CIRCUIT SWITCHING
IN THE INTERNET

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Pablo Molinero Fernández
June 2003

UMI Number: 3090644

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Nick McKeown
(Principal Adviser)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Balaji Prabhakar

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Nicholas Bambos

Approved for the University Committee on Graduate Studies:

Abstract

The motivation for this thesis is our desire to build faster routers and switches to accommodate for the traffic growth in the Internet. For the past few years, Internet traffic has been doubling every year, and nothing indicates that this growth rate will slow down in the near future. The Internet forwards information through packet switching, which has so far proven to scale from the early slow phone modems to the current fast link rates. However, it is unclear whether it will continue scaling to match future optical link rates.

Fiber optics and optical switching elements have demonstrated a capacity to forward information that today looks unattainable by electronic switching elements. As a consequence, one possible way of increasing network capacity is to build all-optical packet switches. However, these switches are not possible today because packet switching requires the buffering and processing of packets, and we do not (yet) know how to perform them in optics. On the other hand, optical circuit switches do not have these constraints, and thus they are already in use. The simplicity of the forwarding path in a circuit switch makes it faster than an equivalent router, even when implemented in electronics. In this thesis, I argue that we would greatly benefit from the use of circuit switching in the core of the network, in either electronic or optical form.

Circuit switching is already used in the Internet. Since the beginning of the Internet it is widely used in the core of the network; when early Internet service providers wanted to interconnect remote sites, the only option was to lease a circuit from the long-distance telephone carrier. Chapters 2 and 3 of this thesis analyze what type of network we would build were we to start with a clean slate. After analysis,

modeling and simulation, I conclude that we would be better off with a hybrid network similar to the current one.

A problem with the current circuits in the core is that they are completely decoupled of packets in the edges. Rather than following traffic patterns in real time, circuits are usually provisioned manually, and thus they change very slowly. IP considers circuits to be static, point-to-point, layer-2 links between routers. Chapters 4 and 5 propose two evolutionary ways of integrating circuit and packet switching, so that circuits are automatically controlled by the traffic carried by IP. The first approach uses lightweight, fine circuits to carry single user flows, whereas the second multiplexes several flows onto heavyweight, coarse circuits.

> ... un primer axioma para establecer cualquier sistema educativo: Es objetivo primordial e irrenunciable mantener el sentido universal de la Ciencia y no sólo en un aspecto informativo, sino en el creativo de la investigación.

D. Luciano Fernández Penedo en "Momentos estelares de la enseñanza Española"

Acknowledgements

I would like to thank my advisor Nick McKeown, for the guidance he has provided me through my Ph.D., as well as the members of my reading committee, Balaji Prabhakar and Nick Bambos, and my former advisor, Fouad Tobagi. I am also very grateful to those that helped me improve this thesis by reading it in its early stages, Sundar Iyer, Nandita Dukkipati, Greg Watson and Mary McDevitt. I also would like to thank Hui Zhang for his help in Chapter 2, Byung-Gon Chun for his implementation of a TCP Switch, and Mor Harchol-Balter for her suggestions for the analysis of the CS-SJF discipline in Chapter 3. I want to thank NLANR, Sprint Labs (Chuck Fraleigh and Brian Lyles), CAIDA and Ciena for providing part of the information that have analyzed in this thesis.

I will not forget everybody with whom I been in Gates 342: Youngmi, Pankaj, Amr, Guido, Paul, Yashar, Giulio and Gireesh, and those other members of the research group, Isaac, Rui, Adisak, Da, Steve, Kersten and Mina. Certainly, my life at Stanford has not always been centered around work, and I also would like to remember John, Oskar, Katya, Krishna, Brad, Waël, Athina, Mansour, Victor, Kristin, Lorenz, Chuck, Kostas, Charlie, and Kevin. I also would like to thank the members of the Spanish communities at Stanford, Iberia, and the Bay Area, AESV, specially Juanjo, Alberto, Carlos y Carlos, Victor y Natacha, Victor y Esperanza, José Manuel, Mario, César y Teresa, David y María, Leo, Diego, and Cintya.

Last but not least, I would like to thank those who really made it possible for me to do my Ph.D. at Stanford because of their encouragement and emotional support: my parents, my grandfather, my twin sisters, my brother, my elder sister and my godmother. My love and gratitude to all of you.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Internet has had phenomenal success in the past 20 years, growing from a small research network to a global network that we use in a daily basis. The Internet is logically composed of end hosts interconnected by links and routers. When a host wants to communicate with other hosts, it uses the Internet Protocol (IP) to place information in packets, which are then sent to the nearest router. The router stores, then forwards, packets to the next hop, and through hop-by-hop routing, packets find their way to the desired destination. In other words, end hosts communicate through *packet switching.* With this communication technique, link bandwidth is shared among all information flows, and so these flows are statistically multiplexed on the link. The resulting service is best effort, in the sense that there are no deterministic guarantees.

Another switching technique that is widely used in communication networks, especially in the phone system, is *circuit switching.* When a terminal wants to communicate with another terminal, this technique creates a fixed-bandwidth channel, called a *circuit*, between the source and the destination. This circuit is reserved exclusively for a particular information flow, and no other flow can use it. Consequently, flows are isolated from each other, and thus their environment is well controlled. This Thesis studies how the Internet could benefit from more circuit switching than is prevalent today.

1.1 Motivation

The Internet has been very successful in part because its decentralized control has permitted the rapid development and deployment of applications and services. The success of the Internet is demonstrated by the enormous traffic growth that has already made the Internet carry more traffic than the phone network [24, 69, 100, 47].

If the Internet is based on packet switching, why would I want to use circuit switching? The answer is simple. There is a mismatch between the evolution rates of traffic and capacity of the Internet, and circuit switching can help bridge the gap between demand and supply.

The capacity of the network has to keep up with Internet traffic growth rates that are 10 times larger than that of voice traffic. Coffman and Odlyzko, among others, have been studying traffic growth in the Internet, and they have found that traffic has been doubling every year since 1997 [47, 135]. Studies by RHK [162] and Papagiannaki et al. [140] indicate similar growth rates. The capacity of the Internet should match these growth rates in order to avoid the collapse of the network. The next section studies the evolution trends of the underlying technologies in a router, and, as we will see, router technology is being outpaced by Internet demand.

If routers cannot keep up with demand, then one can only expand the network capacity by adding more nodes and links. This not only requires more equipment, but also more central offices to house them. It is an expensive proposition, and it also creates a more complex network, making network planning and maintenance more difficult. This Thesis takes a different approach; it focuses on how to improve the performance of the existing network by increasing the capacity of switches and links with the use of circuit switching in the core of the network.

1.2 Technology trends in routers and switches

In order to understand the technology trends to compare them to those of traffic, one has to know the functions that packet and circuit switches do, and the technology used to perform them. In the following, I will focus on the switching function (i.e., the forwarding of information) in a network node. Figure 1.1 shows the functional blocks of a packet switch, also called a *router*. When information arrives at the ingress linecard, the framing module extracts the incoming packet from the link-level frame. The packet then has to go through a route lookup to determine its next hop, and the egress port [82]. Right after the lookup, any required operations on the packet fields are performed, such as decrementing the Time-To-Live (TTL) field, updating the packet checksum, and processing any IP options. After these operations, the packet is sent to the egress port using the router's interconnect, which is rescheduled every packet time. Several packets destined to the same egress port could arrive at the same time. Thus, any conflicting packets have to be queued in the ingress port, the output port, or both. The router is called an input-queued switch, an output-queued switch, or a combined input/output-queued switch depending on where buffering takes place [123].

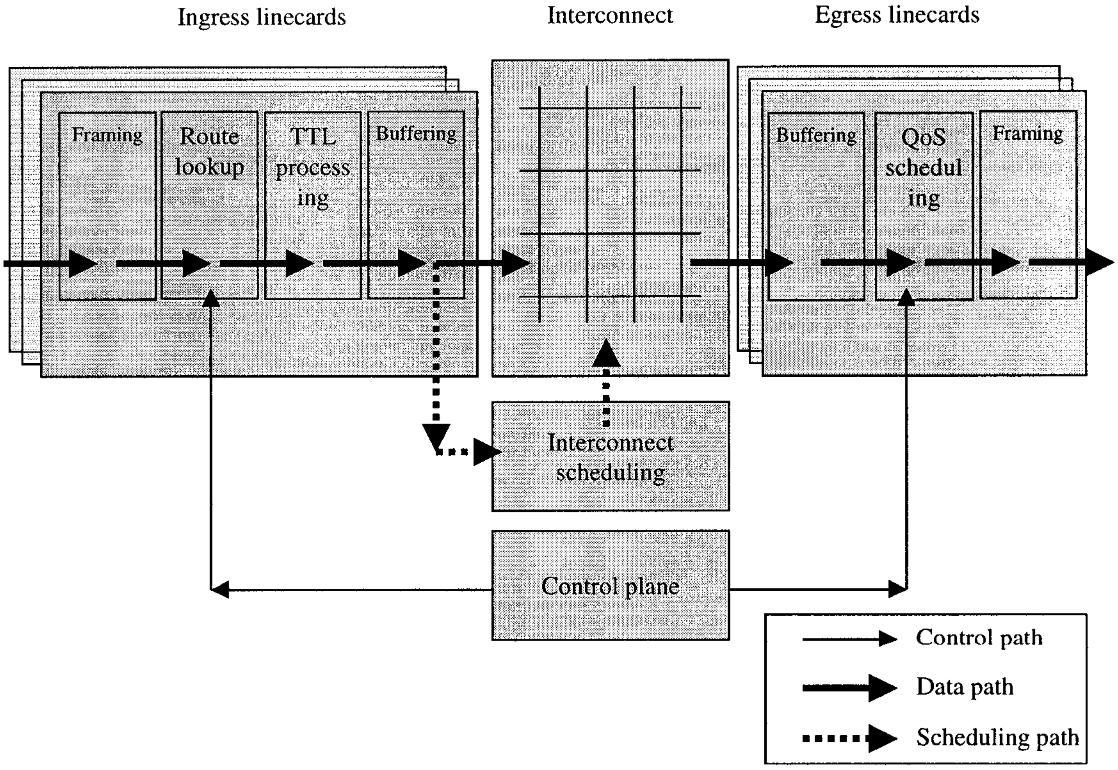

Figure 1.1: Functionality of a packet switch.

In the output linecard, some routers perform additional scheduling that is used to police or shape traffic, so that quality of service (QoS) guarantees are not violated. Finally, the packet is placed in a link frame and sent to the next hop.

In addition to the data path, routers have a control path that is used to populate the routing table, to set up the parameters in the QoS scheduler, and to manage the router in general. The signaling of the control channel is in-band, using packets just as in the data channel. The control plane might obtain the signaling information through a special port attached to the interconnect.

The main distinction between a router and a circuit switch is when information may arrive to the switch. In packet switching, packets may come at any time, and so routers resolve any conflicts among the packets by buffering them. In contrast, in circuit switching information belonging to a flow can only arrive in a pre-determined channel, which is reserved exclusively for that particular flow. No conflicts or unscheduled arrivals occur, which allows circuit switches to do away with buffering, the on-line scheduling of the interconnect, and most of the data-path processing. Figure 1.2 shows the equivalent functions in a circuit switch. As one can see, the data path is much simpler.

In contrast, the control plane becomes more complex: it requires new signaling for the management of circuits, state associated with the circuits, and the off-line scheduling of the arrivals based on the free slots in the interconnect. Usually there is a tradeoff between the signaling/state overhead and the control that we desire over traffic; the tighter the control, the more signaling and state that will be needed. However, in circuit switching, as in packet switching, a slowdown in the control plane does not directly affect the data plane, as all on-going information transmissions can continue at full speed. In general, its data path determines the capacity of the switch.

Another important difference between a router and a circuit switch is the time scale in which similar functions need to be performed. For example, in both types of switches the interconnect needs to be scheduled. A packet switch needs to do it for every packet slot, while a circuit switch only does it when new flows arrive. In general, a flow carries the same amount of information as several packets, and thus packet scheduling needs to be faster than circuit scheduling.

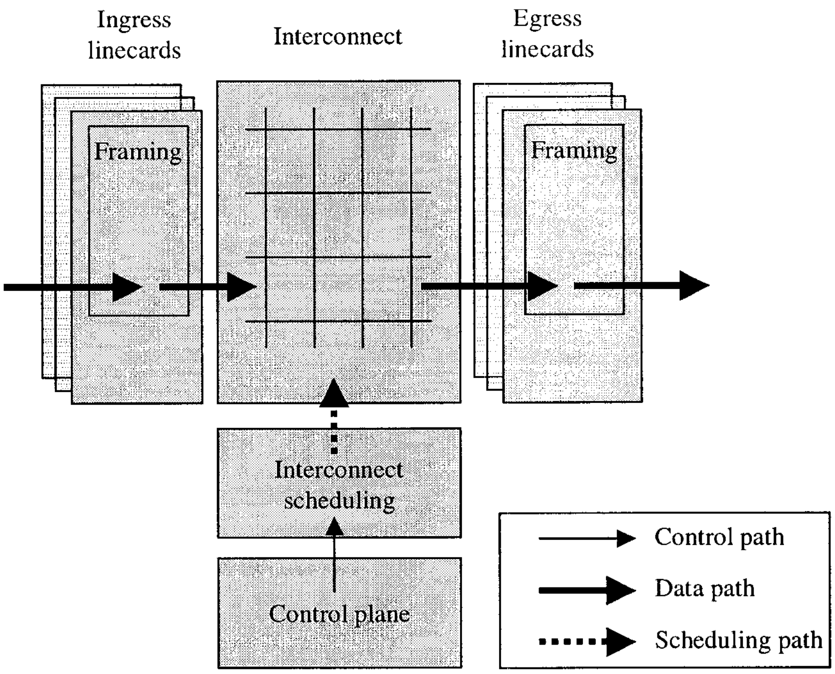

Figure 1.2: Functionality of a circuit switch.

1.2.1 Technology trends

In order to study how the capacity of links and switches will scale in the future, one needs to understand the evolution trends of the underlying technologies used in routers and circuit switches. This enables one to foresee where bottlenecks might occur.

Below, I will focus on the data path of a router, since the data path of a circuit switch is just a subset of it. In general, a router has to:

- **Send and receive packets**: A router receives data through its ingress port and sends it shortly afterwards through the appropriate egress port. Information is sent either through fiber optics for the long haul and high speeds, or through copper cables for the short haul and mid-to-low speeds.

- **Buffer packets**: Packets contend for resources, such as an output port. Conflicts are resolved by deferring the transmission of all but one of the conflicting packets until some later time when the contention has been cleared. As a rule of thumb routers, usually need a *Link Rate* × *Round Trip Time* worth of buffers[1] because of the way the flow control mechanisms of TCP work [182]. For example, for an OC-768 link of 40 Gbit/s and a typical round trip time (RTT) of 250 ms, a linecard needs 1.2 GBytes of memory. Dynamic RAM (DRAM) is thus used to meet this capacity requirement. In addition, some fast Static RAM (SRAM) is needed to cope with the fast arrival rate of packets. The minimum packet size is 40 Bytes, so packets could arrive with a separation of only 8 ns. Designers find it challenging to build packet buffers for a 10-Gbps line card, and it is even more difficult to achieve 40 Gbps, particularly when power consumption is an issue. Most router capacity is limited by memory availability.

- **Process and forward packets**: Routers need to look up the destination address in a routing table to decide where to send a packet next, or in which queue it should be buffered. Packets also need to be scheduled to use the internal interconnect, so that they go from the ingress port to the egress port without contention. Additionally, other fields in the packet header, such as the TTL or the checksum, have to be updated. Currently, this processing and forwarding is done electronically using specialized ASICs, FPGAs or network processors.

To study the performance trends, I will focus on the core of the network, where traffic aggregation stresses network performance the most. The core also uses the state of the art in technology because costs are spread among more users. The backbone of the Internet is built around three basic technologies: silicon CMOS logic, DRAM memory, and fiber optics.

As was mentioned before, Internet traffic has been doubling every year since 1997.[2] In contrast, according to Moore's law, the number of functions per chip and the

[1] In practice, routers are built to handle congestion for a much lesser period of time, needing fewer buffers, but increasing the packet loss probability during overload.

[2] This trend could be broken by the sudden adoption of a new bandwidth-intensive application, such as video streaming, similar to the period of 1995-6 when the massive adoption of the web made traffic double every 3-4 months.

number of instructions per second of microprocessors have historically doubled every 1.5 to 2 years[3] [3, 144]. Historically, router capacity has increased slightly faster than Moore's law, multiplying by 2.2 every 1.5 to 2 years. This has been due to advances in router architecture [123] and packet processing [82].

DRAM capacity has quadrupled on average every three years, but its frequency for consecutive accesses has been increasing less than 10% a year [144, 143], equivalent to doubling every 7 to 10 years. Modern advanced DRAM techniques, such as Synchronous Dynamic RAM (SDRAM) and Rambus Dynamic RAM (RDRAM), are attacking the problem with I/O bandwidth across pins of the chip, but not the latency problem [58]. These techniques increase the bandwidth by writing and reading bigger blocks of data at a time, but they cannot speed up the time it takes to reference a new memory location.

Finally, the capacity of fiber optics has been doubling every 7 to 8 months since the advent of DWDM in 1996. However, the growth rate is expected to decrease to doubling every year as we start approaching the maximum capacity per fiber of 100 Tbit/s [124]. Despite this future growth slowdown of DWDM, the long-term growth rate of link capacity will still be above that of Internet traffic at least past the year 2007 [116].

Figure 1.3 shows the mismatch in the evolution rates of optical forwarding, traffic demand, electronic processing, and electronic DRAM memories. We can see how link capacity will outpace demand, but how electronic processing and buffering clearly drag behind demand. Link bandwidth will not be a scarce resource, but the information processing and buffering will be. Instead of optimizing the bandwidth utilization, we should be streamlining the data path.

Figure 1.3 shows how there is an increasing performance gap that could cause bottlenecks in the future. The first potential bottleneck is the memory system. Routers may be able to avoid it by using techniques that hide the increasingly high access times of DRAMs [91], similarly to what modern computers do. With these techniques access times come close to those of SRAM, which follows Moore's law. However, they

[3] Experts believe that this trend will slow down as microprocessor and ASIC technologies gradually move from the current two-year cycle to a three-year node cycle after 2004.

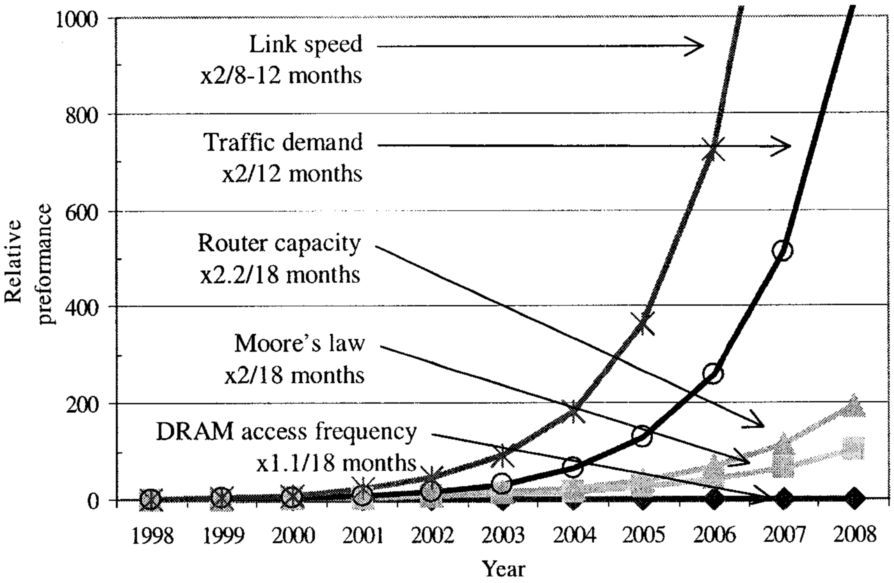

Figure 1.3: Trends of traffic demand and the underlying technologies in the Internet [1998 = 100%]. Trends for Silicon processing and router forwarding capacity are kept at the same value as today, despite talks of a slow down after 2004.

make buffering more complex, with deeper pipelining, longer design cycles and higher power consumption.

The second potential bottleneck is information processing. The trend would argue for the simplification of the data path. However, there is a lot of pressure from carriers to add more features in the routers, such as intelligent buffering, quality-of-service scheduling, and traffic accounting.

If we keep the number of operations per packet constant, in ten years time, the same number of routers that we currently have will be able to process 200 times as much traffic as today. In contrast, traffic will have grown 1000 times by then. This means that we will have a five-fold performance gap. In ten years time we will need five times more routers as today. These routers will consume five times more power, and will occupy five times more space.[4] This means building over five times as many central offices and points of presence to house them, which is a very heavy financial

[4] Actually they will occupy more than five times the space, as many of the routers linecards will be used to connect to other routers within the same central office, rather than to routers in other locations.

burden for the already deeply indebted network carriers. To make matters worse, a network with more than five times as many routers will be more complex and more difficult to manage and evolve. The economical and logistical cost of simply adding more nodes is prohibitive, so we need to be creative, and think out of the box, trying to find a more effective solution that solves the mismatch between traffic demand and router capacity, even if it represents a paradigm shift.

1.2.2 Optical switching technology

One possible solution is to use optical switching elements. Optics is already a very appealing technology for the core because of their long reach and high capacity transmission. Additionally, recent advances in MEMS [15, 83], integrated waveguides [85], optical gratings [101, 27], tunable lasers [187], and holography [149] have made possible very high capacity switch fabrics. For example, optical switches based on MEMS mirrors have shown to be almost line-rate independent, as opposed to CMOS transistors, which saturate before reaching 100 GHz [3, 130]. Ideally, we would like to build an all-optical packet switch that rides on the technology curve of optics. However, building such a switch is not feasible today because packet switching requires buffers in order to resolve packet contention, and we still do not know how to buffer photons while providing random access. Current efforts in high-speed optical storage [178, 109, 151] are still too crude and complex. In current approaches, information degrades fairly rapidly (the longest holding times are around 1 ms), and they can only be tuned for specific wavelengths. It is hard to see how they could achieve, in an economical manner, the high integration and speed that provides 1.2 Gbytes of buffers to a 40 Gbit/s linecard.[5]

Another problem with all-optical routers is that processing of information in optics is also difficult and costly, so most of the time information is processed electronically, and only the transmission, and, potentially, the switching is done in optics. Current optical gates are all electrically controlled, and they are either mechanical (slow and wear rather quickly), liquid crystals (inherently slow), or poled $LiNbO_3$

[5] Needless to say, this vision of the future could completely change if a breakthrough in technology made fast, high-density optical memories possible.

structures (potentially fast, but requiring tens of kV per mm, making them slow to charge/discharge). Switching in optics at packet times, which can be as small as 8 ns for a 40 Gbit/s link, is very challenging, and, thus, there have been proposals to switch higher data units [188], called optical bursts. Rather than requiring end hosts to send data in larger packets, these approaches have gateways at the ingress of the optical core that aggregate regular IP packets into mega-packets. These gateways perform all the buffering that otherwise would be performed in the optical core, so the buffering problem is not eliminated, but rather pushed towards the edges.

In summary, all-optical routers are still far from being feasible. On the contrary, all-optical circuit switches are already a reality [15, 111, 112, 174, 54, 32]. This should not be a surprise, since circuit switching presents a data path that requires no buffering and little processing. For example, Lucent has demonstrated an all-optical circuit switch, using MEMS mirrors, with switching capacities of up to 2 Pbit/s [15]; this is 6000 times faster than the fastest electronic router [94].

