings of ACM SIGCOMM 1997. (p 79)

[Labovitz98] C. Labovitz, G.R. Malan, and F. Jahanian. *Internet Routing Instability*. IEEE/ACM Transactions on Networking, 6(5):515–528, October 1998. (p 79)

[Labovitz00] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. *Delayed Internet Routing Convergence*. Computer Communication Review, 30(4):175–187, October 2000. Proceedings of ACM SIGCOMM 2000. (p 79)

[Labovitz01] C. Labovitz, A. Ahuja, R. Wattenhofer, and S. Venkatachary. *The Impact of Internet Policy and Topology on Delayed Routing Convergence*. In Proceedings of IEEE Infocom 2001, Anchorage, Alaska, April 2001. (pp 71, 79)

[Laevens00] K. Laevens, P. Key, 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. `ftp://ftp.research.microsoft.com/pub/tr/tr-2000-104.ps`. (p 16)

[Leland93] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. *On the Self-Similar Nature of Ethernet Traffic*. Computer Communication Review, 23(4):183–193, October 1993. Proceedings of ACM SIGCOMM 1993. (p 10)

[Leland94] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. *On the Self-Similar nature of Ethernet Traffic (extended version)*. IEEE/ACM Transactions on Networking, 2(1):1–15, February 1994. (p 10)

[Lin97] D. Lin and R. Morris. *Dynamics of Random Early Detection*. Computer Communication Review, 27(4):127–138, September 1997. Proceedings of ACM SIGCOMM 1997. (p 19)

[Linx01] Linx. *The London Internet Exchange*. `http://www.linx.net/`, 2001. (p 36)

[Low01] S. Low, L. Peterson, and L. Wang. *Understanding TCP Vegas: A Duality Model*. In Proceedings of ACM SIGMETRICS, June 2001. (p 16)

[MacKie-Mason95] J.K. MacKie-Mason and H.R. Varian. *Pricing Congestible Network Resources*. IEEE Journal on Selected Areas in Communications, 13(7):1141–1149, September 1995. (pp 20, 21)

[Mae01] Worldcom Inc. *MAE Information Site*. `http://www.mae.net/`, 2001. (p 36)

[Massoulié99] L. Massoulié and J.W. Roberts. *Arguments in Favour of Admission Control for TCP Flows*. In P. Key and D. Smith, editors, *Teletraffic Engineering in a Competitive World: Proceedings of ITC-16*, volume 3a of *Teletraffic Science and Engineering*, pages 33–44. Elsevier Science B.V., June 1999. (pp 45, 46)

[Matrix01] Matrix.net. *Matrix.net*. `http://www.matrix.net/`, 2001. (p 77)

[Measure98] *Measure Web Page*. `http://www.cl.cam.ac.uk/Research/SRG/netos/old-projects/measure/`, 1998. (p 50)

[Morris97] R. Morris. *TCP Behaviour with Many Flows*. In IEEE International Conference on Network Protocols, Atlanta, Georgia, October 1997. (p 40)

[Morris99] R. Morris. *Scalable TCP Congestion Control*. PhD thesis, Harvard University, January 1999. (pp 15, 40)

[Mortier00] R. Mortier, I. Pratt, C. Clark, and S. Crosby. *Implicit Admission Control*. IEEE Journal on Selected Areas in Communications, 18(12):2629–2639, December 2000. (p 48)

[Mortier01] R. Mortier, R. Isaacs, and K. Fraser. *Switchlets and Resource-Assured MPLS Networks*. Technical Report

510, University of Cambridge Computer Laboratory, Cambridge, U.K., January 2001. (pp 25, 26)

[MPLS] *IETF: Multiprotocol Label Switching Working Group*. (p 24)

[Murphy94] J. Murphy and L. Murphy. *Bandwidth Allocation By Pricing In ATM Networks*. In Second International IFIP Conference on Broadband Communications, BB-94, March 1994. (p 21)

[Newman98] P. Newman, G. Minshall, and T. Lyon. *IP Switching: ATM Under IP*. IEEE/ACM Transactions on Networking, 6(2):117–129, April 1998. (p 25)

[NSv2] VINT. *The* UCB/LBNL/VINT *Network Simulator, version 2*. `http://www.isi.edu/nsnam/ns/`, 2000. (pp 54, 81)

[Odlyzko99a] A.M. Odlyzko. *Paris Metro Pricing for the Internet*. In Proceedings ACM Conference on Electronic Commerce (EC'99), pages 140–147, 1999. (p 21)

[Odlyzko99b] A.M. Odlyzko. *Paris Metro Pricing: The Minimalist Differentiated Services Solution*. In Proceedings of the 7th International Workshop on Quality of Service (IWQoS'99), pages 159–161, London, UK, May 1999. (p 21)