Even when we consider electronic circuit switches and routers, the data path of circuit switches is much simpler than that of electronic routers, as shown in Figure 1.1 and Figure 1.2. This simple data path of circuit switches allows them to scale to higher capacity than equivalent electronic routers. This is confirmed by looking at the fastest switches and routers that are commercially available in the market at the time of writing; one can see that circuit switches have a capacity that is 2 to 12 times bigger than that of the fastest routers, as shown in Table 1.1. The simple data path of circuit switches comes at the cost of having a more complex control path. However, it is the data path that determines the switching capacity, not the control path; every packet traverses the data path, whereas the control path is taken less often, only when a circuit needs to be created or destroyed.

In this Thesis, I argue that we could close the evolution gap between Internet traffic and switching capacity by using more circuit switching in the core, both in optical and electronic forms. I also explore different ways of how one could integrate these two techniques.

Product	Type of switch	Bidirectional switching capacity
Cisco 12016	router	160 Gbit/s
Juniper T640	router	320 Gbit/s
Lucent LambdaUnite	circuit switch	320 Gbit/s
Ciena CoreDirector	circuit switch	640 Gbit/s
Tellium Aurora 128	circuit switch	1.28 Tbit/s
Nortel OPTera HDX	circuit switch	3.84 Tbit/s

Table 1.1: Bidirectional switching capacities of commercial switches [42, 94, 40, 174, 132]. While I have tried to be careful in the comparison, comparing product specifications from different vendors is not necessarily comparing "apples with apples", and should be taken only as a rough indication of their relative capacities.

1.3 Circuit and packet switching

If one had to give a very succinct description about how the Internet works, one would say it as being composed of end hosts and routers interconnected by links, as shown in Figure 1.4. In more detail, the Internet is a packet-switched, store-and-forward network that uses hop-by-hop routing and provides a best effort network service. This technology was chosen because it enabled a robust network that made an efficient use of the network resources [11, 43, 172, 14].

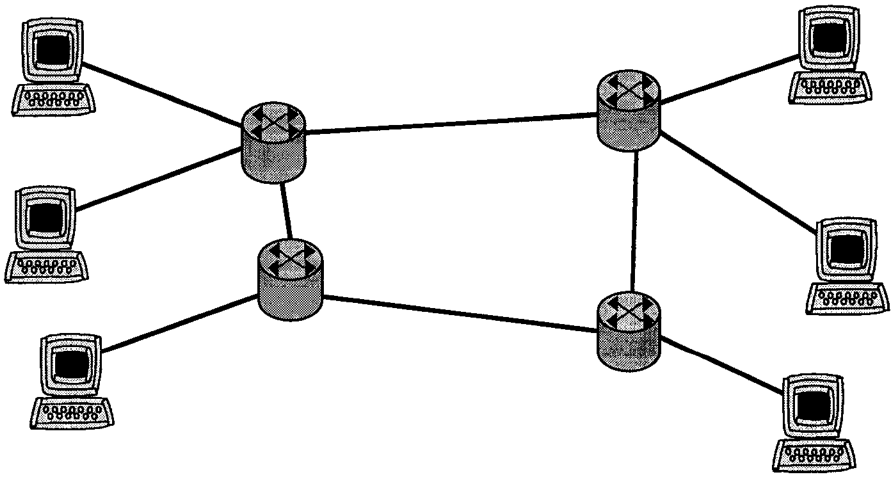

Figure 1.4: Simple architecture of the public Internet as described in textbooks.

However, the real Internet is more complex than this, and if we look closely, we will find that there is plenty of circuit switching in the Internet, as shown in Figure 1.5. We have circuit switches both in the access networks (leased lines, DSL and phone modems), and in the core of the network (SONET/SDH and DWDM). This figure also shows the market sizes in the year 2001, and it shows how the market sizes for the segments that use circuit switching are significant.

The current mix of packet and circuit switching in the Internet is due to historical reasons. In the early days of the Internet, when two Internet Service Providers (ISPs) in different and distant locations wanted to interconnect with each other, they leased a connection from the only companies that had a continent-wide network, that is the long-distance telephone companies, and these companies have always based their service on pure circuit switching. Similarly, the circuits in the edges were one of the few options for an ISP to get to its customers, namely, by using the existing infrastructure of the local telephone company.

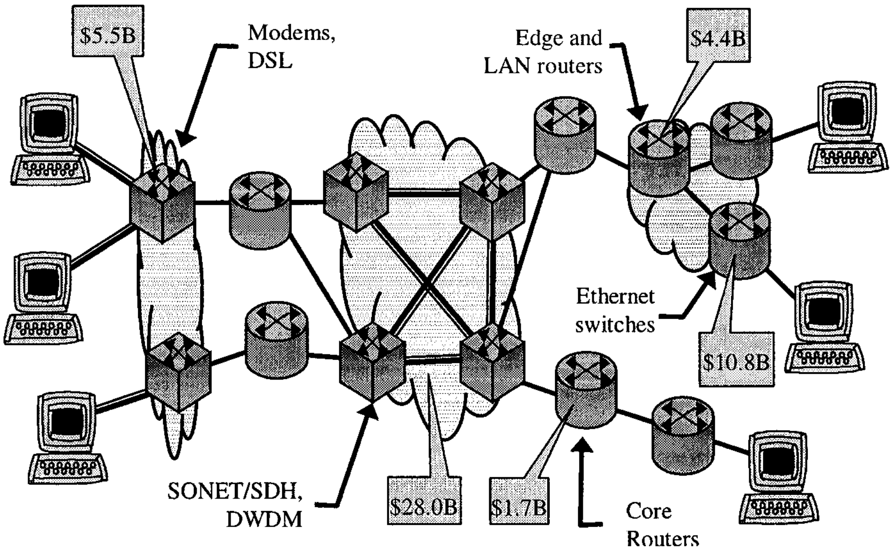

Figure 1.5: Architecture of the public Internet as found in the real world. The figures in the boxes represent the world market sizes in the year 2001. [161, 158, 157, 60, 61]

Given the current situation, one may ask two related questions. First, is this hybrid architecture the right network architecture? If we were to rebuild the Internet from scratch and with unlimited funds, would we choose a solution based on only packet switching, only circuit switching, or a mix of the two? Second, given that it would be too costly to build a brand new network, how can the current legacy Internet evolve in the future? Will the network still follow a hybrid model as today, or will it change? These two questions are the focus of the first part of the Thesis. I will conclude that it makes more sense to use circuit switching in the core and packet switching in the edges of the network.

Currently, the circuits that we find in the Internet are considered by IP as static, layer-2, point-to-point links. In other words, the circuit and packet switched parts of the network are completely decoupled, and changes in IP traffic patterns do not prompt an automatic reconfiguration of the circuits over which IP travels. It is usually the case that circuits are manually provisioned by either the network operator (circuits in the core) or the end user (circuits as access lines). This also means that the time scale in which circuits operate is much larger than that of packets.

We would make a more efficient use of the network resources if we could integrate the world of circuits with that of packets in such a way that circuits follow in real time the fluctuations of the packet switched traffic. In this Thesis, I make two proposals of how to integrate these two technologies in an evolutionary manner, without changing existing end hosts and routers. One approach uses fine-grain, lightweight circuits; the other uses coarse-grain, heavyweight circuits (such as optical wavelengths).

1.3.1 Virtual circuits

There is a third family of networks, which uses virtual circuits, such as ATM or MPLS. This family attempts to get the best of two worlds: on one hand, it takes the statistical multiplexing of packet switching. On the other hand, the traffic management and quality of service of circuit switching. Despite their name, virtual circuits are essentially a connection-oriented version of packet switching; it forwards information as packets (sometimes called cells), but it keeps connection state associated with each flow. In contrast, IP is based on the connectionless switching of packets, where no per-flow state is kept. Switches using virtual circuits are hard to design;

they have the scalability issues of both the data path of packet switching and the state management and signaling of circuit switching. Therefore, virtual circuits will not be studied any further in this Thesis.

In the early 90's, there was a race between IP and ATM to dominate data networks. In the end, IP routers prevailed over ATM switches partly because the former were simpler and thus faster to hit the market and easier to configure. In contrast, MPLS works just below IP, rather than competing with it, and it is an attempt to do simple traffic engineering in the Internet. Only recently has MPLS started getting sizable deployment with some backbone carriers [34].

1.4 Performance metrics for core IP routers

To study what network architecture is better, we need to have some performance metrics to compare the different options. In the network, there are two main stakeholders: the end users and the network carriers. Evidently, they have different concerns and views of the network. The most common use of the network is to request and download pieces of information[6] (be a web page, an image, a song, a video or a record in a database). After *reachability*, the end user is mainly interested in a fast *response time*, defined as the time since we request the object until the last byte arrives. Another important set of user applications (e.g., voice or video conferencing and streaming, or gaming) requires *quality of service (QoS)* guarantees, such as bounds on bandwidth, maximum delay, delay jitter or loss. These network guarantees are often expressed as a service level agreement between the network user and the ISP.

Network carries have a very different set of requirements. Tables 1.2 and 1.3 show a survey by BTexact of several IP backbone carriers about their current concerns and the features they will need in the next 2 years.

After interconnectivity, which is always taken as a given, router *reliability* and *stability* are the greatest concern to carriers today. Significant improvements are required in these areas, particularly in the area of software reliability. It is highly undesirable to have equipment that fails, needs continuous attention, ties up valuable

[6] today, over 65% of the traffic is web browsing and peer-to-peer file sharing [31].

1	equipment reliability and stability
2	scalability
3	performance
4	feature support
5	management
6	total cost of ownership
7	environmental considerations (power, size)

Table 1.2: Concerns of carriers for network equipment in decreasing order of importance [25].

1	denial of service attack mitigation
2	wire-rate performance of interfaces
3	system access security
4	port density improvements
5	quality of service support

Table 1.3: New features required by carriers for network equipment in decreasing order of importance [25].

human resources and spoils the reputation of the carrier. Following in importance are *scalability* and *performance.* Even though the total cost of ownership comes last in the survey, the economic problems that numerous carriers currently face have probably increased its relevance; a good network has to come at a reasonable cost.

In terms of new features that carriers desire, the mitigation of Denial of Service (DoS) attacks ranks first, as such attacks can make the network connection unusable, damaging the reputation of the carrier. Improvements in performance come second, followed by better authentication and access security. Quality of service is last; however, it is more relevant for an operator that wants an integrated network that carries both low-margin data traffic and high-margin voice traffic.

In summary, end users care about low response time and quality of service, whereas network operators desire reliability, scalability and performance. As we will see end users will see no difference when using circuit switching or packet switching in the core of the network, whereas network carriers will clearly benefit from getting a network of higher capacity and reliability.

1.5 Understanding Internet traffic and failures

Before we can choose a solution for the network, we need to understand what types of workloads are currently being injected into the Internet and how reliable different elements in the network are. Different workloads will stress the system in different ways and will have different performance requirements and notions of quality of service. For example, a workload with traffic sent periodically in fixed-sized bursts will behave quite differently than one that has a lot of variation in terms of interarrival times, flow durations and flow rates. The first workload would be best served by a slotted network, while the second one would not.

This Thesis provides a short analysis and discussion of the traffic and failures that we see in real networks, especially in or near the backbone. Some of these results are based on my own analysis of traffic traces [131, 170], other results have been reported elsewhere [31, 30, 102, 107, 106, 105]. In general, one is interested in knowing both the type of application (to prioritize the performance metrics) and the distributions and correlations of:

- interarrival times (of flows and packets)
- sizes or durations (of flows and packets)
- transmission rates of flows
- failures of network elements

Based on those observations, we can make some assumptions of the system workload. The most fundamental one is that flow durations in the Internet have long and heavy tails [146, 56, 183, 23], as shown in Figure 1.6. It shows how fewer than 10% of the flows in a backbone link carry over 90% of the bytes transported in the link. There are, thus, two types of user flows: most flows are short; and then a few are very, very long and carry most of the bytes. These long flows may hog the system for extended periods of time and degrade the overall system performance significantly. All these flow characteristics will be incorporated in the performance study of packet and circuit switching in the core of the network.

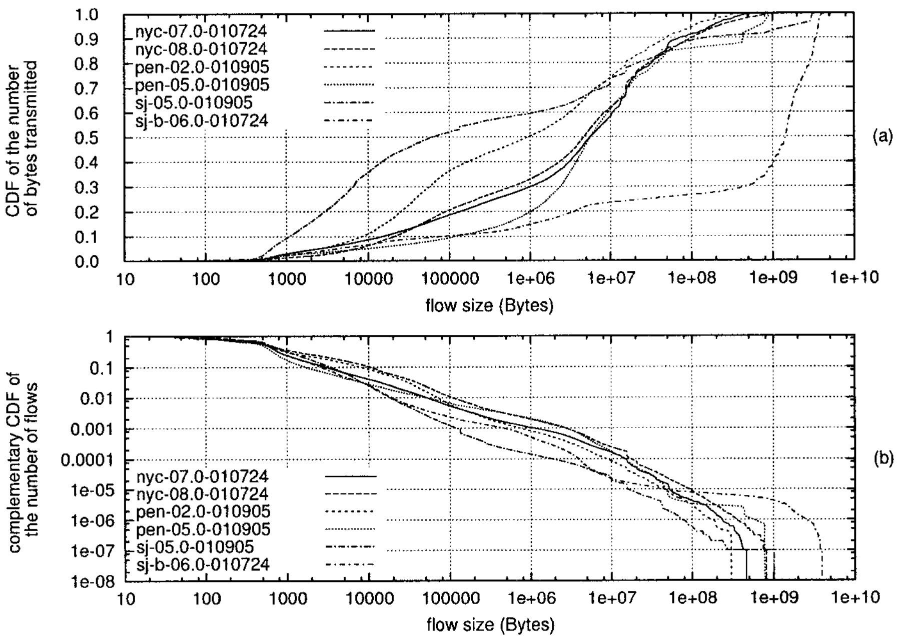

Figure 1.6: Heavy-tailed traffic. The figure shows (a) the empirical cumulative distribution function of the number of bytes transported, and (b) the empirical complementary cumulative distribution function of the flow size frequency with respect to the flow size in bytes. Each of the six backbone traces has between 5 and 40 million flows of different OC-48 links, and they expand over a period of more than 24 hours [170]. They include all types of flows (TCP, UDP and ICMP)

1.6 Organization of the Thesis

Often one can find in the technical press and literature predictions about how IP routers will eventually replace all circuit switches [186, 150, 21, 99, 35, 37, 142, 87, 156, 128, 110]. These articles extend the original arguments for adopting packet switching in the early days of the Internet (namely, efficiency and robustness), by adding the simplicity, cost advantage, and ability to provide QoS of IP. These are some of the *sacred cows* of IP, and in Chapter 2, I evaluate them one by one to demystify the ones that do not hold up to scrutiny and to identify the ones that really apply.

One key claim of packet switching is left for Chapter 3; namely that the statistical multiplexing of packet switching consistently delivers a lower response time than circuit switching while downloading information. This is indeed the most relevant performance metric for end users, and thus it gets its own chapter.

The conclusion of these two chapters is that packet switching is very attractive in Local Area Networks (LANs) and access networks, because of the poor end-user response time of dynamic circuits. On the contrary, circuit switching is more attractive in the core of the network because of its higher capacity, its perfect QoS, and a response time that is similar to that of packet switching. In the future, one can expect a dominant role of IP in the edge of the network, whereas various forms of circuit switching will dominate the core of the network. This partially validates the hybrid network architecture that we currently have and that is shown in Figure 1.5.

However, in the current Internet these two distinct parts are completely decoupled; the edges switch packets independently of the circuits used in the core. Chapter 4 presents a network architecture (TCP Switching) that allows the integration of circuit switching in the core of a packet-switched Internet in an evolutionary way. The chapter starts with a description of what a typical application-level flow in the Internet is, as observed on access points to the Internet of several universities and research institutions. A key observation is that despite the connectionless nature of IP, our use of the network is very connection oriented, and this fits well with the use of circuits. TCP Switching is based on the idea of IP Switching [129], and it maps each application flow to a lightweight circuit. This proposal encompasses a family of solutions, with several design choices. Also in this chapter, I choose one solution based on what constitutes a typical flow in the Internet.

One potential problem with such fine-grain circuits in the core, as thin as 56 Kbit/s, is that they might not fit well with many circuit switch designs. Most core circuit switches have interconnects that only use channels of at least 51 Mbit/s. In addition, optical switches only forward wavelengths carrying channels of over 2.5 Gbit/s. The signaling of these switches might be heavyweight because of the slow reconfiguration of the switch fabrics or because of a signaling mechanism that requires circuit creation confirmation. In Chapter 5, I present another technique for controlling the coarse-grain, heavyweight circuits in the core by monitoring user flows rather than tracking packets or queue lengths. I show the requirements for different circuit setup times. These results could be used in Generalized Multi-Protocol Label Switching (GMPLS), a technique that uses heavyweight circuits to adapt the network capacity dynamically between edge routers.

In Chapter 6, I describe some of the related work in the area of high speed switching in the core of the network. Some proposals include the use of circuit switching in the core (GMPLS [7], OIF [13], ODSI [53], Zing [181]), while others attempt to extend packet switching to all-optical switches (Optical Packet Switching – OPS [186] –, and Optical Burst Switching - OBS [188]). Some emphasis is placed on the comparison of TCP Switching with OPS and OBS. Two metrics are used for the comparison: the loss and blocking probabilities for a given network load, and the complexity of the overall network.

Chapter 7 concludes the Thesis, restating how we would benefit from more circuit switching in the core of the network, and how we could integrate this circuit switched core with the rest of the network in an evolutionary way.

Chapter 2

Circuit and Packet Switching

2.1 Introduction

It is widely assumed that, for reasons of efficiency, the various communication networks (Internet, telephone, TV, radio, ...) will merge into one ubiquitous, packet-switched network that carries all forms of communications. This view of the future is particularly prevalent among the Internet community, where it is assumed that packet-switched IP is the layer over which everything else will be carried. In this chapter, I present evidence so as to argue that this will not happen. This stance is controversial, and is difficult to make concrete, as any attempt to compare the various candidates for the transport infrastructure[1] is fraught with lack of data and the difficulty of making apples-with-apples comparisons. Therefore, the evidence presented here is different from other chapters in this thesis. Observations, case studies, and anecdotal data (rather than controlled experiments, simulations and proofs) are used to take a stance and to predict how the network architecture will evolve.

Whatever the initial goals of the Internet, two main characteristics seem to account for its success: *reachability* and *heterogeneity*. IP, the packet-switching protocol that is the basis for the Internet, provides a simple, single, global address to reach every host, enables unfettered access between all hosts and adapts the topology to restore

[1] In this chapter, transport is used in the sense of the infrastructure over which many service networks run, not in the sense of the OSI protocol layer.

reachability when links and routers fail. IP hides heterogeneity in the sense that it provides a single, simple service abstraction that is largely independent of the physical links over which it runs. As a result, IP provides service to a huge variety of applications and operates over extremely diverse link technologies.

The growth and success of IP has given rise to some widely held assumptions amongst researchers, the networking industry and the public at large. One common assumption is that it is only a matter of time before IP becomes the sole global communication infrastructure, dwarfing, and eventually displacing, existing communication infrastructures such as telephone, cable and TV networks. IP is already universally used for data networking in wired networks (enterprise networks and the public Internet), and is being rapidly adopted for data communications in wireless and mobile networks. IP is also increasingly used for both local and long-distance voice communications, and it is technically feasible for packet-switched IP to replace SONET/SDH.

A related assumption is that IP routers (based on packet switching and datagram routing) will become the most important, or perhaps only, type of switching device inside the network. This is based on our collective belief that packet switching is inherently superior to circuit switching because of the efficiencies of statistical multiplexing and the ability of IP to route around failures. It is widely assumed that IP is simpler than circuit switching and should be more economical to deploy and manage. And with continued advances in the underlying technology, we will no doubt see faster and faster links and routers throughout the Internet infrastructure. It is also widely assumed that IP will become the common convergence layer for all communication infrastructures. All communication services will be built on top of IP technology. In addition to information retrieval, we will stream video and audio, place phone calls, hold video-conferences, teach classes, and perform surgery.

On the face of it, these assumptions are quite reasonable. Technically, IP is flexible enough to support all communication needs, from best-effort to real-time. With robust enough routers and routing protocols, and with extensions such as weighted fair queueing, it is possible to build a packet-switched, datagram network that can support any type of application, regardless of their requirements.

In spite of all the strengths of IP, this chapter will argue how it will be very hard for IP to displace existing networks. It will also conclude how many of the assumptions discussed above are not supported by reality, and do not stand up to close scrutiny.

The goal of this is to question the assumption that IP will be *the* network of the future. The conclusion is that if we started over - with a clean slate - it is not clear that we would argue for a universal, packet-switched IP network. In the future, more and more users and applications will demand predictability from the Internet, both in terms of the availability of service and the timely delivery of data. IP was not optimized to provide either, and so it seems unlikely to displace networks that already provide both. In this chapter, I take the position that while IP will be the network layer of choice for best-effort, non-mission critical and non-real-time data communications (such as information exchange and retrieval), it will live alongside other networks, such as circuit-switched networks, that are optimized for high revenue time-sensitive applications that demand timely delivery of data and guaranteed availability of service.

This is indeed a controversial position. Nevertheless, as researchers we need to be prepared to take a step back, to take a hard look at the pros and cons of IP, and its likely future. As a research and education community, we need to start thinking how IP will co-exist and co-operate with other networking technologies.

2.1.1 Organization of the chapter

Section 2.2 provides a more detailed description of circuit switching and packet switching than in Chapter 1. It also describes part of the earlier work on these two switching techniques. Section 2.3 dissects some of the claims about IP, especially when compared to circuit-switched networks. This section tries to demystify those claims that do not hold up to scrutiny. Section 2.4 discusses the implications for the network architecture. Section 2.5 concludes this chapter.

2.2 Background and previous work

Before starting our discussion about whether IP can be the basis of all communication networks, I will give some background about the two main switching techniques in use today: circuit switching and packet switching.

2.2.1 Circuit switching

Circuit switching was the first switching technique used in communication networks because it is simple enough to carry analog signals. This thesis will just focus on the digital version of circuit switching. Of course, the main example of its use is the phone system [72], but it is also used in the core of the Internet in the form of SONET/SDH and DWDM equipment [81, 126]. In circuit switching, the transmission medium is typically divided into channels using Frequency Division Multiplexing (FDM), [2] Time Division Multiplexing (TDM) or Code Division Multiplexing (CDM) [172]. A circuit is a string of concatenated channels from the source to the destination that carriers an information flow.[3]

To establish the circuits, a signaling mechanism is used. This signaling only carries control information, and it is considered an overhead. It is also the most complex part in circuit switching, as all decisions are taken by the signaling process. It is commonly assumed that the signaling and per-circuit state management make circuit switches hard to design, configure and operate.

In circuit switching the channel bandwidth is reserved for an information flow. To ensure timely delivery of the data, the capacity of the circuit has to be at least equal to the peak transmission rate of the flow. In this case, the circuit is said to be peak allocated, and then the network offers a connection-oriented service with a perfect quality of service (QoS) in terms of delay jitter and bandwidth guarantees, However, this occurs at the cost of wasting bandwidth when sources idle or simply slow down.

Contention only occurs when allocating channels to circuits during circuit/call

[2] (Dense) Wavelength Division Multiplexing, (D)WDM, is a subclass of FDM that uses optical wavelengths as channels.

[3] Note that the source and the destination need not be edge nodes. They can be aggregation nodes in the middle of the network that combine several user flows into one big information flow.

establishment. If there are not enough channels for the request, the call establishment may be delayed, blocked or even dropped. In contrast, once the call is accepted, resources are not shared with other flows, eliminating any uncertainty and, thus, removing the need for buffering, processing or scheduling in the data path. When circuits are peak allocated, the only measure of Quality of Service (QoS) in circuit switching is the blocking probability of a call.