[Odlyzko00] A.M. Odlyzko. *The History of Communications and its Implications for the Internet*. Available from `http://www.research.att.com/~amo/doc/history.communications0.ps`., June 2000. (pp 13, 17)

[Oliver00] H. Oliver and D. Songhurst. *Market Managed Multiservice Internet*. Telektronikk, 96(2):38–44, 2000. Project home page at `http://www.m3i.org/`. (p 23)

[Padhye99] J. Padhye, J. Kurose, D. Towsley, and R. Koodli. *A Model Based TCP-Friendly Rate Control Protocol*. In Proceedings of the Ninth International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'99), July 1999. (p 19)

[Pan00] R. Pan, B. Prabhakar, and K. Psounis. *CHOKE, A Stateless Active Queue Management Scheme for Approximating Fair Bandwidth Allocation*. In Proceedings of

IEEE Infocom 2000, pages 942–951, Tel Aviv, Israel, March 2000. (p 19)

[Paschalidis00] I. Paschalidis and J. Tsitsiklis. *Congestion-Dependent Pricing of Network Services*. IEEE/ACM Transactions on Networking, 8(2):171–184, April 2000. (p 20)

[Paxson94a] V. Paxson. *Empirically-Derived Analytic Models of Wide-Area TCP Connections*. IEEE/ACM Transactions on Networking, 2(4):316–336, August 1994. (p 56)

[Paxson94b] V. Paxson. *Growth Trends in Wide-Area TCP Connections*. IEEE Network Magazine, 8(4):8–17, July/August 1994. (p 56)

[Paxson97] V. Paxson. *Measurements and Analysis of End-to-End Internet Dynamics*. PhD thesis, Computer Science Division, University of California at Berkeley, April 1997. LBNL-40319; UCB//CSD-97-945. (p 32)

[Perlman00] R. Perlman. *Interconnections*. Addison Wesley Longman, 2nd edition, 2000. (pp 26, 27)

[Rangarajan99] A. Rangarajan. *Early Regulation of Unresponsive Flows*. Master's thesis, University of California at Santa Barbara, July 1999. Technical Report TR-CS-99-26. (p 19)

[Rejaie99] R. Rejaie, M. Handley, and D. Estrin. *RAP:An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet*. In Proceedings of IEEE Infocom 1999, March 1999. (p 19)

[RFC768] J. Postel. *User Datagram Protocol*. RFC 768, IETF, August 1980. (p 14)

[RFC791] J. Postel. *Internet Protocol*. RFC 791, IETF, September 1981. (p 13)

[RFC793] J. Postel. *Transmission Control Protocol*. RFC 793, IETF, September 1981. (p 15)

[RFC891] D.L. Mills. *DCN local-network protocols*. RFC 891, IETF, December 1983. (p 30)

[RFC904] D.L. Mills. *Exterior Gateway Protocol formal specification*. RFC 904, IETF, April 1984. (pp 29, 31)

[RFC975] D.L. Mills. *Autonomous confederations*. RFC 975, IETF, February 1986. (p 31)

[RFC1122] R. Braden and Ed. *Requirements for Internet Hosts – Communication Layers*. RFC 1122, IETF, October 1989. (p 52)

[RFC1142] D. Oran and Ed. *OSI IS-IS Intra-domain Routing Protocol*. RFC 1142, IETF, February 1990. (p 27)

[RFC1305] David L. Mills. *Network Time Protocol (Version 3) Specification, Implementation*. RFC 1305, IETF, March 1992. (p 30)

[RFC1583] J. Moy. *OSPF Version 2*. RFC 1583, IETF, March 1994. (p 99)

[RFC1633] R. Braden, D. Clark, and S. Shenker. *Integrated Services in the Internet Architecture: an Overview*. RFC 1633, IETF, June 1994. (p 23)

[RFC1771] Y. Rekhter and T. Li. *A Border Gateway Protocol 4 (BGP-4)*. RFC 1771, IETF, March 1995. (pp 25, 29)

[RFC1889] Audio-Video Transport Working Group, H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. *RTP: A Transport Protocol for Real-Time Applications*. RFC 1889, IETF, January 1996. (pp 16, 39)

[RFC1998] E. Chen and T. Bates. *An Application of the BGP Community Attribute in Multi-home Routing*. RFC 1998, IETF, August 1996. (p 71)

[RFC2178] J. Moy. *OSPF Version 2*. RFC 2178, IETF, July 1997. (pp 29, 31, 99)

[RFC2205] R. Braden, Ed., L. Zhang, S. Berson, S. Herzog, and S. Jamin. *Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification*. RFC 2205, IETF, September 1997. (pp 8, 23, 25)

[RFC2328] J. Moy. *OSPF Version 2*. RFC 2328, IETF, April 1998. (pp 25, 27, 31)

[RFC2439] C. Villamizar, R. Chandra, and R. Govindan. *BGP Route Flap Damping*. RFC 2439, IETF, November 1998. (p 92)

[RFC2475] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. *An Architecture for Differentiated Service*. RFC 2475, IETF, December 1998. (pp 23, 26)

[RFC2481] K. Ramakrishnan and S. Floyd. *A Proposal to add Explicit Congestion Notification (ECN) to IP*. RFC 2481, IETF, January 1999. (pp 14, 15, 40)

[RFC2597] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. *Assured Forwarding PHB Group*. RFC 2597, IETF, June 1999. (p 24)

[RFC2598] V. Jacobson, K. Nichols, and K. Poduri. *An Expedited Forwarding PHB*. RFC 2598, IETF, June 1999. (p 24)

[RFC2622] C. Alaettinoglu, C. Villamizar, E. Gerich, D. Kessens, D. Meyer, T. Bates, D. Karrenberg, and M. Terpstra. *Routing Policy Specification Language (RPSL)*. RFC 2622, IETF, June 1999. (p 71)

[RFC2676] G. Apostolopoulos, S. Kama, D. Williams, R. Guerin, A. Orda, and T. Przygienda. *QoS Routing Mechanisms and OSPF Extensions*. RFC 2676, IETF, August 1999. (p 32)

[RFC2750] S. Herzog. *RSVP Extensions for Policy Control*. RFC 2750, IETF, January 2000. (p 23)

[RFC2764] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis. *A Framework for IP Based Virtual Private Networks*. RFC 2764, IETF, February 2000. (p 25)

[RFC2796] T. Bates, R. Chandra, and E. Chen. *BGP Route Reflection – An Alternative to Full Mesh IBGP*. RFC 2796, IETF, April 2000. (p 78)

[RFC2914] S. Floyd. *Congestion Control Principles*. RFC 2914, IETF, September 2000. (p 15)

[RFC3031] E. Rosen, A. Viswanathan, and R. Callon. *Multiprotocol Label Switching Architecture*. RFC 3031, IETF, January 2001. (p 24)

[RFC3036] L. Andersson, P. Doolan, N. Feldman, A. Fredette, and B. Thomas. *LDP Specification*. RFC 3036, IETF, January 2001. (p 25)

[RFC3065] P. Traina, D. McPherson, and J. Scudder. *Autonomous System Confederations for BGP*. RFC 3065, IETF, February 2001. (p 78)

[Rhee00] I. Rhee, V. Ozdemir, and Y. Yi. *TEAR: TCP Emulation at Receivers*. Technical Report, Department

of Computer Science, NCSU, April 2000. Available from `http://www.csc.ncsu.edu/faculty/rhee/export/tear_page/`. (p 19)

[Sairamesh95] J. Sairamesh, D. F. Ferguson, and Y. Yemini. *An Approach to Pricing, Optimal Allocation and Quality of Service Provisioning in High-Speed Packet Networks*. In Proceedings of IEEE Infocom 1995, pages 1111–1119, June 1995. (p 21)

[Semret99] N. Semret and A.A. Lazar. *Spot and Derivative Markets in Admission Control*. In Key and Smith [Key99b], pages 757–766. (p 23)

[Shenker93] S. Shenker, D. Clark, and L. Zhang. *A Scheduling Service Model and a Scheduling Architecture for an Integrated Services Packet Network*. Technical Report, Xerox PARC, August 1993. (p 20)

[Shenker94] S. Shenker. *Making Greed Work in Networks: A Game-Theoretic Analysis of Switch Service Disciplines*. Computer Communication Review, 24(4):47–57, August 1994. Proceedings of ACM SIGCOMM 1994. (p 20)

[Shenker95] S. Shenker. *Service Models and Pricing Policies for an Integrated Services Internet*. In Public access to the Internet. MIT Press, Cambridge, MA, USA, 1995. (p 20)

[Shenker96] S. Shenker, D. Clark, D. Estrin, and S. Herzog. *The Internet and Telecommunications Policy*, chapter 'Pricing in Computer Networks: Reshaping the Research Agenda'. Lawrence Erlbaum Associates, 1996. (p 20)

[Sisalem98] D. Sisalem and H. Schulzrinne. *The Loss-Delay Based Adjustment Algorithm: A TCP-Friendly Adaptation*. In Proceedings of the 8th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'98), July 1998. (p 19)

[SLA-GENUITY01] Genuity Inc. *Internet Interconnection Guidelines for Genuity*. `http://www.genuity.com/infrastructure/interconnection.htm`, 2001. (pp 97, 98)