To summarize, circuit switching provides traffic isolation and traffic engineering, but at the expense of using bandwidth inefficiently and signaling overhead. It is often said that these two drawbacks make circuit switching highly inflexible, especially in a highly dynamic environment such as the Internet. I will argue in this that these drawbacks are outweighed by the advantages of using more circuit switching in the core of the network.

2.2.2 Packet switching

Packet switching is the basis for the Internet Protocol (IP) [152, 172]. In packet switching, information flows are broken into variable-size packets (or fixed-size cells as in the case of ATM). These packets are sent, one by one, to the nearest router, which will look up the destination address, and then forward them to the corresponding next hop. This process is repeated until the packet reaches its destination. The routing of the information is thus done locally, hop-by-hop. Routing decisions are independent of other decisions in the past and in other routers; however, they are based on network state and topology information that is exchanged among routers using BGP, IS-IS or OSPF [148]. The network does not need to keep any state to operate, other than the routing tables.

The forwarding mechanism is called store-and-forward because IP packets are completely received, stored in the router while being processed, and then transmitted. Additionally, packets may need to be buffered locally to resolve contention for resources. [4] If the system runs out of buffers, packets are dropped.

With the most scheduling policies, such as FCFS and WFQ, packet switching

[4] Resources have contention when they have more arrivals/requests that what they can process. Two examples are the outgoing links and the router interconnect.

remains work conserving; it keeps the link busy as long as there are packets waiting to be sent. This allows it to have a statistical multiplexing gain; that is, the capacity of an outgoing link can be much smaller than the sum of its tributaries and still have a packet delay or drop probability within certain statistical bounds. This gain is higher when traffic is more bursty. The buffering needs and the statistical multiplexing are the main characteristics of packet switching, and they will be crucial in its comparison with circuit switching.

In the Internet, the network service is connectionless and best effort; that is, it provides no delivery guarantees. Reliability, flow control and connection-oriented services are provided by end-to-end mechanisms, such as with TCP [153]. Because the underlying service is best effort, there are no guarantees in terms of packet drops, maximum delay, delay jitter or bandwidth.

Much research was done in the early days of computer networking comparing circuit switching, packet switching and message switching (a variant of packet switching, in which the whole information flow is treated as a single switching unit) [96, 10, 164, 97, 175, 95]. Most of the work was done in the context of packet radio, satellite, and local area networks and shows how in these environments packet switching provided higher throughput for a given bound on the average delay. Packet switching not only made an effective use of the network bandwidth, but it also was robust and resilient to node and link failures.

Later work on different scheduling algorithms and signaling mechanisms, such as Weighted Fair Queueing (WFQ) [62], Generalized Processor Sharing (GPS) [141], Differentiated Services (DiffServ) [16], Integrated Services (IntServ) [20] and Deficit Round Robin (DRR) [113], showed how packet switching can also provide QoS guarantees if the admission of new flows to the network can be controlled.

2.3 IP Folklore

This section tries to identify some folkloric assumptions about IP and the Internet, and it examines each in turn. I will start with the most basic assumption, and the easiest to dispel: that the Internet *already* dominates global communications. This is not true by any reasonable metric: market size, number of users, or the amount of traffic. Of course, this is not to say that the Internet will not grow over time to dominate the global communications infrastructure; after all, the Internet is still in its infancy. It is possible — and widely believed — that packet-switched IP datagrams will become the *de-facto* mechanism for all communications in the future. And so one has to consider the assumptions behind this belief and verify whether packet-switched IP offers inherent and compelling advantages that will lead to its inevitable and unavoidable dominance. This requires the examination of some "sacred cows" of networking; for example, that packet switching is more efficient than circuit switching, that IP is simpler, it lowers the cost of ownership, and it is more robust when there are failures in the network.

2.3.1 IP already dominates global communications

It has been reported that the Internet already carries more traffic than the phone system [122, 162], and that the difference in traffic volume will become bigger and bigger over time because Internet traffic is growing at a rate of 100% per annum versus a rate of 5.6% per year for voice traffic [48].

Despite this phenomenal success of the Internet, it is currently only a small fraction of the global communication infrastructure, which consists of separate networks for telephones, broadcast TV, cable TV, satellite, radio, public and private data networks, and the Internet. In terms of revenue, the Internet is a relatively small business. The US business and consumer-oriented ISP markets have revenues of $13B each (2000) [28, 29], in contrast, the TV broadcast industry has revenues of $29.8B (1997), the cable distribution industry $35.0B (1997), the radio broadcast industry $10.6B (1997) [180], and the phone industry $268.5B (1999), of which $111.3B correspond to long distance and $48.5B to wireless [88]. The Internet reaches 59% of US households [133], compared to 94% for telephones and 98% for TV [127, 147]. Even though Internet traffic doubles every year, revenues only increase 17% annually (2001) [162], whereas long-distance phone revenues increase 6.7% per year (1994-97) [136]. If these growth rates were kept constant, IP revenues would not surpass those of the long-distance phone industry until 2017.[5]

If we restrict our focus to the data and telephony infrastructure, the core IP router market still represents a small fraction of the public infrastructure, contrary to what happens in the private enterprise data networks. As shown in Table 2.1, the expenditure on core routers worldwide was \$1.7B in 2001, compared to \$28.0B for transport circuit switches. So in terms of market size, revenue, number of users, and expenditure on infrastructure, it is safe to say that IP does not currently dominate the global communications infrastructure.

Segment	Market size
Core routers	\$1.7B
Edge routers	\$2.4B
SONET/SDH/WDM	\$28.0B
Telecom MSS	\$4.5B

Table 2.1: World market breakup for the public telecommunications infrastructure in 2001 [161, 158, 159, 157].

Figure 2.1 illustrates the devices currently used in the public Internet. The current communication infrastructure consists of a transport network — made of circuit-switched SONET/SDH and DWDM devices — on top of which run multiple service networks. The service networks include the voice network (circuit-switched), the IP network (datagram, packet-switched), and the ATM/Frame Relay networks (virtual-circuit-switched). Notice the distinction between the circuit-switched transport network, which is made of SONET/SDH and optical switches that switch coarse granularity ($n \times STS-1$, where an STS-1 channel is 51 Mbit/s), and the voice service circuit switches, which include Class 4 and Class 5 systems that switch 64Kbps voice circuits and handle various telephony-related functions. When considering whether IP has or will take over the world of communications, one needs to consider both the transport and service layers. In other words, for universal packet transport I am considering using a packet network to replace the transport infrastructure; and for

[5] It is interesting to note that for IP revenues to surpass those of long-distance telephony the Internet revenue per household would have to multiply by 358%.

voice-over-IP (VoIP) I am considering an application built on top of an IP network that replaces the traditional Class 4/5 TDM voice switches.

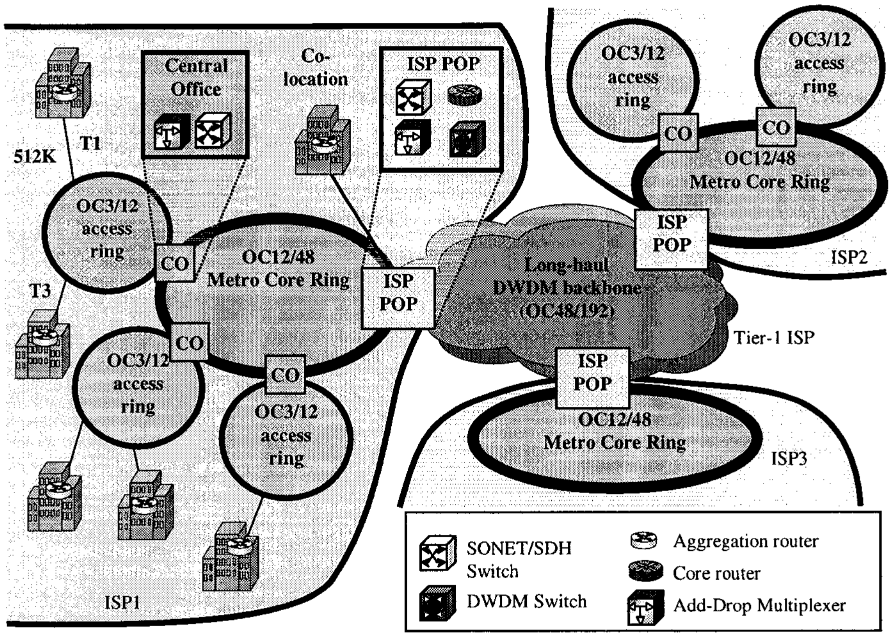

Figure 2.1: Architecture of the public Internet. There are also many large private voice and data networks that consist of IP routers, LAN switches and voice switches at customer premises.

In order to examine the merits of a packet-switched IP network, one needs to compare it with an alternative. The obvious alternative is circuit switching. In one respect, this is not an apples-with-apples comparison; the packet-switched IP data network today already operates over a circuit-switched transport infrastructure. If we consider only the core of the network, we find essentially a central core of circuit switching surrounded by IP routers. It helps to think of the comparison as a question as to which one of two outcomes is more likely: Will the packet-switched IP network grow to dominate and displace the circuit-switched transport network, or will the (enhanced) circuit-switched TDM and optical switches continue to dominate the core transport network?

2.3.2 IP is more efficient

> "Analysts say [packet-switched networks] can carry 6 to 10 times the traffic of traditional circuit-switched networks." — **Business Week.**

From the early days of computer networking, it has been well known that packet switching makes efficient use of scarce link bandwidth [10]. With packet switching, statistical multiplexing allows link bandwidth to be shared by all users, and work-conserving link sharing policies (such as FCFS and WFQ) ensure that a link is always busy when packets are queued-up waiting to use it. In contrast, with circuit switching, each flow is assigned its own channel, so a channel could go idle even if other flows are waiting. Packet switching (and thus IP) makes more efficient use of the bandwidth than circuit switching, which was particularly important in the early days of the Internet when long haul links were slow, congested and expensive.

It is worth asking: What is the current utilization of the Internet, and how much does efficiency matter today? Odlyzko and others [135, 47, 90, 23] report that the core of the Internet is heavily overprovisioned, and that the average link utilization in links in the core is between 3% and 20% (compared to 33% average link utilization in long-distance phone lines [135, 160]). The reasons that they give for low utilization are threefold: First, Internet traffic is extremely asymmetric and bursty, but links are symmetric and of fixed capacity; second, it is difficult to predict traffic growth in a link, so operators tend to add bandwidth aggressively; third, with falling prices for coarser bandwidth granularity as faster technology appears, it is more economical to add capacity in large increments.

There are other reasons to keep network utilization low. When congested, a packet-switched network performs badly, becomes unstable and can experience oscillations and synchronization. Many factors contribute to this. Complex and dynamic interaction of traffic means that congestion in one part of the network will spread to other parts. Further, the control packets (such as routing packets) are transmitted *in-band* in the Internet, and hence they are more likely to get lost and delayed when the data-path is congested. When routing protocol packets are lost or delayed due to network congestion or control processor overload, it causes an inconsistent routing state, and may result in traffic loops, black holes, and disconnected regions of the network, which further exacerbate congestion in the data path [107, 55]. Currently, the most effective way for network providers to address these problems is by preventing congestion and keeping network utilization low.

But perhaps the most significant reason that network providers overprovision their network is to give low packet delay. Users want predictable behavior, which means low queueing delay, even under abnormal conditions (such as the failure of several links and routers) [90, 77]. As users, we already demand (and are willing to pay for) huge overprovisioning of Ethernet networks (the average utilization of an Ethernet network today is about 1% [47]) simply so that we do not have to share the network with others, and so that our packets can pass through without queueing delay. We will demand the same behavior from the Internet as a whole. We will pay network providers to stop using statistical multiplexing and to instead overprovision their networks. The demand for lower delay will drive providers to decrease link utilization even more than it is today.

Therefore, even though in theory a statistical multiplexed *link* can potentially yield a higher network utilization and throughput, in practice, to maintain a consistent performance and reasonably stable *network*, network operators significantly overprovision their network, thus keeping the network utilization low.

But simply reducing the *average* link utilization will not be enough to make users happy. For a typical user to experience low utilization, the *variance* of the network utilization also needs to be low. There are two flavors of variance that affect the perceived utilization: variance in time (short-term increases in congestion during busy times of the day), and variance by location (while most links are idle, a small number are heavily congested). If we pick some users at random and consider the network utilization their traffic experiences, our sample is biased in favor of users who find the network to be heavily congested. This explains why, as users, we know the average utilization to be low, but find that we often experience long queueing delays.

Reducing variations in link utilization is hard. Without sound traffic management and traffic engineering, the performance, predictability and stability of large IP networks deteriorate rapidly as load increases. Today, we lack effective techniques to reduce the unpredictability of performance introduced by variations in link utilization. It might be argued that the problem will be solved by research efforts on traffic management and congestion control (to control and reduce variations in time), as well as work on traffic engineering and multipath routing (to load-balance traffic over a number of paths). But to date, despite these problems being understood for many years, effective measures are yet to be introduced.

We can expect that over time users will demand lower and lower queueing delays in the Internet. This means that as users, we collectively want network providers to stop using statistical multiplexing and to instead overprovision their networks *as if they were circuit switched* [115, 137, 77]. To date, network providers have responded to our demands by overprovisioning, by publishing delay measurements for their network, and by competing on the basis of these numbers. In the long term, the demand for lower delay will drive providers to make link utilization even lower than it is today, and network utilization will continue to decrease as the world economy becomes more dependent on the Internet.

One can take the demand for low delay one step further, and ask whether users experience the lowest response times in a packet-switched network. Intuition suggests that packet switching will lead to lower delay: A packet-switched network easily supports heterogeneous flow rates, and flows can always make forward progress because of processor sharing in the routers. In practice, it does not make much difference whether packet switching or circuit switching are used. This is studied in detail in Chapter 3, which (by analysis and simulation) studies the effect of replacing the core of the network with dynamic fine-granularity circuit switches, as described in Chapter 4. Let's define the user response time as the time it takes from when a user requests a file until this file finishes downloading. Web browsing and file sharing represent over 65% of Internet transferred bytes today [31], and so the request/response model is representative of typical user behavior. Now consider two types of network: one is the current packet-switched network in which packets share links and each flow makes constant, albeit slow, forward progress over congested links. The other network is a hypothetical comparison. Each new application flow triggers the creation of a low bandwidth circuit in the core of the network, similar to what happens in the phone network. If there are no circuits available, the flow is blocked until a channel is free. As we will see in Chapter 3, at the core of the network, where the rate of a single flow is limited by the data-rate of its access link, simulations and analysis suggest that the average user response time of both techniques is the same, independent of the flow length distribution.

In summary, even though packet switching can lead to more efficient link utilization, unpredictable queueing delays force network operators to operate their networks very inefficiently. One can conclude that while efficiency was once a critical factor, it is so outweighed by our need for predictability, stability, immediate access, and low delay that network operators will be forced to run their networks very inefficiently. Network operators have already concluded this; they know that their customers care more about predictability than efficiency, and we know from the dynamics of queueing networks, that in order to achieve predictable behavior, network operators must continue to utilize their links very lightly, forfeiting the benefits of statistical multiplexing. As a result, they are paying for the extra complexity of processing every packet in routers, without the benefits of increased efficiency. In other words, the original goal of "efficient usage of expensive and congested links" is no longer valid, and it would provide no benefit to users.

2.3.3 IP is robust

> "The Internet was born during the cold war 30 years ago. The US Department of Defence [decided] to explore the possibility of a communication network that could survive a nuclear attack." — **BBC**

The Internet was designed to withstand a catastrophic event in which a large number of links and routers were destroyed. This goal is in line with users and businesses who rely more and more on network connectivity for their activities and operations, and who want the network to be available at all times. Much has been claimed about the reliability of the current Internet, and it is widely believed to be inherently more robust and capable of withstanding failures of different network elements. Its robustness comes from using soft-state routing information; upon a link or router failure, it can quickly update the routing tables and direct packets around the failed element. In contrast, a circuit-switched network needs to reroute all affected active circuits, which can be a large task for a high-speed link carrying hundreds or thousands of circuits.

The reliability of the current Internet has been studied by Labovitz et al. [107]. They have studied different ISPs over several months, and report a median network availability equivalent to a downtime of 471 min/year. In contrast, Kuhn [102] found that the average downtime in phone networks is less than 5 min/year. As users, we have all experienced network downtime when our link is unavailable or some part of the network is unreachable. On occasions, connectivity is lost for long periods while routers reconfigure their tables and converge to a new topology. Labovitz et al. [106] also observed that the Internet recovers slowly, with a median BGP convergence time of 3 minutes, and frequently taking over 15 minutes. In contrast, SONET/SDH rings, through the use of pre-computed backup paths, are required to recover in less than 50 ms [51], a glitch that is barely noticeable to the user in a network connection or phone conversation.

While it may be argued that the instability and unreliability of the Internet can be attributed to its rapid growth and the ad-hoc and distributed way that it has grown, a more likely explanation is that it is fundamentally more difficult to achieve robustness and stability in packet networks than circuit networks. In particular, since routers/switches need to maintain a distributed routing state, there is always the possibility that the state may become disconnected. In packet networks, inconsistent routing state can generate traffic loops and black holes and disrupt the operation of the network. In addition, as discussed in Section 2.3.2, the likelihood of a network getting into a inconsistent routing state is much higher in IP networks because (a) the routing packets are transmitted in-band, and therefore are more likely to incur congestion due to high load of user traffic; (b) the routing computation in IP networks is very complex; it is, therefore, more likely for the control processor to be overloaded; (c) the probability of misconfiguring a router is high. And misconfiguration of even a single router may cause instability in a large portion of the network. It is surprising that we have continued to use routing protocols that allow one badly behaved router to make the whole network inoperable [105]. Conversely, high availability has always been a government-mandated requirement for the telephone network, and so steps have been taken to ensure that it is an extremely robust infrastructure. In circuit networks, control messages are usually transmitted over a separate channel or network. This has the added advantage of security for network control and management. In addition, the routing in circuit networks is much simpler.

In datagram networks, inconsistent routing state may cause black holes or traffic loops so that the service to existing user traffic is disrupted – i.e., inconsistent routing is *service impacting.* In circuit networks, inconsistent routing state may result in unnecessary rejection of request for new circuits, but none of the established circuits is affected. In summary, currently with IP, not only are failures more common, but also they take longer to be repaired and their impact on users is deeper.

On the face of it, then, it seems that packet-switched IP networks experience more failures and take longer to re-establish connectivity. However, it is not clear that reliability and fault tolerance are a direct consequence of our choice of packet switching or circuit switching. One can attribute much of the growth of the Internet to the ad-hoc and distributed way that it has grown; so it should not be surprising that there are frequent misconfigurations of routers and poorly maintained equipment [114]. Table 2.2 shows that router operations are the most common source of network failures.

The key point here is that there is nothing inherently unreliable about circuit switching, and there is an existence proof that it is both possible and economically viable to build a robust circuit-switched infrastructure, that is able to quickly reconfigure around failures. There is no evidence yet that we can define and implement the dynamic routing protocols to make the packet-switched Internet as robust. Perhaps the problems with BGP will be fixed over time and the Internet will become more reliable. But it is a mistake to believe that packet switching is inherently more robust. In fact, the opposite may be true.

Type of failure	Frequency of occurrence	description
Router Operations	36.8 %	Maintenance, power failures, congestion
Link Failure	34.1 %	Fiber cuts, unreachable, interface down
Router Failures	18.9 %	Hardware and software problems, routing problems, malicious attacks
Undefined	10.5%	Miscellaneous and unknown

Table 2.2: Frequency of occurrence of recorded network failures in a regional ISP in a one-year period [107].

2.3.4 IP is simpler

> "IP-only networks are much easier and simpler to manage, leading to improved economics." — **Business Communications Review**

It is an oft-stated principle of the Internet that the complexity belongs at the end-points, so as to keep the routers simple and streamlined. While the general abstraction and protocol specification are simple, implementing a high performance router and operating an IP network are extremely challenging tasks.

In terms of router complexity, while the general belief in the academic community is that it takes 10's of instructions to process an IP packet, the reality is that the complexities of a high performance router has as much to do with the forwarding engine as with the routing protocols (BGP, IS-IS, OSPF etc), where all the intelligence of the IP layer resides, as well as the interactions between the routing protocols and forwarding engine. A high performance router is extremely complex, particularly as the line rates increase. One subjective measure of the complexity is the failure rate of the start-ups in this space. Because of the perceived high growth of the market, a large number of well-financed start-ups with very capable talents and strong backing from carriers have attempted to build high performance routers. Almost all have failed or are in the process of failing— putting aside the business/market-related issues, none have succeeded technically and delivered a product-quality core router. The core router market is still dominated by two vendors, and many of the architects of one came from the other. The bottom line is that building a core router is far from simple, mastered by only a very small group of people.

If we are looking for simplicity, then we would do well to look at how circuit-switched transport switches are built. First, the software is simpler. The software running in a typical transport switch is based on about three million lines of source code [154], whereas Cisco's Internet Operating System (IOS) is based on eight million [66], over twice as many. Routers have a reputation for being unreliable, crashing frequently and taking a long time to restart, so much so that router vendors frequently compete on the reliability of their software, pointing out the unreliability of their competitor's software as a marketing tactic. Even a 5ESS service telephone switch from Lucent, with its myriad of features for call establishment and billing, has only about twice the number of lines of code as a core router [179, 67].

The hardware in the forwarding path of a circuit switch is also simpler than that of a router, as shown in Figure 1.1 and Figure 1.2. At the very least, the line card of a router must unframe/frame the packet, process its header, find the longest-matching prefix that matches the destination address, generate ICMP error messages for expired TTLs, process optional headers, and then buffer the packet (a buffer typically holds 250ms of packet data). If multiple service levels are added (for example, differentiated services), then multiple queues must be maintained, as well as an output link scheduling mechanism. In a router that performs access control, packets must be classified to determine whether or not they should be forwarded. Further, in a router that supports virtual private networks, there are different forwarding tables for each customer. A router carrying out all these operations typically performs the equivalent of 500 CPU serial instructions per packet (and we thought that all the complexity was in the end system!).

On the other hand, the linecard of an electronic transport switch typically contains a SONET framer to interface to the external line, a chip to map ingress time slots to egress time slots, and an interface to a switch fabric. Essentially, one can build a transport linecard (Figure 1.2) by starting with a router linecard (Figure 1.1) and then removing most of the functionality.

One measure of this complexity is the number of logic gates implemented in the linecard of a router. An OC192c POS linecard today contains about 30 million gates in ASICs, plus at least one CPU, 300 Mbytes of packet buffers, 2 Mbytes of forwarding table, and 10 Mbytes of other state memory. The trend in routers has been to put more and more functionality on the forwarding path: initially, support for multicast (which is rarely used), and now support for quality of service, access control, security and VPNs.[6] In contrast, the linecard of a typical transport switch contains a quarter of the number of gates, no CPU, no packet buffer, no forwarding table, and an on-chip state memory (included in the gate count).

In terms of power consumption, a high-end router dissipates 75% of the power in the linecards, half of which comes from inter-chip I/O communication. IP linecards require many chips, and thus they consume much power. The use of Ternary Content Addressable Memories (TCAMs) for parallel route lookups further exacerbates this power consumption. In contrast, electronic circuit switches consume less power because they use simpler hardware, allowing more linecards (and thus more capacity) to be placed in a single rack.

It should come as no surprise that the highest capacity commercial transport switches have two to twelve times the capacity of an IP router, and sell for about half to one twelfth the price per gigabit per second, as shown in Table 1.1. So, even if packet switching might be simpler for low data rates, it becomes more complex for high data rates. IP's "simplicity" does not scale.