[SLA-GIGABELL01] Gigabell AG. *Gigabell Peering Policy*. `http://rs1.gigabell.net/public/peer.html`, 2001. (p 97)

[SLA-LEVEL3] Level 3 Communications. *Global IP Interconnection Peering Policy*. `http://www.level3.com/us/info/network/interconnection/`, 2001. (p 97)

[SLA-MAE01] MCIWorldcom Inc. MAE *Connection Guidelines*. `http://www.mae.net/doc/maecheck.html`, 2001. (p 97)

[SLA-SPRINT00] Sprint Corporation. *Sprint's Bi-Lateral Peering Policy*. `http://gullfoss2.fcc.gov/prod/ecfs/retrieve.cgi?native_or_pdf=pdf&id_document=6011256512`, 2000. Filed with the FCC as a result of the proposed Sprint-MCIWorldcom merger. (p 97)

[SLA-UUNET00] MCIWorldcom Inc. *UUnet North American Peering Policy*. `http://gullfoss2.fcc.gov/prod/ecfs/retrieve.cgi?native_or_pdf=pdf&id_document=6011256523`, 2000. Filed with the FCC as a result of the proposed Sprint-MCIWorldcom merger. (pp 97, 98)

[SLA-UUNET01] MCIWorldcom Inc. *WorldCom Policy for Settlement-Free Interconnection with Internet Networks*. `http://www.uu.net/peering/`, 2001. (pp 97, 98)

[Sprint00] Sprint. *Sprint Press Release*. `http://www.sprintbiz.com/press/0003/000322roundtrip.html`, March 2000. (p 48)

[Stewart99] J.W. Stewart III. *BGP4 Inter-Domain Routing in the Internet*. Addison Wesley Longman, 1999. (p 29)

[Tangmnarunkit01] H. Tangmnarunkit, R. Govindan, D. Estrin, and S. Shenker. *The Impact of Routing Policy on Internet Paths*. In Proceedings of IEEE Infocom 2001, Anchorage, Alaska, April 2001. (p 32)

[Tassel97] J. Tassel, B. Briscoe, and A. Smith. *An End to End Price-Based QoS Control Component Using Reflective Java*. Lecture Notes in Computer Science, 1356:18–32, 1997. (p 21)

[UKERNA01] UKERNA. *JANET Transatlantic Charges*. `http://www.ja.net/documents/UKERNA_News/1998/september/UKERNA_News5.htm%l/`, 2001. (p 40)

[Varadhan00] K. Varadhan, R. Govindan, and D. Estrin. *Persistent Route Oscillations in Inter-Domain Routing*. Computer Networks, March 2000. Also Technical Report 98-631, Computer Science Department, University of Southern California, September 1997. (p 79)

[Vic01] Lawrence Berkeley National Laboratory/UCB. *The VIC Video-Conferencing Tool*. `http://www-mice.cs.ucl.ac.uk/multimedia/software/vic/`, 2001. (p 64)

[Wang99] J.L. Wang and A. Erramilli. *A Connection Admission Control Algorithm for Self-Similar Traffic*. In Proceedings of IEEE Globecom 1999: Symposium on High Speed Networks, pages 1623–1628, December 1999. (p 20)

[Wang01] Z. Wang, Y. Wang, and L. Zhang. *Internet Traffic Engineering without Full Mesh Overlaying*. In Proceedings of IEEE Infocom 2001, pages 565–571, Anchorage, Alaska, April 2001. (p 32)

[Wetherall00] D. Wetherall. *OTcl Object Oriented Extensions to Tcl*. `ftp://ftp.tns.lcs.mit.edu/pub/otcl/README.html`, 2000. (p 54)

[Xiao00] Xipeng Xiao, A. Hannan, B. Bailey, and L.M. Ni. *Traffic Engineering with MPLS in the Internet*. IEEE Network Magazine, 14(2):28–33, March/April 2000. (p 5)

[Zaumen91] W.T. Zaumen and J.J. Garcia-Luna-Aceves. *Dynamics of Distributed Shortest-Path Routing Algorithms*. Computer Communication Review, 21(4):31–42, September 1991. Proceedings of ACM SIGCOMM 1991. (p 27)

[Zaumen92] W.T. Zaumen and J.J. Garcia-Luna-Aceves. *Dynamics of Link-State and Loop-Free Distance-Vector Routing Algorithms*. Internetworking: Research and Experience, 3(4):161–188, December 1992. (p 27)

[Zebra00] DML Networks, Inc. *The GNU Zebra Routeing Protocol Suite*. `http://www.zebra.org/`, 2000. (p 81)