One might argue that the reason the circuit switches cost less is that they solve a simpler problem. Instead of being aware of individual application flows, they deal with large trunk lines in multiples of 51 Mbit/s. So for the sake of comparison, it is worth considering the cost and complexity of building a core transport switch that could establish a new circuit for each (TCP) application flow. Let's assume that each user connects to the network via a 56 Kbit/s modem; this will define the granularity

[6] Interestingly, these features are added to provide traffic isolation and engineering, features that are intrinsic to circuit switching.

of the switch. While such a small circuit might not be the best way to incorporate circuit switching into the Internet, using such small flow granularity provides an upper bound on the complexity of doing so. A 10 Gbit/s linecard needs to manage at most 200,000 circuits of 56 Kbit/s. The state required to maintain the circuits, and the algorithms needed to quickly establish and remove circuits, would occupy only a fraction of one ASIC. This suggests that the hardware complexity of a circuit switch will always be lower than the complexity of the corresponding router.

It is interesting to explore how optical technology will affect the performance of routers and circuit switches. In recent years, there has been a good deal of discussion about all-optical Internet routers. As was mentioned in Chapter 1, there are two reasons why this is not feasible. First, a router is a packet switch and so inherently requires large buffers to hold packets during times of congestion, and currently no economically feasible ways exist to buffer large numbers of packets optically. The buffers need to be large because TCP's congestion control algorithms currently require at least one bandwidth-delay product of buffering to perform well. For a 40 Gbit/s link and a round-trip time of 250 ms, this corresponds to 1.3 GBytes of storage, which is a large amount of electronic buffering and (currently) an unthinkable amount of optical buffering. The second reason that all-optical routers do not make sense is that an Internet router must perform an address lookup for each arriving packet. Neither the size of the routing table, nor the nature of the lookup, lends itself to implementation using optics. For example, a router at the core of the Internet today must hold over 100,000 entries, and must search the table to find the longest matching prefix — a non-trivial operation. There are currently no known ways to do this optically.

Optical switching technology is much better suited to circuit switches. Devices such as tunable lasers, MEMS switches, fiber amplifiers and DWDM multiplexers provide the technology to build extremely high capacity, low power circuit switches that are well beyond the capacities possible in electronic routers [15].

In summary, packet switches and IP linecards have to perform more operations on the incoming data. This requires more chips, both for logic functions and buffering; in addition, these chips are more complex. In contrast, circuit switches are simpler, which allows them to have higher capacities and to be implemented in optics.

2.3.5 Cost of ownership of IP is small

> "Packet technology is just inherently much less expensive and more flexible than circuit switches." — CTO of Sonus.

IP networks are usually marketed as having a lower cost of ownership than the corresponding circuit-switched network, and so they should displace circuit switching from the parts of the network that it still dominates; however, this has not (yet) happened. For example, Voice over IP (VoIP) promises lower communication costs because of the statistical multiplexing gain of packet switching and the sharing of the physical infrastructure between data and voice traffic. Despite these potential long-term cost savings, less than 6% of all international traffic used VoIP in 2001 [38, 98]. VoIP has become less attractive because fierce competition among phone companies has dramatically driven down the prices of long-distance calls [26]. In addition, the cost savings of a single infrastructure can only be realized in new buildings.

One of the most important factors in determining a network architecture is the total cost of ownership. Given two options with equivalent technical capabilities, the least expensive option is the one that gets deployed in the long term. So, in order to see whether IP will conquer the world of communications, one needs to answer this question: Is there something inherent in packet switching that makes packet-switched networks less expensive to build and operate? Here, the metric to study is the total cost per bit/s of capacity.

As we saw in Section 2.3.1, the market for core routers is much smaller than that of circuit switches. One could argue that the market difference is because routers are far less expensive than circuit switches and that carriers are stuck into supporting expensive legacy circuit-switched equipment; however, IP, SONET/SDH and DWDM reached maturity almost at the same time,[7] so a historical advantage does not seem to be a valid explanation for the market sizes. A more likely explanation is that there are simply more circuit switches than routers in the core because routers are

[7] In April 1995, commercial Internet was born after the decommissioning of the NSFnet. In March 1994, Sprint first announced its deployment of directional SONET rings. The first deployments of WDM were from June 1996.

not ready to take over the transport infrastructure, and thus the market size cannot be used as a good indication of the equipment cost.

To analyze the total cost of packet and circuit switching, I will start breaking down the cost structure of an ISP. Table 2.3 shows the capital expenditure (capex), operation expenses (opex) and transport costs (interconnection fees) of an Internet carrier [184]. Similar numbers are found in [119].

Routing/switching equipment (capex)	20%
Network management and staff (opex)	45%
Transport/transmission	35%

Table 2.3: Cost structure for an Internet carrier averaged over ten tier-1 and tier-2 ISPs in the US and Europe [184].

Capital expenditure is the cost to build a network. Because there is little difference in the links and link terminations in routers and circuit switches, the difference in capital expenditure lays in the cost of the boxes. Production and design costs are related to the complexity of the system. Figures 1.1 and 1.2 show how routers need more components, and these are more complex, and thus routers are more expensive to design and produce. It should not be surprising that an OC192c packet-over-SONET (POS) linecard for a router costs $30-40K, whereas the equivalent SONET TDM linecard costs only $10-20K. If we consider that linecards are the most expensive part of a full router/switch, it is fair to say that it is more expensive to build a router than a circuit switch of the same capacity.

Anyhow, capital expenditure is the smaller part of the pie, and operating expenses represent the biggest cost factor for an ISP. To grasp the importance of the latter, let me point out to a study by McKinsey and Goldman-Sachs [118] that shows that unless per-bit operating expenses are reduced 25%-30% per year through 2005, no reasonable amount of per-bit capital expenditure reduction will allow carriers to achieve sustainable Return on Invested Capital (ROIC). However, this reduction in operating cost is not easy to achieve, as operating expenses are difficult to quantify, and their reduction may have a direct impact on the service quality.

Certainly there seems no reason to believe that IP networks are simpler to operate and maintain. Indeed, a report by Merrill Lynch [121] shows that the normalized operating expenditure for data networking is typically significantly larger than for voice networks. If we look at the number of network administrators present in most companies, usually there are far more operators for the IP network than for the phone network.[8]

Operating expenses are tied to the reliability, manageability and complexity of the network, and IP does not seem to win in any of these three fronts: First, as argued in Section 2.3.3, IP has not demonstrated to be as reliable as SONET/SDH, and thus requires more attention. Second, Internet management platforms are rudimentary and lack integration and interoperability, and tools for capacity planning, traffic engineering and monitoring are almost non-existent in IP [184, 118]. Finally, as mentioned in Chapter 1 and Section 2.3.4 routers do not scale as well as circuit switches in terms of switching capacity. Consequently, one needs more routers than circuit switches to carry the same traffic. This creates a more complex network that is more expensive to build, harder to control and with more network elements demanding attention from operators.

However, there is one area in which IP can potentially reduce costs. IP networks require less network capacity to carry the same information (especially when traffic is bursty) because of the statistical multiplexing gain of packet switching. However, as we saw in Section 2.3.2, carriers do not take advantage of this characteristic of IP, and they prefer to operate their networks at very low utilization, as to ensure the reliability of their network.

To summarize, packet-switched networks seem to be more expensive to build and operate than circuit-switched networks. While some of the causes for the high costs of IP may be addressed in the future (better router software and software tools), others will remain (more complex boxes, less scalable routers). Nevertheless, IP is more flexible than circuit switching, and so there is a tradeoff between cost and flexibility. It is up to the carriers to decide when the need for flexibility justifies the extra cost of packet switching.

[8] Stanford University (with a population of about 15,000 people) employs 80 full-time telephone engineers, 25 full-time IP network engineers, and 350 part-time local IP network administrators.

2.3.6 Support of telephony and other real-time applications over IP networks

> "All critical elements now exist for implementing a QoS-enabled IP network." — **IEEE Communications Magazine**

There is a widely-held assumption that IP networks can support telephony and other real-time applications that require minimum guaranteed bandwidth, bounded delay jitter and limited loss. If one looks more closely, one finds that the reasons for such an optimistic assumption are quite diverse. One school holds the view that IP is ready today. There are two reasons for such a belief. First, IP networks are and will continue to be heavily overprovisioned, and the average packet delay in the network will be low enough to satisfy the real-time requirements of these applications. Second, most interesting real-time applications, including telephony, are *soft* real-time in the sense that they can tolerate occasional packet delay/loss and *adapt* to these network variabilities. While today's IP networks are heavily overprovisioned, it is doubtful whether a new solution (far from complete yet) that provides a worse performance can displace the reliable and high quality service provided by today's TDM-based infrastructure (which is already paid-for).

Another school believes that for IP to succeed, it is critical for IP to provide Quality of Service (QoS) with the same guarantees as TDM but with more flexibility. In addition, the belief is that there is no fundamental technical barrier to build a connection-oriented service (Tenet [75] and IntServ [20]) and to provide guaranteed services in the Internet. The technical ingredients for a complete solution include efficient packet classification and scheduling algorithms. Unfortunately, after more than ten years of extensive research and efforts in the standards bodies, the prospect of end-to-end per-flow QoS in the Internet is nowhere in sight. The difficulty seems to be the fact that there is huge culture gap between the connection and datagram design communities. By blaming the failure on "connections", a third school holds the view that a simpler QoS mechanism such as DiffServ is the right way to go. Again, we are several years into the process, and it is not at all clear that the "fuzzy" QoS provided by DiffServ (with no route pinning support and no per flow QoS scheduling) will be good enough for customers who are used to the simple QoS provided by the existing circuit-switched transport networks.

The truth is that many of these QoS mechanisms, such as DiffServ and IntServ, are implemented in most routers deployed in the Internet; however, few service providers enable them and use them. The reasons are that these mechanisms are difficult to understand and configure and that they require an active cooperation among ISPs for them to provide end-to-end QoS.

Finally, no matter what technology we intend to use to carry voice over the Internet, there are few financial incentives to do so. As Mike O'Dell[9] recently said [134]: "[to have a Voice-over-IP (VoIP) service network one has to] create the most expensive data service to run an application for which people are willing to pay less money everyday [...] and for which telephony already provides a better solution with a marginal cost of almost zero." The result is that despite the promised cost reductions of Voice over IP, in 2001 less than 6% of all international voice traffic out of the US used VoIP.

On the other hand, because circuits are peak-allocated, circuit switching provides simple (and somewhat degenerate) QoS, and thus there is no delay jitter. The user (or server) can inform the network of a flow's duration, and specify: its desired rate and blocking probability (or a bound on the time that a flow can be blocked). These measures of service quality are certainly simpler for users to understand and for operators to work with, than those envisaged for packet-switched networks.

2.4 Discussion

Up until this point, I have considered some of the folklore surrounding the packet-switched Internet. The overall goal is to provoke discussion and research on fundamental issues that need to be addressed so that IP can continue to revolutionize the world of communications. As a research community, we need to think beyond the daily challenges of maintaining and optimizing the expanding Internet, and move on

[9] Former Senior Vice President of UUNET, responsible for technical strategic direction and architecture of the network.

to consider the enormous challenges that lie ahead.

It seems that there are two main limitations to the widespread adoption of IP: dependability and the right way for IP to co-exist with circuits. In what follows, I will discuss each in turn.

2.4.1 Dependability of IP networks

High dependability, in the broadest sense, is a must if IP is to become a successful transport technology (to compete or displace circuit-based transport networks), and if the Internet is to become the universal infrastructure for high value applications. For example, voice services are a high-revenue, and very profitable business. Trusting them to today's unreliable, and unpredictable IP networks would be an unnecessary risk, which is why — despite predictions to the contrary — telephone carriers have not done so.

High dependability means several things: robustness and stability, traffic isolation, traffic engineering, fault isolation, manageability, and last but not least, the ability to provide predictable performance in terms of bounded delay and guaranteed bandwidth (QoS). In its current form, the Internet excels in none of these areas. Although it is clearly a challenge to achieve each of these goals, they must all be solved for IP to become dependable enough for use as a transport mechanism.

2.4.2 Interaction of IP and circuits

The current Internet is based on packet-switched routers in the edges, interconnected by a circuit-switched transport network. Given the benefits of circuit switching, it would seem perverse for the packet-switched network to grow to subsume the transport network. It is inconceivable that the network providers would remove the existing, robust, reliable, predictable and largely paid-for transport network, and replace it with a technology that seems more complex, less reliable, more expensive and not yet installed.

What seems more likely is that packet switching will continue to exist at the edge of the network, aggregating and multiplexing traffic from heterogeneous sources for applications that have no delay or quality requirements. In other words, packet-switched IP will continue to provide a simple service abstraction for a variety of applications. However, this does not preclude the existence of highly specialized service networks living alongside IP and using other switching techniques. In fact, it is unlikely that the phone or TV cable service networks will be completely replaced by an IP network any time soon as it would require a huge amount of capital to build a new network.

At the core of the network, one can expect the circuit-switched transport network to remain as a means to interconnect the packet-switched routers and as a means to provide high reliability and performance guarantees. Over time, more and more optical technology will be introduced into the transport network, leading to capacities that (necessarily) electronic routers cannot achieve.

One remaining question is whether or not the circuit-switched network will be controlled by IP. In other words, will the IP network decide dynamically when to create new circuits between routers? For example, a router could monitor the occupancy of its queues or the number of active flows and periodically add or remove circuits to other routers based on current demand [7, 181]. Such a system has the benefit of enabling IP to gain the benefits of fast optical circuit switches in the core, yet maintain the simple service model for heterogeneous sources at the edge.[10]

However, while a complete control by IP of the circuit-switched backbone seems appealing to IP, one needs to remember that the majority of the revenue for the circuit switches will still be from other applications, such as voice. Since the packet-switched network is unlikely to provide the predictability needed for voice traffic, it will continue to operate over its own, separate circuit-switched edge network and to be carried over the shared transport network at the core. In this environment, it is unlikely that the routers will be allowed to control the entire capacity of the transport switches, unless the revenue for the Internet exceeds that of telephony. At the current growth rates, it will take over 15 years for data traffic to surpass telephony as the main source of revenue in telecommunications. In the future, it is more likely that the routers will be allocated a fraction of the circuit-switched transport infrastructure,

[10] Chapter 4 and Chapter 5 describe two ways of integrating IP and circuit switching in the core.

which they can control and adapt to best serve their needs.

With the dynamic control of circuit-networks (possibly by an IP-based control plane), it is also conceivable that the IP routers at the edge can signal to the transport network to dynamically create new circuits or change the bandwidth of existing circuits.

2.4.3 What if we started with a clean slate?

In the preceding discussion, an outcome was depicted based on historical conditions, in the context of a pre-existing circuit-switched transport network. So if we started again, with the benefit of hindsight, would we build a network with circuit switching at the core, and packet switching at the edge? I believe that we would, and that it would look something like this:

- **Addressing scheme.** A simple, unique and universal addressing scheme (like IP's) would allow us to communicate with any sort of device or application anywhere in the world. This addressing scheme defines the routing algorithms in the intermediate network nodes, but it is completely independent of the forwarding or switching mechanisms that they use.

- **Switching in the edges of the network.** Packet switching would be used in the edges of the network as well as in those links where bandwidth is scarce (such as some satellite and wireless links, and underwater cables). The reasons for this are threefold. First, packet switching makes a very efficient use of the bandwidth in these cases. Second, as will be emphasized in Chapter 3, it can greatly improve the end-user response time by borrowing all available link bandwidth when other users are not active. Finally, packet switches can be cost effective for lower link rates. The packet-switched network should ideally gather traffic from disparate sources, and multiplex it together in preparation for carriage over a very high capacity, central, circuit-switched core. In this environment, local switching at the edge of the network is an optimization that may or may not be necessary. Without it, the packet-switched network is simply a hierarchy of statistical multiplexers, with little or no forwarding decisions. All traffic can be multiplexed towards the core, then demultiplexed again towards the edge. While less efficient, it provides a simplified environment in which to deploy the delay guarantees needed by telephony. And so it might be feasible to carry the traffic from access voice switches to the core over the statistically multiplexed edge network.

- **Switching in the core of the network.** At the core of the network, there seem a number of compelling reasons to use circuit switching. First, circuit switching has already demonstrated its robustness and its ability to quickly recover from failures. Circuit switching is inherently simpler than packet switching, requiring less work to forward data, and consequently will cost less as a result, will consume less power, and will take up less space. Last, but not least, circuit switching provides an easy way to adopt the huge potential of high capacity optical switches. Without electronics on the forwarding path, one can expect optical switches to provide abundant capacity at low cost.

- **Integration of both switching mechanisms.** Rather than working independently, both these mechanisms would be tightly integrated, in such a way that an action in one provokes an appropriate reaction in the other. For example, packet switching would have to export the QoS and connection-oriented nature of the circuit-switched core to the applications that require it. On the other hand, circuit switching has to respond to the increases in activity of packet switching, by adapting its capacity among core/edge gateways accordingly. Additionally, we will find more hybrid switches that can do both circuit and packet switching, serving as gateways between the two worlds. Chapter 4 and Chapter 5 describe two ways of bridging packet switching and circuit switching. Finally, the idea of using circuit switching to interconnect distant routers can also be extended to using a circuit-switched crossconnect to interconnect the packet-switched linecards of a router.

2.5 Conclusions and summary of contributions

While it is technically pleasing to believe that IP will dominate all forms of communication, our delight in its elegance is making us overlook its shortcomings. IP is an excellent means to exchange data, which explains its success. This chapter has demystified some of the proclaimed advantages of IP, such as the claims that IP is simpler, more robust, more efficient, that it dominates world communications, and that it can support QoS-aware applications. I have reserved the rebuttal of what is probably the most important claim for next chapter; namely, that IP can achieve better response time for the end user.

IP remains ill suited as a means to provide many other types of service, and is too crude to form the transport infrastructure in its own right. To allow the continued success of IP, we must be open-minded to it living alongside, and co-operating with, other techniques (such as circuit switching) and protocols that are optimized to different needs.

The conclusion is that while packet-switched IP will continue to dominate best-effort data services at the edge of the network, the core of the network will use circuit switching as a transport platform for multiple services. Circuit switching allows the construction of networks with very high capacity, scalability, flexibility, self-healing, reliability and auto-adaptation to current network traffic conditions; thus, IP will have a hard time replacing the circuit switching that already exists in the core. We should instead start thinking of how to integrate the two technologies: circuit switching in the core and packet switching in the edges.

Chapter 3

Response Time of Circuit and Packet Switching

3.1 Introduction

As we saw in Chapters 1 and 2, packet switches (routers) do not offer any significant advantages with respect to circuit switches in terms of simplicity, robustness, cost-efficiency, or quality of service (QoS). In addition, circuit switches scale better in terms of switching capacity than routers, and it is possible to develop circuit switches with an all-optical data path because they do not have the buffering and per-packet processing requirements of routers. As a result, circuit switching can be used to close the current gap between the growth rates of traffic demand and router capacity. All this indicates that circuit switching should be a good candidate for the core of the Internet — where capacity is needed the most.

We could indeed benefit from using more circuit switching in the core of the network; however, we need to answer two questions first: How would the network perform as far as the end-user is concerned if there were circuits at the core? And how do we introduce circuit switching at the core (not the edge) of the Internet in an evolutionary way?

In Chapters 4 and 5, I will concentrate on the second question, by proposing two approaches for integrating circuit and packet switching, and analyzing their feasibility.

This chapter concentrates on the first question. In particular, it looks at the response time seen by end users in the extreme case in which each application flow at the edge of the network triggers a new circuit at the core (this is called TCP Switching). TCP Switching exploits the fact that most data communications are connection-oriented, and, thus, existing connections can easily be mapped to circuits in the backbone. Despite its name, TCP Switching works with any application flow, and so it also works with less common UDP flows, as well as ICMP and DNS messages. I recommend the reader to also read the next chapter, as it provides more information about the problem and some discussion of the salient advantages and disadvantages of TCP Switching. However, Section 3.5 provides enough information about TCP Switching for the purposes of this chapter, and so it is not necessary to have read the next chapter to understand the performance evaluation done here.

However, it is not the purpose of this chapter to argue how good or bad TCP Switching is in terms of its implementation or ease of integration into the current Internet. Instead, this chapter explores how the Internet would perform if it included a significant amount of fine-grain circuit switching. In particular, the goal is to examine the obvious question (and preconception): Won't circuit switching lead to a much less efficient Internet because of the loss of statistical multiplexing? And, consequently, doesn't packet switching lead to lower costs for the operator and faster response times for the users? While I am not necessarily arguing that TCP Switching is the best way to introduce circuit switching into the core of the Internet, it is possible to analyze this extreme approach. The results of this chapter are not limited to TCP Switching, and they should give us an indication of how any dynamic circuit switching technique will perform as far as the user is concerned and whether increased deployment of circuit switching (in optical or electronic forms) makes sense.

In Chapter 2, we already saw how QoS-aware applications can benefit from the simpler and clearer QoS definitions of circuit switching. However, the most important performance metric for the end user is currently the *response time* of a flow, defined as the time from when a user requests a file from a remote server until the last byte of that file arrives. This metric is so relevant because the most common use of the Internet today is to download files,[1] whether they are web pages, programs, images, songs, or videos. After modeling and simulating the response time of equivalent packet- and circuit-switched systems, this chapter concludes that, while circuit switching does not make much sense for the local area or access network due to its poor response time in that environment, there would be little change in end-user performance if we introduced circuit switching into the core of the Internet. Given the relevant advantages of circuit switching that were described in Chapter 2 (namely, the higher capacity of circuit switches, their higher reliability, their lower cost, and their support for QoS), one can conclude that we would clearly benefit from more circuit switching in the core of the Internet.

3.1.1 Organization of the chapter

This chapter is solely devoted to the study of the most important end-user metric, the response time. Section 3.2 describes some early work on the response time of packet switching. Then, Section 3.3 analyzes the response time in LANs and shared access networks; it starts with two motivating examples, one in which circuit switching outperforms packet switching, and one in which packet switching outperforms circuit switching. I then use a simple analytical model derived from an M/GI/1 queueing system to determine the conditions under which one technique outperforms the other. Special emphasis is given to flow-size distributions that are heavy tailed, such as the ones found in the Internet. Section 3.4 performs an analysis similar to that in Section 3.3, but for the core of the network. These analytical results do not include many network effects that may affect the response time, and so Section 3.5 uses ns-2 simulations to validate the results for the core. Section 3.7 concludes this chapter.

3.2 Background and previous work

Early work in the late 60s and in the 70s [96, 10, 164, 97, 175, 95] studied the response time of packet and circuit switching in the context of radio networks, satellite

[1] Web browsing and file sharing represent over 65% of Internet transferred bytes today [31].

communications and the ARPANET (the precursor of the modern Internet). These three network scenarios had something in common: links had less capacity than the bandwidth that an end host could process, and so a single end host could take all link bandwidth along a path if no one else was using it. The conclusion of this early work was that packet switching is more efficient than circuit switching, and it provides better response time under these scenarios. These results were obtained using M/M/N queueing models, where arrivals are Poisson and service times are exponential.

With time, these results have been extrapolated to form part of the IP folklore despite the fact that much has changed in the Internet. First, a single end host is no longer capable of filling up a link in the core (2.5 Gbit/s and above) on its own. Second, it has been shown that whereas flow/session arrivals are Poisson (or close to Poisson) [78, 45], flow sizes are not exponential, but rather heavy-tailed, and thus they are closer to a Pareto distribution than an exponential one [84, 57, 183]. This chapter evaluates the end-user response time with consideration of the characteristics of the current Internet.

3.3 LANs and shared access networks

I will start with some examples to illustrate what may happen when circuit or packet switching is used. I will use a simple example to demonstrate a scenario under which circuit switching leads to improved user response time, and one in which the opposite is true.

3.3.1 Example 1: LANs with fixed-size flows

Consider the network in Figure 3.1 with 100 clients in the East Coast of the US all trying simultaneously to download a 10-Mbit file[2] from a remote server that is connected to a 1 Gbit/s link. With packet switching, all 100 clients share the link bandwidth in a fair fashion. They receive 1/100 of the 1 Gbit/s, and so all 100 clients will finish at the same time, after 1 sec. On the other hand, with circuit switching

[2] For purposes of this chapter, I will define "1 Mbit" to be equal to 10^6 bits, not 2^{10} bits, in order to simplify our examples.

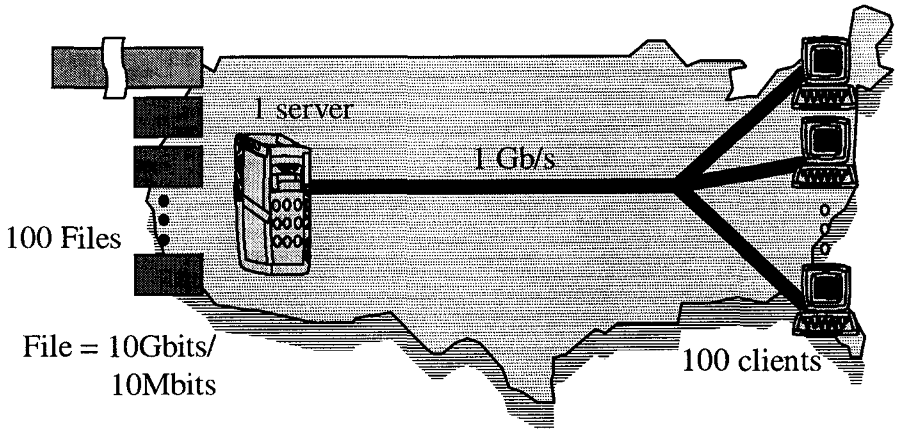

Figure 3.1: Network scenario for both motivating examples. The bottleneck link is a transcontinental link of 1 Gbit/s. In the first scenario, all files are of the same length (10 Mbits). In the second scenario, the first file that gets requested is 1000 times longer (with a size of 10 Gbits) than the other 99 files (of 10 Mbits).

and circuits of 1 Gbit/s, the average response time is 0.505 sec, half as much as for packet switching. Furthermore the worst client using circuit switching performs as well as all the clients using packet switching, and all but one of the clients (99% in this case) are better off with circuit switching. We observe that in this case it is better to complete each job one at a time, rather than making slow and steady progress with all of them simultaneously. It is also worth noting that, even when circuits are blocked for some time before they start, they all finish before the packet-switched flows that started earlier, and it is the finishing time that counts for the end-user response time. This result is reminiscent of the scheduling in operating systems where the *shortest remaining job first* policy leads to the fastest average job completion time.

	Circuit switching	Packet switching
Flow bandwidth	1 Gbit/s	10 Mbit/s
Average response time (s)	0.505	1
Maximum response time (s)	1	1

Table 3.1: Average and maximum response times in Example 3.3.1

3.3.2 Example 2: LANs with heavy-tailed flow sizes

This second example demonstrates a scenario under which packet switching leads to improved response time. Consider the previous scenario, but assume that one client starts downloading a much longer file of 10 Gbits slightly before the others. With circuit switching, this client hogs the link for 10 sec, preventing any of the other flows from making progress. So, the average response time for circuit switching is 10.495 sec versus just 1.099 sec for packet switching. In this case all but one of the clients are better off with packet switching. Since active circuits cannot be preempted, the performance of circuit switching falters as soon as a big flow monopolizes the link and prevents all others from being serviced. With packet switching, one long flow can slow down the link, but it will not block it.

	Circuit switching	Packet switching
Flow bandwidth	1 Gbit/s	10 Mbit/s, later 1 Gbit/s
Average response time (s)	10.495	1.099
Maximum response time (s)	10.99	10.99

Table 3.2: Average and maximum response times in Example 3.3.2

Which scenario is more representative of the Internet today? I will argue that the second scenario (for which packet switching performs better) is similar to the edge of the network (i.e., LAN and access networks) because as we will see flow sizes in the Internet are not constant and they follow a heavy-tailed distribution. However, I will also argue that neither scenario represents the core of the Internet. This is because core links have much higher capacity than edge links, and so a single flow cannot hog the shared link. A different model is needed to capture this effect. But first, I will consider a simple analytical model of how flows share links at the edge of the network.

3.3.3 Model for LANs and access networks

I start by modeling the average response time for parts of the network where a single flow can fill the whole link. Below, I use a simple continuous time queueing model for packet and circuit switching to try and capture the salient differences between them. I will use an M/GI/1 queue. The model assumes that traffic consists of a sequence of jobs, each representing the downloading of a file. Performance is assumed to be dominated by a single bottleneck link of capacity R, as shown in Figure 3.2. A service policy decides the order in which data is transferred over the bottleneck link. To model packet switching, we assume the service policy to be *Processor Sharing (PS-PrSh)*, and so all jobs share the bottleneck link equally, and each makes progress at rate R/k, where k is the number of active flows. To model circuit switching, we assume that the server takes one job at a time and serves each job to completion, at rate R, before moving onto the next.

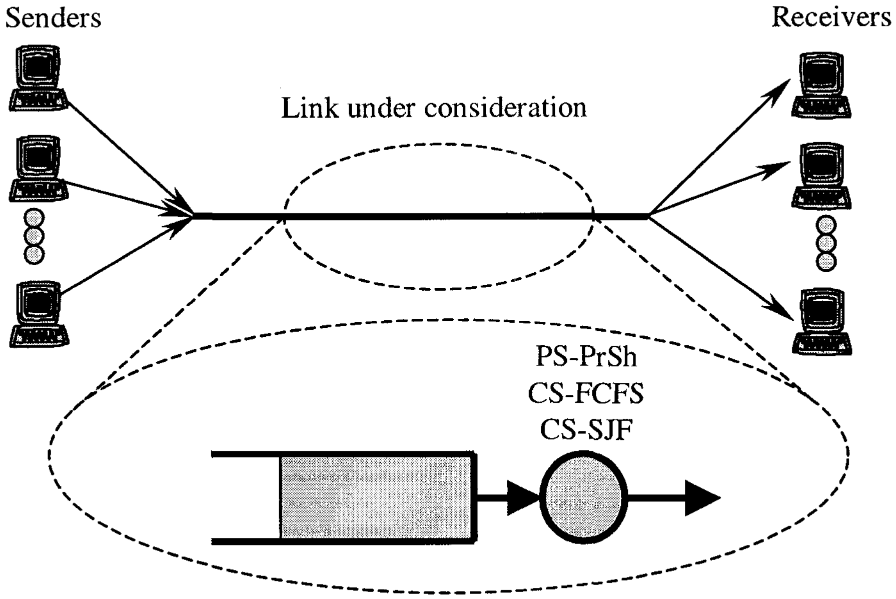

Figure 3.2: Queueing model used to analyze a bottleneck link using circuit and packet switching.

The circuit-switching model is non-preemptive, modeling the behavior of a circuit that occupies all the link bandwidth when it is created, and which cannot be preempted by another circuit until the flow finishes. To determine the order in which flows occupy the link, we consider two different service disciplines: *First Come First*

Serve (CS-FCFS) and *Shortest Job First (CS-SJF)*. It is well known [96] that CS-SJF has the smallest average response time among all non-preemptive policies in an M/GI/1 system, and so CS-SJF represents the best-case performance for circuit-switching policies. However, CS-SJF requires knowledge of the amount of work required for each job. In this context, this means the router would need to know the duration of a flow before it starts, which is information not available in a TCP connection or other types of flows. Therefore, CS-FCFS is considered a simpler and more practical service discipline, since it only requires a queue to remember the arrival order of the flows.

In our model, flows are assumed to be reactive and able to immediately adapt to the bandwidth that they are given. The model does not capture real-life effects such as packetization, packet drops, retransmissions, congestion control, etc., all of which will tend to increase the response time. This model can be seen as a benchmark that compares how the two switching techniques fare under idealized conditions. Later, the results will be corroborated in simulations of full networks.

The average response time, $E[T]$, as a function of the flow size, X, is [96, 52]:

For M/GI/1/PS-PrSh:

$$E[T] = E[X] + E[W] = E[X] + \frac{\rho}{1-\rho}E[X] \tag{3.1}$$

for M/GI/1/CS-FCFS:

$$E[T] = E[X] + E[W] = E[X] + \frac{\rho}{1-\rho} \times \frac{E[X^2]}{2E[X]} \tag{3.2}$$

for M/GI/1/CS-SJF:

$$E[T] = E[X] + \frac{\rho E[X^2]}{2E[X]} \int_0^{\infty} \frac{f(x)dx}{(1 - \frac{\rho}{E[X]}\int_0^{x^+} yf(y)dy)(1 - \frac{\rho}{E[X]}\int_0^{x^-} yf(y)dy)} \tag{3.3}$$

where $0 \leq \rho < 1$ is the system load, W is the waiting time of a job in the queue, and $f(x)$ is the distribution of the flow sizes.

To study the effect of the flow size variance, I use a bimodal distribution for the flow sizes, X, such that $f(x) = \alpha\delta(x - A) + (1 - \alpha)\delta(x - B)$, with $\alpha \in (0, 1)$. A is held constant to 1500 bytes, and B is varied to keep the average job size, $E[X]$, (and thus the link load) constant.

Figure 3.3 shows the average response time for circuit switching (for both CS-SJF and CS-FCFS) with respect to that for PS-PrSh for different values of α and for link loads of 25% and 90%. A value of below 1 indicates a faster response time for circuit switching, a value above 1 shows that packet switching is faster.

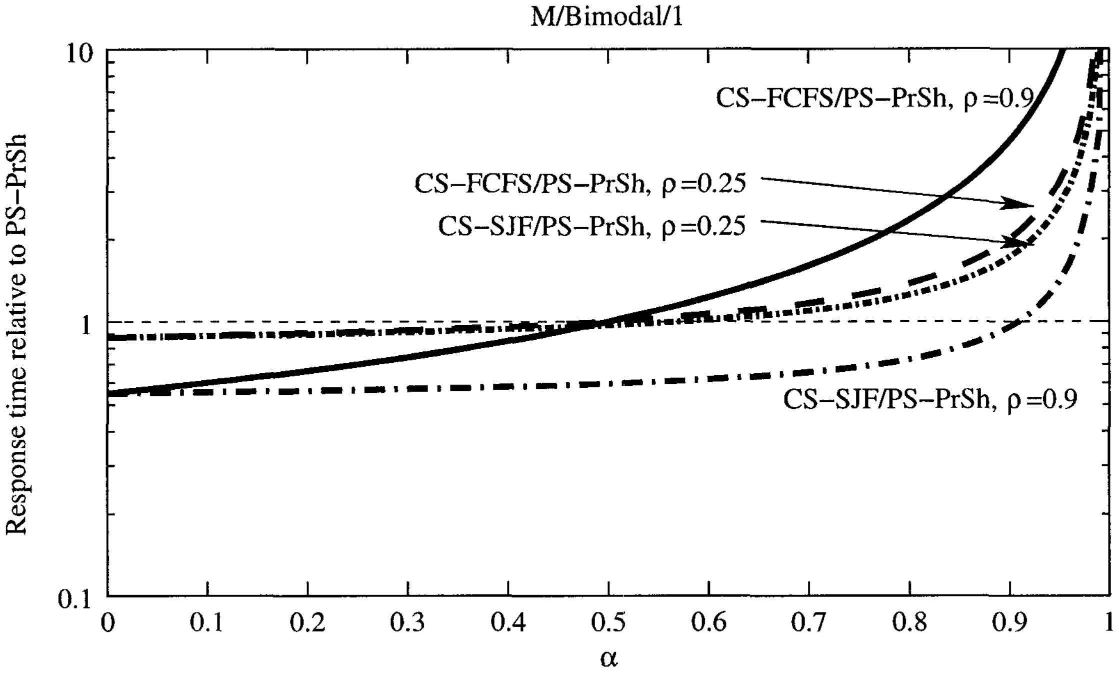

Figure 3.3: Relative average response time of CS-FCFS and CS-SJF with respect to PS-PrSh for a single server. Arrivals are Poisson and flow sizes are bimodal with parameter α. Link loads are $\rho = 0.25$ and $\rho = 0.9$.

The figure is best understood by revisiting the motivating examples in sections 3.3.1 and 3.3.2. When α is small, almost all flows are of size $B \approx E[X]$, and the flow size variance is small, $\sigma_X^2 = (E[X] - A)^2 \times \alpha/(1-\alpha) \ll (E[X])^2$. As we saw in the first example, in this case the average waiting times for both CS-FCFS and CS-SJF are about 50% of those for PS-PrSh for high loads.

On the other hand, when α approaches 1, most flows are of size A, and only a few are of size $B = (E[X] - \alpha A)/(1-\alpha) \longrightarrow \infty$. Then $\sigma_X^2 = (E[X] - A)^2 \times \alpha/(1-\alpha) \longrightarrow \infty$, and so the waiting time of circuit switching also grows to ∞. This case is similar to our second example, where occasional very long flows block short flows, leading to very high response time.

We can determine exactly when CS-FCFS outperforms PS-PrSh based on Equations 3.1 and 3.2. The ratio of their expected waiting time is $E[X^2]/(2E[X]^2)$, and so as long as the coefficient of variation $C^2 = E[X^2]/E[X]^2 - 1$ is less than 1, CS-FCFS always behaves better than PS-PrSh. On the other hand, CS-SJF behaves better than CS-FCFS, as expected, and it is able to provide a faster response time than PS-PrSh for a wider range of flow size variances, especially for high loads when the job queue is often non-empty and the reordering of the queue makes a difference. However, CS-SJF cannot avoid the eventual hogging by long flows when the job-size variance is high, and then it performs worse than PS-PrSh.

It has been reported [84, 57, 183] that the distribution of flow durations is heavy-tailed and that it has a very high variance, with an equivalent α greater than 0.999. This suggests that PS-PrSh is significantly better than either of the circuit-switching disciplines. I have further verified this conclusion using a Bounded-Pareto distribution for the flow size, such that $f(x) \propto x^{-\gamma-1}$. Figure 3.4 shows the results. It should not be surprising how bad circuit switching fares with respect to packet switching given the high variance in flow sizes of the Pareto distribution.

We can conclude that in a LAN environment, where a single end host can fill up all the link bandwidth, packet switching leads to more than 500-fold lower expected response time that circuit switching because of link hogging.

3.4 Core of the Internet

In the previous section, we saw what happens when a circuit belonging to a user flow blocks the link for long periods of time. However, this is not possible in the core of the network. For example, most core links today operate at 2.5 Gbit/s (OC48c) or above [30], whereas most flows are constrained to 56 Kbit/s or below by the access link [59]. Even if we consider DSL, cable modems and Ethernet, when the network

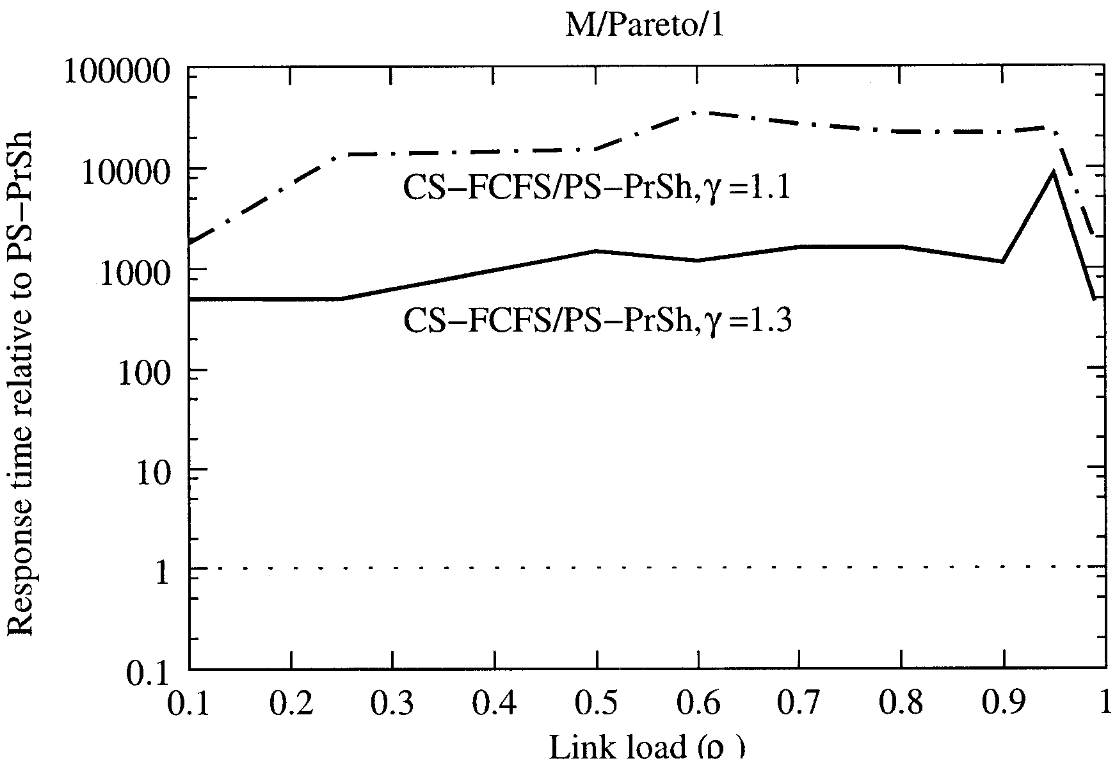

Figure 3.4: Relative average response time of CS-FCFS with respect to PS-PrSh for a single server. Arrivals are Poisson and flow sizes are bounded Pareto with parameter $\gamma = 1.1$ and $\gamma = 1.3$. Link loads are $\rho = 0.25$ and $\rho = 0.9$.

is empty a single user flow cannot fill the core link on its own. For this case, we need a different analysis.

3.4.1 Example 3: An overprovisioned core of the network

For the core of the network, I will consider a slightly modified version of the last example for LANs. Now, client hosts access the network through a 1 Mbit/s link, as shown in Figure 3.5. Again, the transcontinental link has a capacity of 1 Gbit/s, and there are 99 files of 10 Mbits and a single 10-Gbit file. In this case, flow rates are capped by the access link at 1 Mbit/s no matter what switching technology is used. With circuit switching, it does not make sense to allocate a circuit in the core that has more capacity than the access link because the excess bandwidth would be wasted. So, all circuits get 1 Mbit/s, and because the core link is of 1 Gbit/s, all 100 circuits of 1 Mbit/s can be admitted simultaneously. Similarly, we can fit all 100 packet-switched flows of 1 Mbit/s. If there is no difference in the flow bandwidth or the scheduling, then there is absolutely no difference in the response time of both techniques, as shown in Table 3.3. These results are representative of an overprovisioned core network.

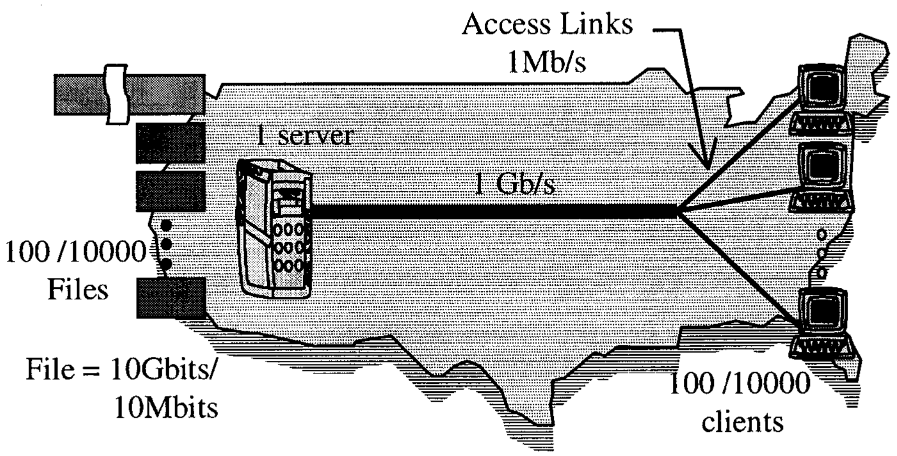

Figure 3.5: Network scenario for motivating examples 3.3.1 and 3.3.2. Access links of 1Mbit/s have been added, while the transcontinental link of 1 Gbit/s is kept the same. 1% of the files are long files of 10 Gbits, and the rest are only 10-Mbit long. In Example 3.4.1, there are only 100 clients, and in Example 3.4.2, 10000 clients.

	Circuit switching	Packet switching
Flow bandwidth	1 Mbit/s	1 Mbit/s
Average response time (s)	109.9	109.9
Maximum response time (s)	10000	10000

Table 3.3: Average and maximum response times in Example 3.4.1

3.4.2 Example 4: An oversubscribed core of the network

I will consider a fourth motivating example to illustrate what happens when we oversubscribe the core of the network. Consider the same scenario as before, but with 100 times as many clients and files; namely, we have 10000 clients requesting 9900 files of 10 Mbits and 100 files of 10 Gbits (which get requested slightly before the shorter ones). With circuit switching, all circuits will be of 1 Mbit/s again, and 100 Mbit/s of the core link will be blocked by the long flows for 10000 s, whereas short flows will be served in batches of 900 flows that last 10 s.

	Circuit switching	Packet switching
Flow bandwidth	1 Mbit/s	100 Kbit/s, later 1 Mbit/s
Average response time (s)	159.4	199.9
Maximum response time (s)	10000	10090

Table 3.4: Average and maximum response times in Example 3.4.2

With packet switching, all 10000 flows will be admitted, each taking 100 Kbit/s of the core link. After 100 s, all short flows finish, at which point the long flows get 1 Mbit/s until they finish. The long flows are unable to achieve 10 Mbit/s because the access link caps their peak rate. As a result, the average response time for packet switching is 199.9 s vs. the 159.4 s of circuit switching. In addition, the packet-switched system is not longer work conserving, and, as a consequence, the last flow finishes later with packet switching. The key point of this example is that by having more channels than long flows one can prevent circuit switching from hogging the link for long periods of time. Moreover, the oversubscription of a link with flows hurts packet switching because the flow bandwidth is squeezed.

Which of these two scenarios for the core is more representative of the Internet today? I will argue that it is Example 3.4.1 (for which circuit switching and packet switching perform similarly) because it has been reported that core links are heavily overprovisioned [135, 47, 90].

3.4.3 Model for the core of the Internet

At the core of the Internet, flow rates are limited by the access link rate, and so a single user cannot block a link on its own. To reflect this, the analytical model of Section 3.3.3 needs to be adjusted by capping the maximum rate that a flow can receive. I will use N to denote the ratio between the data rates of the core link and the access link. For example, when a flow from a 56 Kbit/s modem crosses a 2.5 Gbit/s core link, $N = 44,000$. Now, in the fluid model a single flow can use at most $1/N$ of the whole link capacity, so rather than having a full server, I will use N parallel servers, each with $1/N$ of the total capacity. In other words, I will use an M/GI/N model instead. For this model, there is an analytical solution for the PS-PrSh discipline [49]; however, there is no simple closed-form solution for CS-FCFS or CS-SJF, so I resort to simulation for these disciplines instead.

With circuit switching, the more circuits that run in parallel, the less likely it is that enough long flows appear at the same time to hog all the circuits. It is also interesting to note that CS-SJF will not necessarily behave better than CS-FCFS all the time, as CS-SJF tends to delay all long jobs and then serve them in a batch when there are no other jobs left. This makes it more likely for hogging to take place, blocking all short jobs that arrive while the batch of long jobs is being served. On the other hand, CS-FCFS spreads the long jobs over time (unless they all arrived at the same time), and it is therefore less likely to cause hogging. For this reason and because of the difficulties implementing CS-SJF in a real network, I will no longer consider it in our M/GI/N model.

Figure 3.6 compares the average response time for CS-FCFS against PS-PrSh for bimodal service times and different link loads. The ratio, N, between the core-link rate and the maximum flow rate varies from 1 to 512. We observe that as the number of flows carried by the core link increases, the performance of CS-FCFS improves and approaches that of PS-PrSh. This is because for large N the probability that there are more than N simultaneous long flows is extremely small. The waiting time becomes negligible, since all jobs reside in the servers, and so circuit switching and packet switching behave similarly. Figures 3.7a and 3.7b show similar results for bounded Pareto flow sizes as we vary the link load. Again, as N becomes greater or equal to

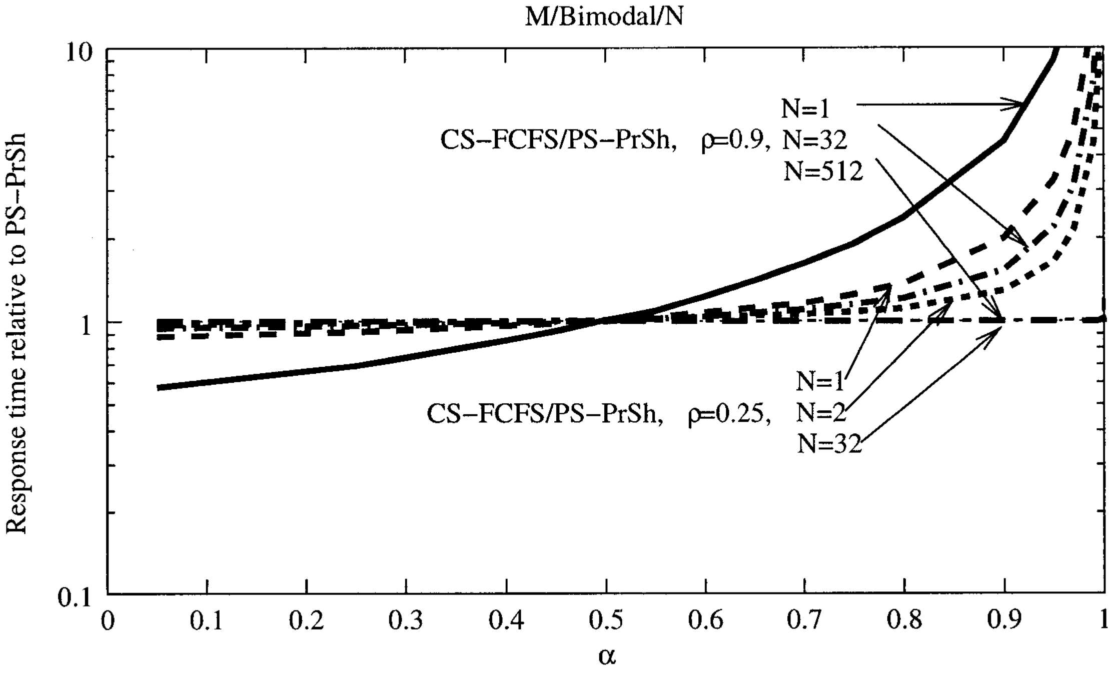

Figure 3.6: Relative average response time of CS-FCFS and CS-SJF with respect to PS-PrSh for an increasing core-to-access link-capacity ratio, N. Arrivals are Poisson and flow sizes are bimodal with parameter α. Link loads are $\rho = 0.25$ and $\rho = 0.9$. The value of N at which the curve flattens increases with the load, ρ, but it is smaller than 512 even for high loads.

512, there is no difference between circuit and packet switching in terms of average response time for any link load. I have also studied the standard deviation of the response time, σ_T, for both flow size distributions, and there is also little difference once N is greater than or equal to 512.

To understand what happens when N increases, we can study Figure 3.8. As we can see, the number of flows in the queue (shown in the upper three graphs) increases drastically whenever the number of long jobs in the system (shown in the bottom three graphs) is larger than the number of servers, which causes a long-lasting hogging. Until the hogging is cleared, there is an accumulation of (mainly) short jobs, which increases the response time. As the number of servers increases, the occurrence of hogging events is less frequent because the number of long flows in the

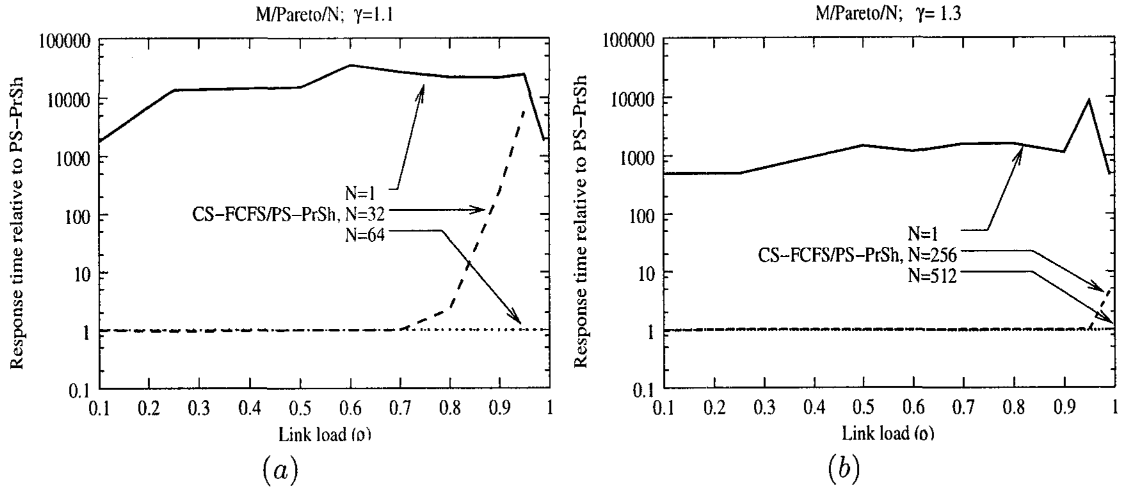

Figure 3.7: Relative average response time for CS-FCFS with respect to PS-PrSh for an increasing core-to-access link-capacity ratio, N. Arrivals are Poisson and flow sizes follow a bounded Pareto distribution with $\gamma = 1.1$ (a) and $\gamma = 1.3$ (b). The value of N at which the curves flatten is smaller than 512.

system is smaller than the number of servers, N, almost all the time. The results for an M/Bimodal/N system are very similar.

In the core, N will usually be very large. On the other hand, in metropolitan networks, N might be smaller than the critical N, at which circuit switching and packet switching have the same response time. Then, in a MAN a small number of simultaneous, long-lived circuits might hog the link. This could be overcome by reserving some of the circuits for short flows, so that they are not held back by the long ones, but it requires some knowledge of the duration of a flow when it starts. One way of forfeiting this knowledge of the flow length could be to accept all flows, and only when they last longer than a certain threshold, they are classified as long flows. However, this approach has the disadvantage that long flows may be blocked in the middle of the connection.

One might wonder why a situation similar to Example 3.4.2 does not happen. In that example, there were many more active flows than the ratio N, and then the packet-switched flows would squeeze the available bandwidth. However, if we consider Poisson arrivals, the probability that there are at least N arrivals during the duration

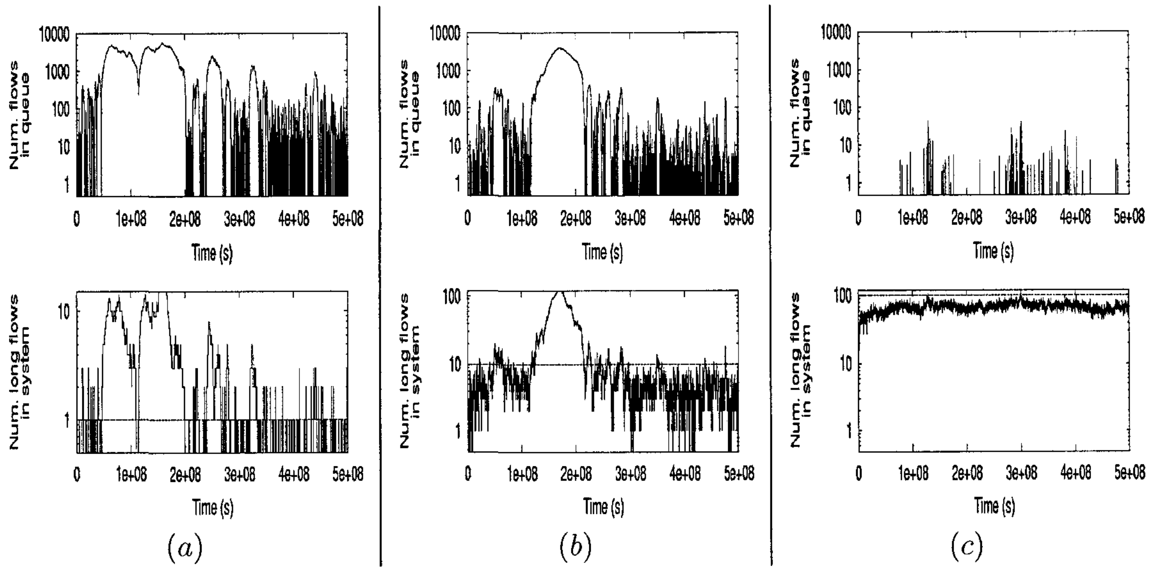

Figure 3.8: Time diagram of three M/Pareto/N/CS-FCFS systems. The top three graphs show the total number of jobs in the queue for (a) $N = 1$, (b) $N = 10$, and (c) $N = 100$. The bottom graphs only show the number of long jobs in the system (both in the queue and in the N servers). Whenever there are more than N long jobs, the queue builds up. A long job is one that is three times longer than the average job size.

of a "short" flow is very small because most bytes (work) are carried by long flows as shown in Figure 1.6. As a result, $P(\textit{at least N arrivals during short flow}) = 1 - F_{\hat{\lambda}}(N-1) \rightarrow 0$ as $N \rightarrow \infty$, where $F_{\hat{\lambda}}$ is cumulative distribution function (CDF) of a Poisson distribution with parameter $\hat{\lambda}$, and:

$$\hat{\lambda} = \lambda \times \frac{E[X|short]}{R/N} = \lambda\frac{E[X]}{R} \times \frac{E[X|short]}{E[X]} \times N = \rho \times \frac{E[X|short]}{E[X]} \times N \ll \rho \times N < N$$

where $0 < \rho < 1$ is the total system load, $E[X|short] \ll E[X]$ is the average size of short flows, and R is the link capacity. As a result, $P(\textit{at least N arrivals during short flow}) \approx 0$.

In summary, the response time for circuit switching and packet switching is similar for current network workloads at the core of the Internet, and, thus, circuit switching remains a valid candidate for the backbone. As a reminder, these are only theoretical results and they do not include important factors like the packetization of information, the contention along several hops, the delay in feedback loops, or the flow control algorithms of the transport protocol. The next section explores what happens when these factors are considered.

3.5 Simulation of a real network

To complete this study, I have used ns-2 [89, 125] to simulate a computer network, where I have replaced the packet-switched core with a circuit-switched core using TCP Switching. I will briefly describe how TCP Switching works below for the purposes of calculating the end-user response time. Chapter 4 provides a more detailed description of this network architecture, in case the reader is eager to know more.

With TCP Switching, end hosts operate as they would normally do in a packet-switched Internet. When the first packet of a flow arrives to the edge of a circuit-switched cloud (see Figure 3.9), the boundary router establishes a dedicated circuit for the flow. All subsequent packets in the flow are injected into the same circuit to traverse the circuit-switched cloud. At the egress of the cloud, data is removed from the circuit, reassembled into packets and sent on its way over the packet-switched network. The boundary routers are regular Internet routers, with new linecards that can create and maintain circuits for each flow. The core switches within the cloud are regular circuit switches with their signaling software replaced by the Internet routing protocols. When a core switch sees data arriving on a previously idle circuit, it examines the first packet to determine its next hop, then it creates a circuit for it on the correct outgoing interface. In the simulations, I assume that the local area and access networks are packet switched because, as we have already seen, there is little use in having circuits in the edge links.

In the setup, web clients are connected to the network using 56 Kbit/s links. Servers are connected using 1.5 Mbit/s links, and the core operates at 10 Mbit/s. Later, access links are upgraded to 1.5 Mbit/s, and the rest of the network is scaled proportionally. As one can see, flow rates are heavily capped by the access links of the end hosts, with a core-to-access ratio $N > 180$. Average link loads in the core links are less than 20%, which is consistent with previous observations in the Internet

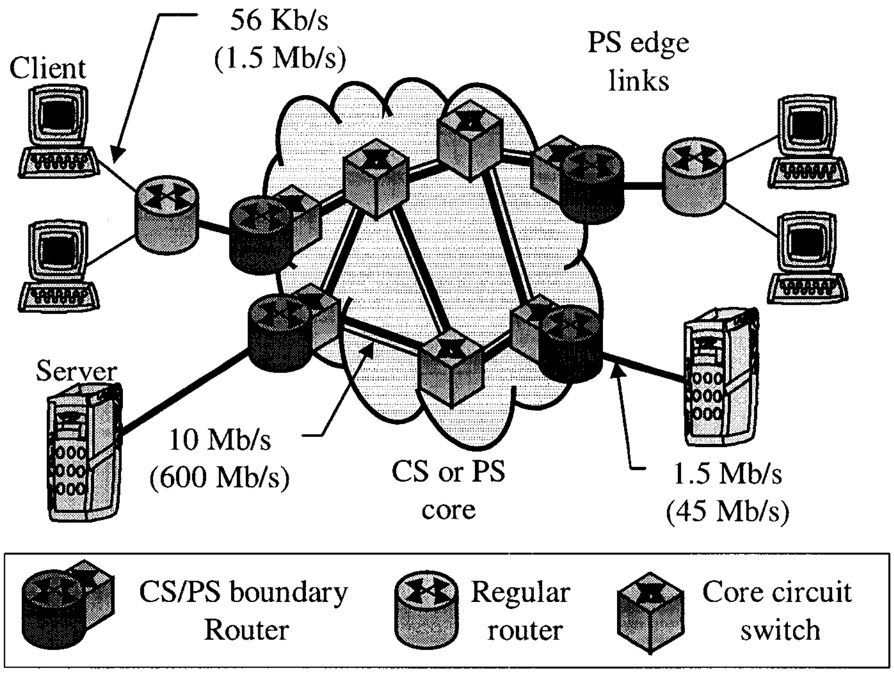

Figure 3.9: Topology used in the ns-2 simulation. Two separate access link speeds of 56 Kbit/s (phone modem) and 1.5 Mbit/s (DSL, cable modem) were considered for the clients. The capacities of server and core links were scaled accordingly.

[135, 47, 90].

We assume that the circuits are established using fast, lightweight signaling, as described in Chapter 4, which does not require confirmation from the egress boundary router, and thus it does not have to wait for a full round-trip time (RTT) to start injecting data into the circuit.

Figures 3.10 and 3.11 show the goodput and the response time, respectively, as a function of the file size. One can see the effects of TCP congestion control algorithms; the shortest flows have a very low goodput. This is mainly due to the slow-start mechanism that begins the connection at a low rate.

The key observation is that packet switching and circuit switching behave very similar, with circuit switching having a slightly worse average response time (14% worse for 56 Kbit/s access links), but the difference becomes smaller, the faster the access link becomes (only 1% worse for 1.5 Mbit/s access links).

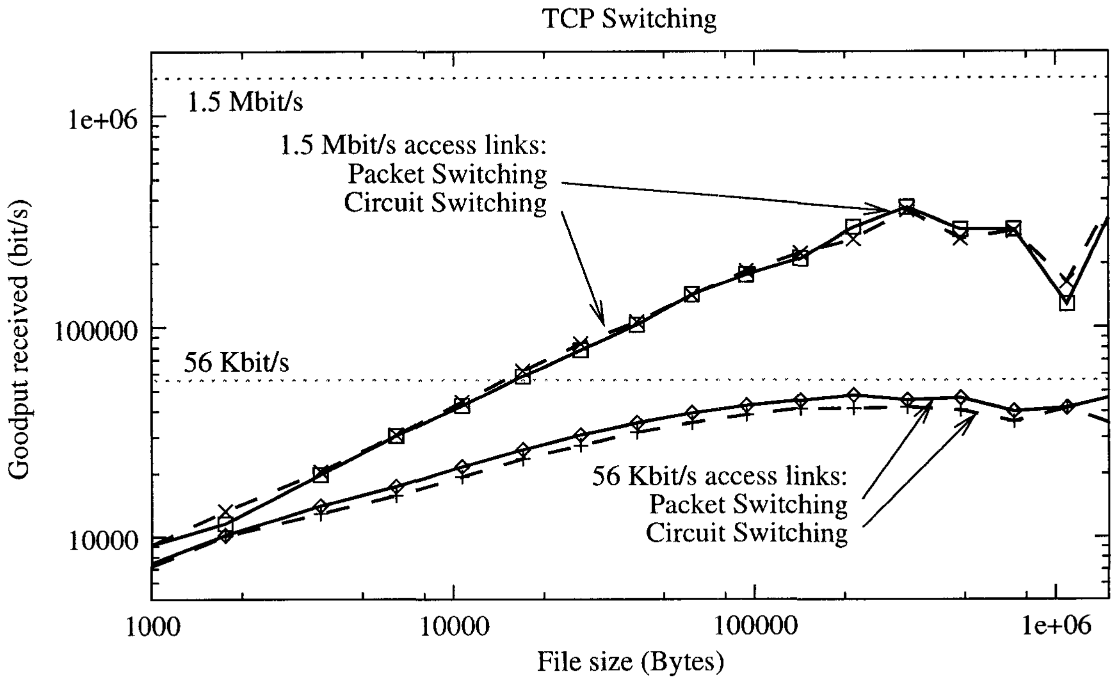

Figure 3.10: Average goodput as a function of the size of the transferred file for access links of 56 Kbit/s and 1.5 Mbit/s.

The reason for circuit switching having worse goodput (and thus response time) is that the transmission time of packets along thin circuits in the core increases the RTT, and this reduces the TCP throughput [139]. For example, to transmit a 1500-byte packet over a 56 Kbit/s circuit takes 214 ms (vs. the 8 ms of a 1.5 Mbit/s link), which is comparable to the RTT on most paths in the Internet. Packet switching does not have this problem because packets are transmitted at the rate of the core link (10 Mbit/s or 600 Mbit/s). In the future, as access links increase in capacity, this increase in the RTT will become less relevant, and circuit switching and packet switching will deliver the same response time. The simulations confirm that, whereas the use of circuit switching in LANs and access networks is undesirable for the end user, users will see little or no difference in terms of response time when using either circuit switching or packet switching in the core.

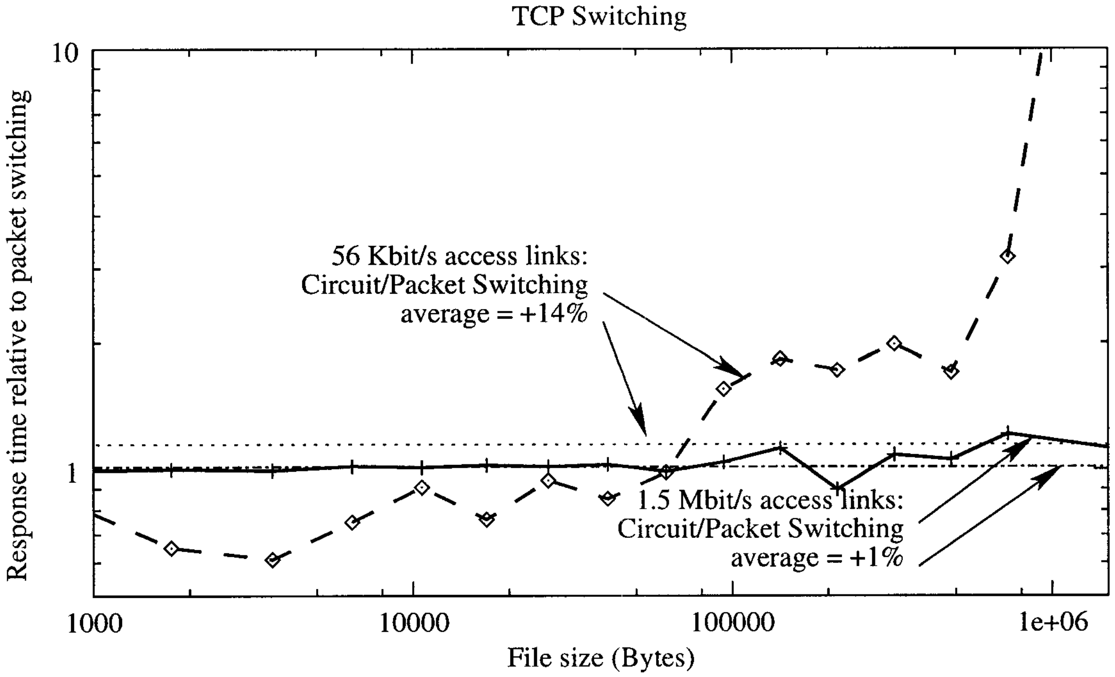

Figure 3.11: Average relative response time as a function of the size of the transferred file for access links of 56 Kbit/s and 1.5 Mbit/s. The average response time of TCP Switching over all file transfers is 14% greater than that of packet switching for 56 Kbit/s access links, and only 1% greater for 1.5 Mbit/s access links.

3.6 Discussion

The results presented in this chapter rely on the fact that the bandwidth ratio between core and access links, N, is greater than 500 for the core of the network today. In the future, this ratio will continue to remain high, as the network topology design will probably not change: with lower-speed tributaries feeding into higher-speed links as we move further into the core. The reason behind this topology design is that the network performance is more predictable. If the ratio N were small, even with packet switching, end users would perceive a big difference between an unloaded network and a moderately loaded one with only a few other users.

In addition, as we have just seen, in the future access links will become faster and so the difference between the response time of circuit switching and packet switching

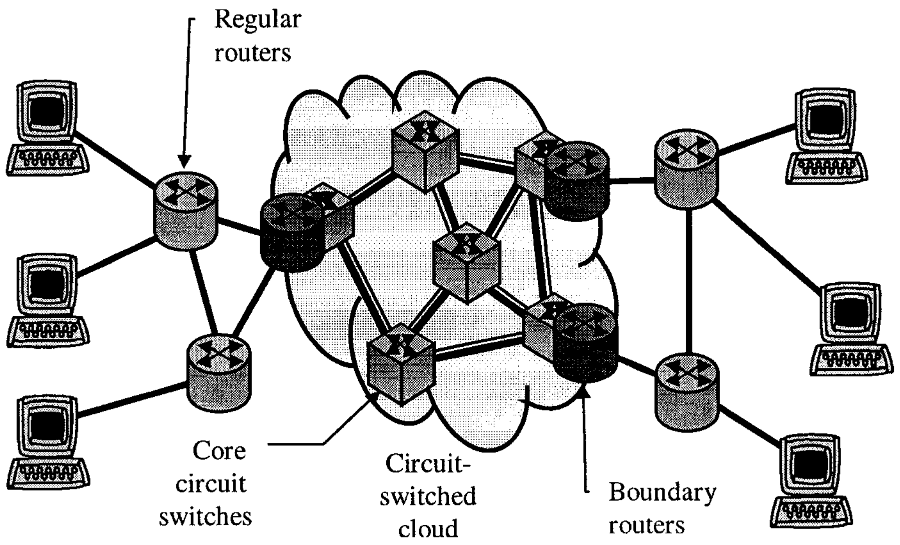

Figure 3.12: Hybrid network architecture using circuit switching in the core and packet switching in the edges that is recommended in this thesis.

will become negligible. So, it is safe to assume that the response time of circuit and packet switching in the core of the network will remain the same in the future.

3.7 Conclusions and summary of contributions

In this chapter, I have argued that, for a given network capacity, users would experience little difference in performance if circuit switching were used instead of packet switching at the core of the Internet. However, this result is not extensible to LAN environments, since big variances of flow sizes in the Internet produce link blocking by circuits, and this in turn makes circuit switching deliver a very poor response time.

The main opportunity for circuit switches comes from their simplicity, and therefore they can be made faster, smaller and to consume less power. As a result, the capacity of the network can be increased without decrementing end-user performance by using more circuit switching in the core. In this thesis, I recommend a network architecture that uses circuit switching in the core and packet switching in the edges, so as to meet Internet's challenging demands for high aggregate bandwidth and low end-user response time at a reasonable cost. This hybrid architecture is shown in

Figure 3.12.

Chapter 4

TCP Switching

4.1 Introduction

In Chapters 1, 2 and 3, I have argued that the Internet would benefit from having more circuit switching in the core of a network that uses packet switching everywhere else. With such an architecture, carriers obtain an infrastructure that is reliable and cost-effective, that can scale to meet the growth demands of Internet traffic, that provides quality of service guarantees, and that does not deteriorate the response time end users currently receive from the network.

Now we need to check that it is not too burdensome to implement a circuit-switched solution for the core. The main concerns arise from the characteristics that set circuit switching apart from packet switching. Namely, won't the amount of state be too large or complex to be handled in real time? Isn't the bandwidth and processing overhead of circuit management too large? What is the effect of the bandwidth inefficiencies and call blocking probabilities of circuit switching? This chapter will look at these and other issues.

As pointed out in Chapter 1, circuit switches are already used in the core of the network in the form of SONET, SDH and DWDM switches. However, IP treats these circuits as static, point-to-point links connecting adjacent nodes; the physical circuits and IP belong to different layers, and they are completely decoupled since they operate autonomously and without cooperation. Decoupling of layers has many advantages. It lets the circuit-switched physical layer evolve independently of IP and vice versa. IP runs over a large variety of physical layers regardless of the underlying technology. At the same time, much of repetition exists between the packet-switched IP layer and the circuit-switched physical layer. For example, a network must route both IP datagrams and circuit paths, yet they use different routing protocols, and their implementations are incompatible. This makes simple and obvious operations infeasible. As a result, provisioning of circuits is done manually, and it can take weeks to allocate a new circuit or to change the capacity of an existing one. Circuit allocation is also rather inflexible because the circuit capacity has to be a multiple of a coarse STS-1 channel (51 Mbit/s). Consequently, circuit provisioning is inefficient and slow to react in real time to changes in traffic patterns.

This chapter and the next present two different ways of integrating circuit switching and packet switching in an evolutionary fashion; that is, these chapters show how end hosts and edge routers are not required to change their protocol stacks or add new signaling mechanisms. This chapter focuses on the mapping of application flows to fine-grain circuits using lightweight signaling, whereas the next chapter maps inter-router flows to coarse circuits with heavyweight signaling.

Below, I present one of the main contributions of this thesis: I propose a technique, called TCP Switching, that exposes circuits to IP; each application flow triggers its own end-to-end circuit creation across a circuit-switched core. TCP Switching takes its name from, and strongly resembles, IP Switching [129], in which a new ATM virtual circuit[1] is established for each application-level flow. Instead, TCP Switching maps these flows to true circuits, thus reaping the advantages of circuit switching.

The proposed architecture is called TCP Switching because most flows today (over 90%) are TCP, and so this architecture is optimized for the common case of TCP connections; but this technique is not limited to TCP, and any uni- or bidirectional flow can be accommodated, albeit less efficiently.

TCP Switching requires no additional signaling, as the first observed packet in a flow triggers the creation of a new circuit. It incorporates modified circuit switches that use existing IP routing protocols to establish circuits. Routing, thus, occurs hop

[1] It is worth noting that virtual circuits are just a connection-oriented packet-switching technique.

by hop, and circuit maintenance uses soft state; that is, it is removed through an inactivity timeout.

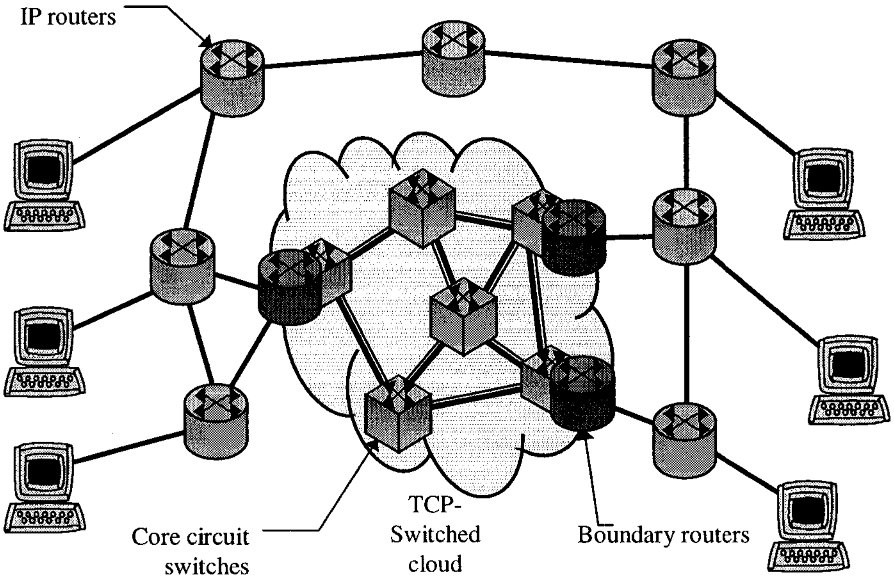

Figure 4.1: An example of a TCP-Switching network.

TCP Switching can be deployed incrementally in the current Internet by creating self-contained TCP-Switching clouds inside a packet-switched network, as Figure 4.1 shows. The packet-switched portion of the network remains unchanged. The core of the circuit-switched portion of the network is built from pure circuit switches (such as SONET cross connects) with simplified signaling to create and destroy circuits. Boundary routers act as gateways between the domains of packets and circuits, and they are most likely conventional routers with circuit-switched line cards.

We are interested in how to make circuit switches and IP routers cooperate. TCP Switching presents a method of interaction, enabling automatic and dynamic circuit allocation. Needless to say, TCP Switching is not the only way of integrating circuit switching and packet switching in the Internet. Indeed, there are several other approaches that I will describe in Chapter 6.

4.1.1 Organization of the chapter

Section 4.2 summarizes the advantages of circuit switching and then describes the potential disadvantages and pitfalls of circuit switching. Section 4.3 describes the network architecture of TCP Switching. Next, Section 4.3.1 analyzes what a typical application flow is in the Internet. Based on these observations about flows and on the discussion in Section 4.2, Section 4.3.3 makes some design choices for TCP Switching. Section 4.3.4 describes the results of the implementation of a TCP Switch. Section 4.4 provides some discussion of the proposed architecture. Finally, Section 4.5 concludes this chapter.

4.2 Advantages and pitfalls of circuit switching

Let us review the main advantages of circuit switching that were described in Chapters 1, 2 and 3:

- Lack of buffers in the data path (Chapter 1).
- Possibility of all-optical data paths (Chapter 1).
- Higher switching capacity (Chapter 2).
- Simple and intuitive QoS (Chapter 2).
- Simple admission control (Chapter 2).
- No degradation of the response time (Chapter 3).

4.2.1 Pitfalls of circuit switching

Despite the advantages listed above, circuit switching has some potential implementation problems that may preclude its utilization if they prove to be too cumbersome. However, I will argue in this chapter that with the proper implementation they are not significant enough to prevent the adoption of circuit switching in the core of the Internet.

4.2.2 State maintenance

Circuit switching requires circuits and their associated state to be established before data can be transferred. A large number of circuits might require a circuit switch to maintain a lot of state. In practice, by observing real packet traces (see Section 4.3.1), I have found that the number of flows, and the rate at which they are added and removed, to be quite manageable in simple hardware using soft state. This holds true even for a high-capacity switch.

4.2.3 Signaling overhead and latency

In order to set up and tear down circuits, switches need to exchange information in the form of signaling. This signaling may represent an important overhead in terms of bandwidth or processing requirements. Depending on how inactive circuits are removed, this state is said to be hard or soft state. If it is hard state, then maintenance is complex because it requires explicit establishments and teardowns, and it has to take into account Byzantine failure modes. In contrast, soft state is simpler to maintain because it relies on end hosts periodically restating the circuits that they use. If a circuit remains idle for a certain period of time, it is timed out and deleted. With the use of hard or soft state, there is a tradeoff between signaling complexity and signaling overhead.

In addition, a considerable latency may be added if additional handshakes are required to establish a new circuit. As I will show with TCP Switching, it is possible to avoid any signaling overhead or latency with circuit switching by piggybacking on the end-to-end signaling that already exists in most user connections.

4.2.4 Wasted capacity

Circuit switching requires circuits to be multiples of a common minimum circuit size. For example, SONET commonly cross connects to provision circuits in multiples of STS-1 (51 Mbit/s). Having flows whose peak bandwidth is not an exact multiple wastes link capacity. Yet using smaller circuit granularity increases the amount of state maintained by the switch. In addition, because bandwidth is reserved, capacity is wasted whenever the source idles and the circuit is active.

In any case, network carriers do not seem to worry much about bandwidth inefficiencies, since networks today are lightly used, and they will likely remain that way since carriers are more interested in operating a reliable network than an efficient one, as shown in Chapter 2. Furthermore, the wasted capacity is not a problem if the speedup of circuit switches with respect to packet switches is bigger than the bandwidth inefficiency.

4.2.5 Blocking under congestion

If no available circuit exists in circuit switching, any new circuit request cannot be processed (gets blocked) until a circuit is free. This data flow system works differently from the link-sharing paradigm present in the Internet today, in which packets will still make (albeit slow) progress over a congested link. However, as we saw in Chapter 3, this blocking does not affect the end-user response time. In the circuit-switched core, some flows may take a longer time to start, but, on average, they finish at the same time as the packet-switched flows.

4.3 TCP Switching

TCP Switching consists of establishing fast, lightweight circuits triggered by application-level flows. Figure 4.1 shows a self-contained TCP Switching cloud inside the packet-switched Internet. The network's packet-switched portion does not change, and circuit switches, such as SONET crossconnects, make up the core of the circuit-switching cloud. These circuit switches have simplified mechanisms to set up and tear down circuits. Boundary routers are conventional routers with circuit-switched line cards, which act as gateways between the packet switching and circuit switching.

The first arriving packet from an application flow triggers the boundary router to create a new circuit. An inactivity timeout removes this circuit later. Hence, TCP Switching maintains circuits using soft state.

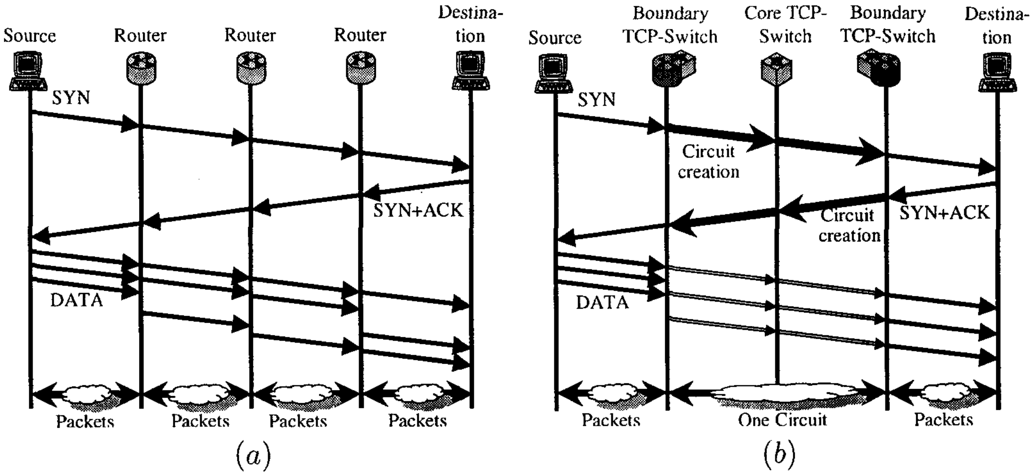

Figure 4.2: Sample time diagram of (a) a regular TCP connection over a packet-switched Internet, and (b) a TCP connection traversing a TCP Switching cloud. The network topology is shown in Figure 4.1.

In the most common case, the application flow is a TCP connection, where a SYN/SYN-ACK handshake precedes any data exchange, as shown in Figure 4.2a. In this case, the first packet arriving at the boundary router is a TCP synchronization (SYN) packet. This automatically establishes an unidirectional circuit as part of the TCP connection setup handshake, and thus no additional end-to-end signaling mechanism is needed, as shown in Figure 4.2b. The circuit in the other direction is established similarly by using the SYN-ACK message. By triggering the circuit establishment when the router detects the first packet — whether or not it is a TCP SYN packet—, TCP Switching is also suitable for non-TCP flows and for on-going TCP flows that experience a route change in the packet-switched network. This is why TCP Switching, despite its name, also works for the less common case of UDP and ICMP user flows.

An examination of each step in TCP Switching, following Figure 4.2b, shows how this type of network architecture establishes a circuit end to end for a new application flow. When the boundary router (shown in Figure 4.3) detects an application flow's first packet, it examines the IP packet header and makes the usual next-hop routing decision to determine the outgoing circuit-switched link. The boundary router then checks for a free circuit on the outgoing link (for example, an empty time slot or an unused wavelength). If one exists, the boundary router begins to use it, and forwards the packet to the first circuit switch in the TCP Switching cloud. If no free circuits exist, the protocol can buffer the packet with the expectation that a circuit will become free soon, it could evict another flow, or it could just drop the packet, forcing the application to retry later. Current implementations of TCP will resend a SYN packet after several seconds, and will keep trying for up to three minutes (depending on the implementation) [19].

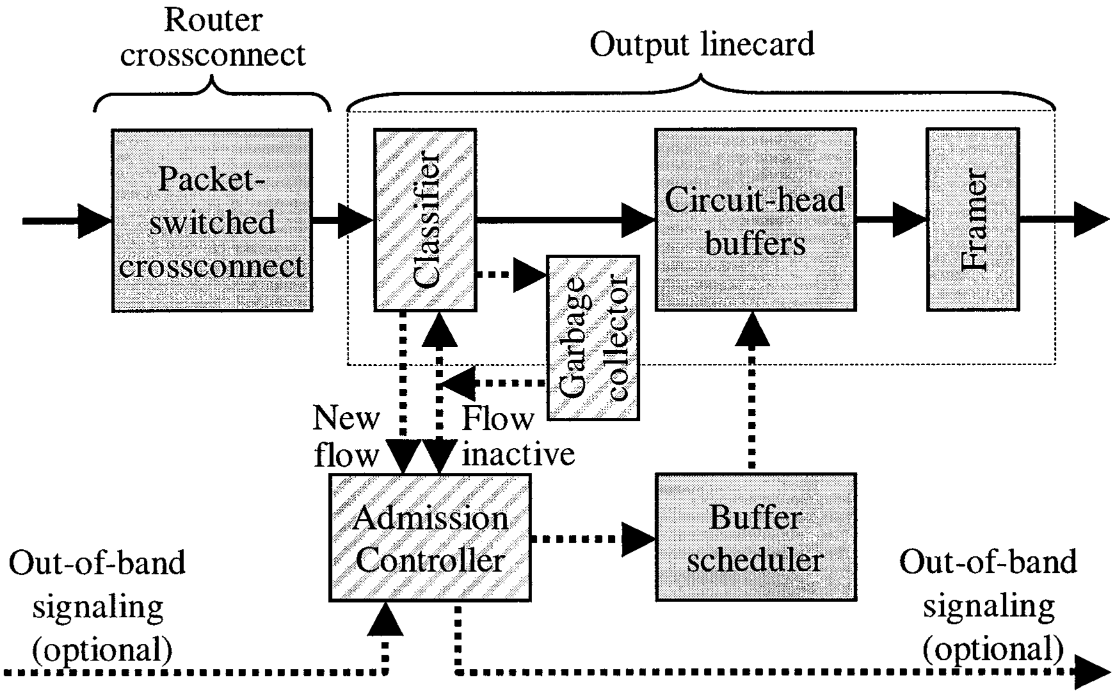

Figure 4.3: Functional block of a TCP-Switching boundary router. The data path is represented by continuous arrows, the control path by the dashed ones. The shaded blocks are not present in a regular router. The classifier and the garbage collector are shown in the output linecard, but they could also be part of the input linecard.

If the circuit is successfully established on the outgoing link, the packet is forwarded to the next-hop circuit switch. The core circuit switch (shown in Figure 4.4) will detect that a previously idle circuit is in use. It then examines the first packet on the circuit to make a next-hop routing decision using its IP routing tables. If a free

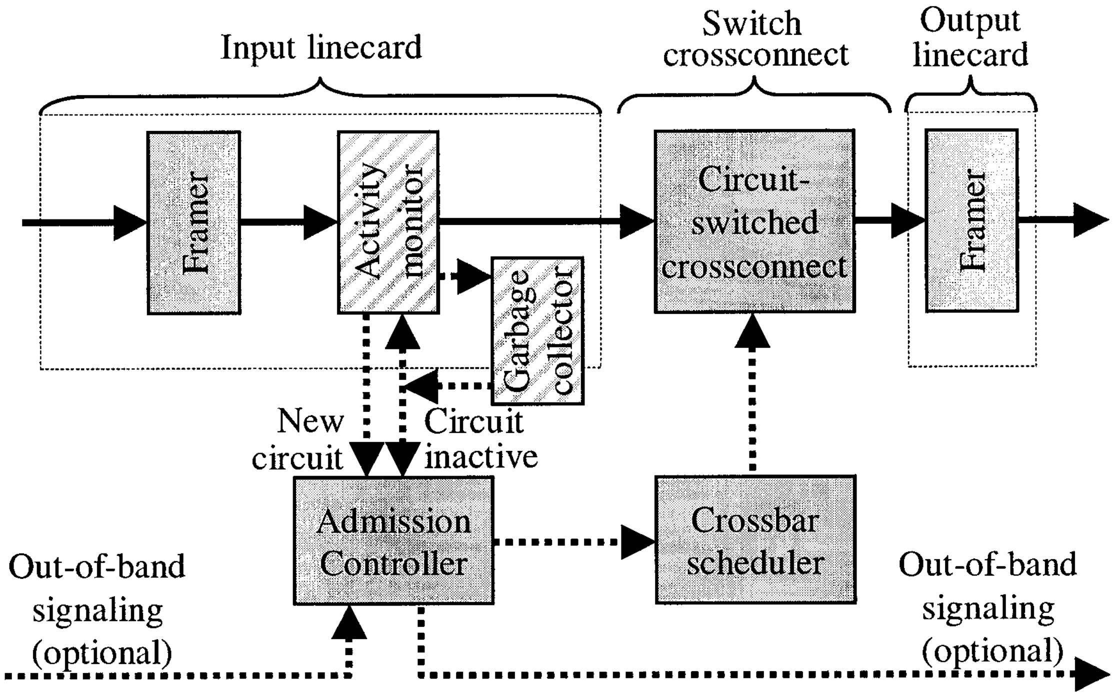

Figure 4.4: Functional block of a TCP-Switching core circuit switch. The data path is represented by continuous arrows, the control path by the dashed ones. The shaded blocks are not present in a regular circuit switch.

outgoing circuit exists, it connects the incoming circuit to the outgoing circuit. From then on, the circuit switch does not need to process any more packets belonging to the flow.

The circuit establishment process continues hop by hop across the TCP-Switching cloud until, hopefully, the circuit is established all the way from the ingress to the egress boundary router. The egress boundary router receives packets from the circuit as they arrive, determines their next hop, and sends them across the packet-switched network toward their destination.

In its simplest form, TCP Switching allows all boundary routers and circuit switches to operate autonomously. They can create circuits, and remove (timeout) circuits, independently. Obvious alternative approaches include buffering the first packet while sending explicit signals across the circuit-switched cloud to create the circuit. However, this removes autonomy and complicates state management, and so it is preferable to avoid this method.

The boundary-router and the circuit-switch complexities are minimal, as shown in Figures 4.3 and 4.4. The ingress boundary router performs most packet processing. It has to map incoming packets from existing flows into the corresponding outgoing circuits, like any flow-aware router would. Additionally, the ingress boundary router processes new flows: it must recognize the first packet in a flow and then determine if the outgoing link has sufficient capacity to carry the new circuit; in other words, it has to do admission control. On the other hand, core circuit switches only need to do processing once per flow, rather than once per packet. These circuit switches only require a simple activity monitor to detect new (active) circuits and expired (idle) circuits. Alternatively, a design could use explicit out-of-band signaling in which the first packet is sent over a separate circuit (or even a separate network) to the signaling software on the circuit switch. In this case, hardware changes to the circuit switch are not necessary because the activity monitor and the garbage collector would not be needed.

Recognizing the first packet in a new flow requires the boundary router to use a four-field, exact-match classifier using the (*source IP address, destination IP address, source port, destination port*) tuple. This fixed-size classifier is very similar to the one used in gigabit Ethernet, and it is much simpler than the variable-size matching that is used in the IP route lookup. When packets arrive for existing circuits, the ingress boundary router must determine which flow and circuit the packet belongs to (using the classifier). The short life of most flows requires fast circuit establishment and tear down.

4.3.1 Typical Internet flows

To provide an understanding of the feasibility and sensibility of TCP Switching, I now study some of the current characteristics of Internet traffic in the backbone. This section focuses on application flows, since TCP Switching establishes a circuit for each application flow. More precisely, I start discussing what a TCP flow is, and how it behaves, because over 90% of Internet traffic is TCP — both in terms of packets, bytes and flows. I have studied `traceroute` measurements, as well as packet traces from OC-3c and OC-12c links in the vBNS backbone network[2] [131]. These results are similar to the ones obtained from flow traces from OC-48c links in the Sprint backbone [170].

Traffic characteristics	80-percentile	Average	Median
TCP flow duration (seconds)	$\leq$ 4 – 10	$\leq$ 3 – 7	$\leq$ 0.5 – 1.2
Packets per flow	$\leq$ 12	$\leq$ 10 – 200*	$\leq$ 5 – 9
Flow size (Kbytes)	$\leq$ 2.5 – 4	$\leq$ 9 – 90*	$\leq$ 0.6 – 1.3
Flow average bandwidth [size/duration] (Kbit/s)	$\leq$ 50 – 100	$\leq$ 20 – 140*	$\leq$ 8 – 15
Fraction of flows with retransmissions	$\leq$ 7.8%	$\leq$ 5.6%	$\leq$ 4.7%
Fraction of flows experiencing reroutes	$\leq$ 0.19%	$\leq$ 0.39%*	$\leq$ 0.02%
Asymmetrical connections	Around 40% of the flows transmit an ACK after the FIN; i.e., they keep acknowledging data packets that are sent in the other direction.		

Table 4.1: Typical TCP flows in the Internet. The figures indicate the range for the 80-percentile, the average and the median for the different links taken in August, 27-31, 2001 [131]. The magnitudes marked with * present a non-negligible amount of samples with very high values; that is, their statistical distribution has long and heavy tails. This is why the average is higher than the 80-percentile.

Table 4.1 describes the typical TCP flow in the Internet. TCP connections usually last less than 10 seconds, carry less than 4 Kbytes of data and consist of fewer than 12 packets in each direction. Less than 0.4% of connections experience a route change. The typical user requests a sequence of files for downloading and wants the fastest possible download for each file. In most cases, the requested data is not used until the file has completely arrived at the user's machine.

Figure 4.5 shows the cumulative histogram of the average flow bandwidth —

[2] This data was made available through the National Science Foundation Cooperative Agreement No. ANI-9807479, and the National Laboratory of Applied Network Research.

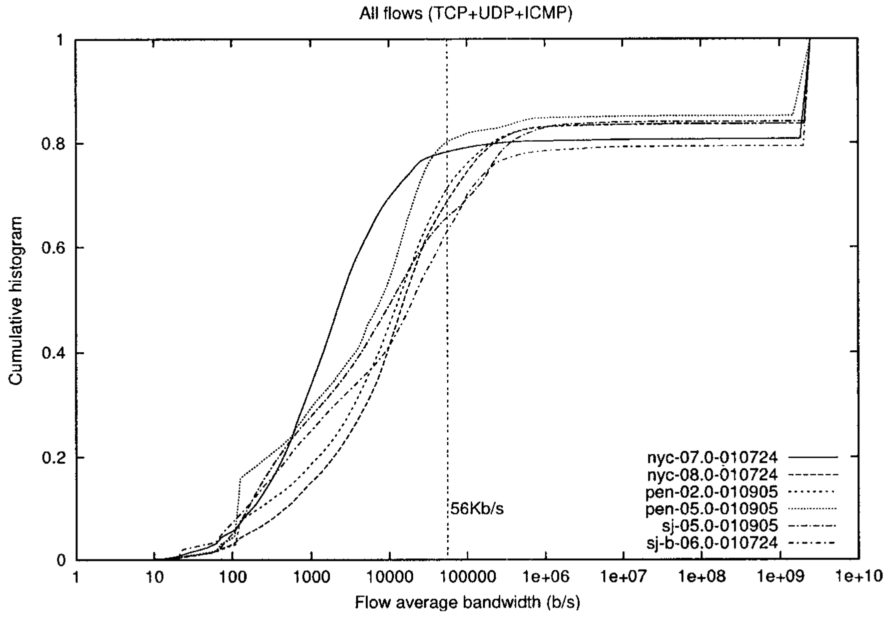

Figure 4.5: Cumulative histogram of the average flow bandwidth for TCP and non-TCP flows. The traces were taken in July and September 2001 from OC-48c links in the Sprint backbone network [170]. The flows with the peak bandwidth (2.5 Gbit/s) are single-packet flows (usually UDP and ICMP flows, and a few broken TCP connections).

defined as the ratio between the flow size[3] and the flow duration — for both TCP and non-TCP flows from several OC-48c traces from the core of the Internet. As one can see, with the exception of single-packet flows, very few flows achieve an average bandwidth that is greater than 1 Mbit/s. Furthermore, most of the multi-packet flows (78%-97% of them) receive less than 56 Kbit/s from the network either because one of the access links is a 56-Kbit/s modem or because the application does not take advantage of all the available bandwidth. This confirms that, as discussed in Chapter 3, the bandwidth ratio between core and access links, N, is much greater than 500.

Figure 4.6 shows the correlation between the flow duration and the flow size. Most flows are both short in size (80-4000 bytes) and medium in duration (0.1-12 s). This

[3] The flow size is the number of bytes transported by the flow, including the header overhead, control messages and retransmissions.

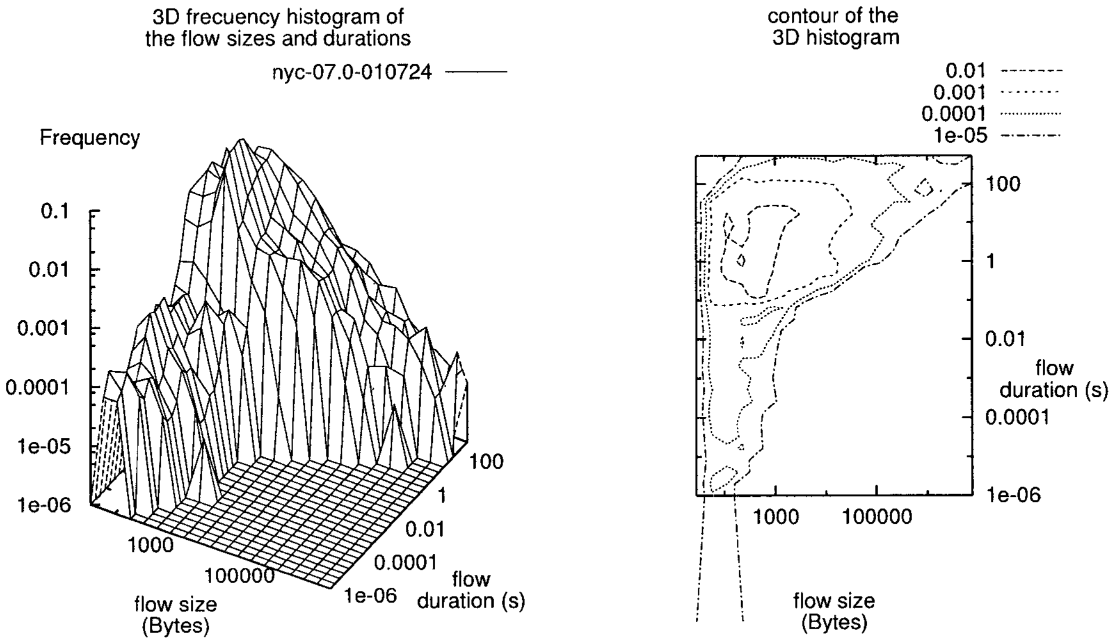

Figure 4.6: 3D Frequency histogram of flow sizes and durations for both TCP and non-TCP flows in one trace from OC-48c links in the Sprint backbone network [170].

is because the source cannot fill up the core link on its own; the slow-start phase of the TCP connections requires several round trips before the source can transmit at the available rate for the flow, and, in addition, the access link forces spacing between consecutive packets belonging to the same flow.

4.3.2 Design options

TCP Switching is in fact a family of network architectures in which there are numerous design options. These options indicate tradeoffs between implementation simplicity, traffic control and efficiency. Below, I list several of these design options:

Circuit establishment

Option 1	Triggered by first packet seen in a flow (can be any packet type).
Option 2	Triggered by TCP SYN packets only.
Notes	If there is a path reroute outside the TCP switched cloud, the switch will not detect the SYN packet. This is rare in practice.

Circuit release

Option 1	Triggered by inactivity timeout (soft state).
Option 2	Triggered by a finish (TCP FIN) signal (hard state).
Notes	Neither option is perfect. The switch might sever connections that either have asymmetrical closings (hard state) or long idle periods (soft state).

Handling of non-TCP flows

Option 1	Treats user datagram protocol (UDP) and TCP flows the same way.
Option 2	Multiplex UDP traffic into permanent circuits between boundary routers.
Notes	UDP represents a small (but important) amount of traffic.

Signaling

Option 1	None. Circuit establishment is implicit based on observed packets.
Option 2	Explicit in-band or out-of-band signaling to establish and remove circuits.
Notes	In-band signaling requires no additional exchanges, but it is more complex to implement.

Circuit routing

Option 1	Hop-by-hop routing.
Option 2	Centralized or source routing.
Notes	A centralized algorithm can provide global optimization and path diversity, but it is slower and more complex.

Circuit granularities

Option 1	Flat. All switches have the same granularity.
Option 2	Hierarchical. Fine circuits are bundled in coarser circuits as we move towards the inner core.
Notes	A coarser granularity means that the switch can go faster because it has to process less.

4.3.3 Design choices

Using the observations in Section 4.3.1, I now describe some design choices that I have used in experiments of TCP Switching.

Circuit signaling

In my design, I use implicit signaling, that is, the arrival of a packet on a previously inactive circuit triggers the switch to route the packet and create a new circuit. Circuits are removed after they have been idle for a certain period of time. This eliminates any explicit signaling at the small cost of adding a simple activity monitor to the data path.

Bandwidth assignment

I assume in the experiments that the core circuit switches carry 56-Kbps circuits to match the access links of most network users. High capacity flows use multiple circuits. There are two ways of assigning a peak bandwidth to a flow: the preferred one is to make the decision locally at the ingress boundary router. The alternative is to let the source use an explicit signaling mechanism like RSVP [18] or some TCP-header option, but this requires a change in the way applications currently use the network. With the local bandwidth assignment, users would be allocated 56-Kbit/s by default unless their address appears in a local database listing users with higher data-rate access links and/or who have paid for a premium service.

Flow detection

The exact-match classifier detects new flows at the ingress boundary router. The classifier compares the headers of arriving packets against a table of active flows to check if the flow belongs to an existing circuit, or whether a new circuit needs to be created. The size of the classifier depends on the number of circuits on the outgoing link. For example, an OC-192c link carrying 56-Kbps circuits requires 178,000 entries in its table, an amount of state that fits on an on-chip SRAM memory. Given the duration of measured flows, in a one-second period one expects about 31 million lookups, 36,000 new connections, and 36,000 old connections to be timed out for an OC-192c link. This is quite manageable in dedicated hardware [4, 71].

I use soft state and an inactivity timer to remove connections. For TCP flows, an alternative could be to remove circuits when the router detects a FIN signal, but in about 40% of TCP flows, acknowledgement (ACK) packets arrive after the FIN because the communication in the other direction is still active.

Inactivity timeouts

In my design, the timeout duration is a tradeoff between efficiency in bandwidth and signaling. For example, my simulations suggest that a 60-second timeout value will reliably detect flows that have ended (which is similar to results by IP Switching [129] and Feldman et al. [74]). This timeout value ensures that flows are neither blocked nor severed during the connection lifetime.[4] But, the cost of using such a long timeout value is high because the circuit remains unused for long time, especially if the flow duration is of only a few seconds.

To reduce the bandwidth inefficiencies, one could use a very short timeout value so that there is some statistical multiplexing among active flows. However, if the timeout is very short, the control plane of circuit switching would often have to be visited more than two times during the lifetime of a flow. In the extreme case of a timeout of zero, circuit switching degenerates into packet switching, where every

[4] If the circuit were timed out during the lifetime of the flow, it can be reestablished rapidly. However, there is a risk that a new request gets rejected because the old resources have been claimed by another flow.

piece of information has to be routed, processed and buffered, which severely limits the switch performance. If one wants to avoid this degenerate behavior, the timeout should be greater than the maximum transmission time of a packet through the circuit (214 ms for 56-Kbit/s circuits, 8 ms for 1.5-Mbit/s circuits).

To choose the right timeout value, one has to take into account the timing of the TCP mechanisms to avoid severing the circuit during a naturally occurring pause. The first observation would be that the minimum retransmission timeout value of TCP is 1 s [145]. However, retransmission timeouts are rare, they represent less than 0.5% of all transmissions, and so it should not be very expensive to have inactivity timeouts of less than 1 s.

A more important factor is the slow-start mechanism used by all TCP connections to ramp-up the flow rate. This mechanism creates some silence periods that occur during the initial round trips. Having an inactivity timeout that is smaller than the round trip time (RTT) is very expensive, especially for the frequent short TCP flows (the so-called *mice* [23]). It is then recommended that inactivity timeout values greater than the RTT be used (on earth, most RTTs for minimum-size packets are smaller than 250 ms).

Circuit replacement policies

In my experiments, when a circuit remained inactive for a certain period of time it was torn down. Circuits that time out need not be evicted immediately; they may just be marked as candidates to be replaced by a new circuit when a request arrives. This reduces the per-circuit processing for circuits that are incorrectly marked as inactive. If the new circuit request uses the same path, it is then possible to reuse the existing circuit without any new signaling. One could use different replacement policies, as with the cache of a computer system. The simplest policy is the Least Recently Used (LRU), but others are possible. In case of contention, preemptive policies could be used to evict lower-priority circuits to accommodate higher-priority ones.

Switching unit

Several applications, such as web browsing, open several parallel TCP connections between the same two end hosts. These parallel flows share the same access link, and thus it would be wasteful to allocate the bandwidth of the access link to each of them. Instead, all these parallel flows should be sharing a single circuit. So, rather than using TCP flows as the switching unit, it is better to use IP flows (i.e., flows between pairs of end hosts).

4.3.4 Experimentation with TCP-Switching networks and nodes

I experimented with TCP-Switching networks via simulation using ns-2 [89]. The main results are presented in Section 3.5, and they show that TCP Switching does not yield a worse response time than packet switching for the core of the network, despite the bandwidth inefficiencies and call blocking that are typical of circuit switching.

These simulations assume that TCP Switching nodes can process the requests for new circuits as quickly as needed. This hypothesis was validated through the implementation of a TCP Switching boundary router.[5] The boundary router was implemented as a kernel module in Linux 2.4 running on a 1-GHz Pentium III. Neither this platform nor the implementation were particularly optimized to perform this task, and yet in TCP Switching forwarding a packet in the boundary router took $17-25$ μs (as opposed to 17 μs for regular IP forwarding and the $77-115$ μs of IP's QoS forwarding that comes standard with Linux). In this non-optimized software, the circuit setup time is approximately 57 μs, fast enough to handle new connection requests of an OC-48c link (an OC-192c link has an average flow interarrival time of $16-39$ μs at full capacity, an OC-48c of $64-156$ μs). These numbers should drop dramatically if part of the software were implemented in dedicated hardware.

[5] This prototype was built by Byung-Gon Chun, an M.S. student at Stanford, for a 10-week class project under my supervision [39].

4.4 Discussion

TCP Switching exploits the fact that most of our communications are connection oriented and reliable. Rather than using complex signaling mechanisms to create and destroy circuits, TCP Switching piggybacks on existing end-to-end mechanisms to manage circuits. More specifically, TCP Switching uses the initial handshake of the most common type of flows, TCP connections, to create a circuit in the core of the network. When a circuit request message gets dropped, TCP Switching relies on the TCP retransmission mechanisms to set up the circuit again at some later time.

In addition, TCP Switching tries to exploit some of the statistical multiplexing that exists among flows. Obviously, it does not achieve the statistical multiplexing of packet switching because, if a flow does not fully utilize its circuit capacity, this unused bandwidth is wasted. However, TCP Switching does not reserve resources for inactive flows, and so those resources can be employed by other active flows. In this way, TCP Switching achieves some statistical multiplexing gain.

TCP Switching is indeed an extreme technology. In what follows, I will discuss some of the concerns that arise when this approach is described; namely, I will discuss the impact of single-packet flows, bandwidth inefficiencies and denial of service.

4.4.1 Single-packet flows

The longer flows are, the more efficient TCP Switching because the circuit setup cost is amortized over the longer data transfer. It is unclear how flow sizes will evolve in the future. On one hand, there is a trend for longer flows from downloads and streaming of songs and video. On the other hand, traffic from sensors is likely to consist of very short exchanges, perhaps consisting of single-packet flows. Even though most probably those single-packet flows will be aggregated before being sent through the Internet backbone [2], it is worth asking what would happen if single-packet flows became a large fraction of Internet traffic.

As mentioned in Section 4.3.3, packet switching can be considered to be a special case of circuit switching in which all flows are 1-packet long. The processing and forwarding of a circuit request in the control plane of a circuit switch is similar to the forwarding of a packet in the data plane of a router. When a circuit arrives, a next-hop lookup has to be performed and resources have to be checked. If these resources are available, the crossconnect needs to be scheduled; otherwise, the request has to be buffered or dropped. The only difference with packet switching is that state is maintained so that next time data arrives for that circuit, the data path can forward the information without consulting the control plane.

In TCP Switching, single-packet flows are forwarded as if using packet switching by the control plane, while long flows are forwarded by the data path of circuit switching at a much higher rate. In order to avoid interactions in the control plane between these two classes of flows, one can create two separate queues and process them differently (e.g., the single-packet flows would not write any state).

4.4.2 Bandwidth inefficiencies

In TCP Switching, the wastage of bandwidth is evident because it suffers from acute fragmentation; the bandwidth allocated to a circuit is reserved for its associated flow, and if the flow does not use it, the bandwidth is wasted. Nevertheless, it is not clear to what extent the bandwidth inefficiency is a problem because, as shown in Figure 1.3, optical-link capacity does not have the technology limitations of buffering and electronic processing. One should then ask the question, how much speedup is needed in a circuit switch to compensate for the wasted bandwidth?

In order to quantify how much bandwidth remains unused, we need to look at the time diagram of a typical circuit, as shown in Figure 4.7. As we can see, during the lifetime of a TCP-Switching circuit, there are three phases when bandwidth is wasted by a TCP flow: (1) during the slow start phase, when the source has not yet found the available bandwidth in the circuit, (2) during the congestion avoidance phase, and (3) during the inactivity period that is used to timeout and destroy the circuit. The total amount of bandwidth that is wasted in each phase will depend on the source activity, the flow length, the round-trip time, and the inactivity timeout. For example, the so-called TCP "mice" or "dragonflies" [23] are so short that they do not enter the congestion avoidance phase.

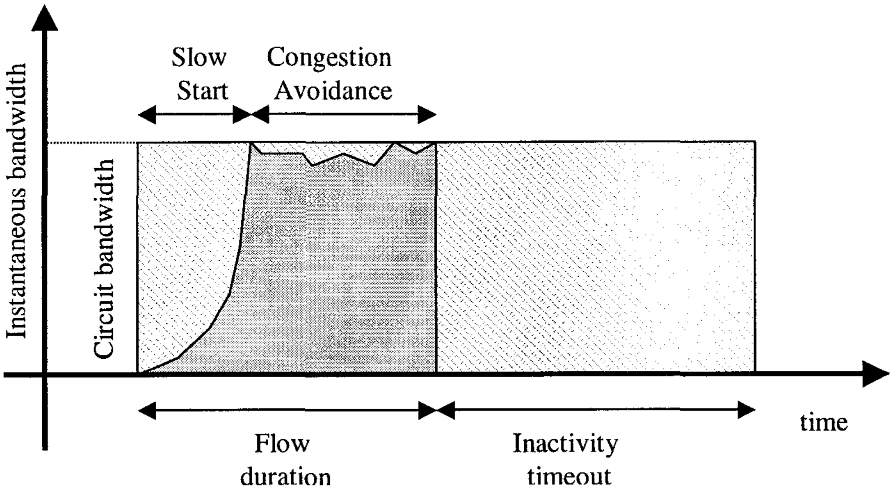

Figure 4.7: Bandwidth inefficiencies in TCP Switching. The dashed circuit bandwidth is wasted.

Given that application flow sizes are typically shorter than 10 s, an inactivity timeout of 60 s such as the one proposed in [129, 74] is extremely wasteful. Better efficiencies can be achieved with a timeout value that is a little larger than the RTT (as proposed in Section 4.3.3) because this timeout value is comparable to the duration of the slow-start phase, which lasts a few RTTs.

In any case, traffic is highly asymmetric. Usually, one end host holds the information, and the other simply downloads it, such as with web browsing. This means that one direction of the connection will be filling up the pipe with large packets (typically of 1500 bytes), whereas the other direction will be sending 40-Byte long acknowledgements. If the two circuits belonging to a bi-directional flow are symmetric, then even if we achieved a bandwidth efficiency of close to 100% in the direction of the download, the reverse direction will get an efficiency of less than 2.7%. The overall efficiency would be only 51%. However, the direction of download is not uniformly distributed, as servers tend to be placed in PoPs and co-location facilities, and so the direction in which the bottleneck occurs will get a bandwidth efficiency closer to 100% than to 2.7%.[6] If the bandwidth inefficiency of the reverse circuit proved to be critical, one

[6] Since links in the core are usually symmetrical, the bandwidth inefficiencies caused by this traffic unbalance also affect packet switching.

could allocate less bandwidth to the return channel for the acknowledgements.

4.4.3 Denial of service

Denial of service is an important concern for TCP Switching. With only a few well-crafted packets one can reserve a huge amount of bandwidth, preventing others from using it. This problem is not new, and it is common to other systems that do resource reservation. Two solutions are possible here: one is to use external economic incentives and penalties to deter a user from taking more resources than he/she needs. The other is to restrict the maximum number of simultaneous flows that an ingress boundary router may accept from a single user.

On the other hand, one of the advantages of TCP Switching is that circuits are reserved exclusively for one flow, so, contrary to packet-switched networks, it is easy to track a circuit back to its source, and it is virtually impossible for others to spoof a circuit or to hijack it without the cooperation of a switch. This inherent authentication makes the enforcement of policies across domains easier than in the current Internet.

4.5 Conclusions and summary of contributions

This chapter has focused on how the existing IP infrastructure can incorporate fast, simple (and perhaps optical) circuit switches. Several approaches to this already exist, but I have proposed a technique called TCP Switching in which each application flow (be an individual TCP connection or other types of flows) triggers its own end-to-end circuit creation across a circuit-switched core. Based on IP Switching, TCP Switching incorporates modified circuit switches that use existing IP routing protocols to establish circuits. Routing occurs hop by hop, and circuit maintenance uses soft state; i.e., it is removed through an inactivity timeout. TCP Switching exploits the fact that our usage of the Internet is very connection-oriented and has end-to-end reliability to provide lightweight signaling mechanisms for circuit management. Finally, this chapter has showed how despite the fine granularity of its circuits, TCP

Switching is implementable with simple hardware support, and so it is capable of providing the advantages of circuit switching to the Internet. TCP Switching is an extreme approach that shows how one can integrate circuit switching in the core of the existing Internet while extracting the benefits of circuit switching that were listed in Chapters 1, 2 and 3: higher switching capacity, robustness, simple QoS and end-user response time similar to that of packet switching.

Chapter 5

Coarse circuit switching in the core

5.1 Introduction

In Chapters 1, 2 and 3, we have seen how we can benefit from having more circuit switching in the core of the Internet; circuit switching allows us to build switches with fast all-optical data paths. Moreover, circuit switching can provide higher capacity and reliability than packet switching without degrading end-user response time. Chapter 4 described TCP Switching, an evolutionary way of integrating a circuit-switched backbone with the rest of the Internet. This integration of circuit and packet switching was done by mapping application flows to fine-grain, lightweight circuits.

One problem of TCP Switching is that currently most crossconnects in circuit switches cannot switch at the 56-Kbit/s granularity that was proposed in Chapter 4.[1] It would be extremely wasteful to reserve an STS-1 channel or a wavelength exclusively for a single user flow whose peak rate is limited to only 56 Kbit/s by its access link. Another shortcoming, caused by several crossconnect technologies and signaling mechanisms in circuit switching, is that it may take tens or hundreds of milliseconds to reconfigure the switch or to exchange the signaling messages that create a new circuit. These long circuit-creation latencies occur when the crossconnect has to

[1] Typically, electronic SONET circuit switches use circuit granularities of STS-1 (51 Mbit/s) or higher, whereas DWDM switches have granularities of OC-48 (2.5 Gbit/s) or higher.

move electromechanical parts, such as MEMS mirrors, or when the signaling requires a positive acknowledgement from the egress circuit switch.

This chapter addresses these two limitations of circuit-switching technologies. More precisely, it explores ways of using circuit switching in the Internet when cross-connects have granularities that are much larger than the peak rate of user flows, and when circuit switches are slow to reconfigure.

In this chapter, I propose monitoring the bandwidth that is carried by all user flows between each pair of boundary routers around a circuit-switched cloud in the core. This measurement provides an anticipating and stable estimation of the traffic matrix that is then used to properly size the coarse-granularity circuits that interconnects the boundary routers. In order to compensate for the circuit-creation latency of the network, I propose allocating these circuits using an additional safeguard band that prevents the circuits from overflowing.

One of the themes developed in Chapter 4 is again the basis for this chapter; namely, how one can configure the circuit-switched backbone by monitoring the activity of user flows. The difference is that now a core circuit is carrying many user flows simultaneously, and so the mapping between user flows and circuits is not as straightforward as that discussed in Chapter 4. It is no longer a matter of when to create the circuits but how much capacity to assign to them.

5.1.1 Organization of the chapter

Section 5.2 defines the problem addressed in this chapter and describes other approaches that have been proposed by other researchers. Section 5.3 shows how one can control a circuit-switched Internet core by monitoring user flows. Section 5.4 builds a model for the buildup of user flows. Then, Section 5.5 discusses some of the implications of this approach. Finally, Section 5.6 concludes this chapter.

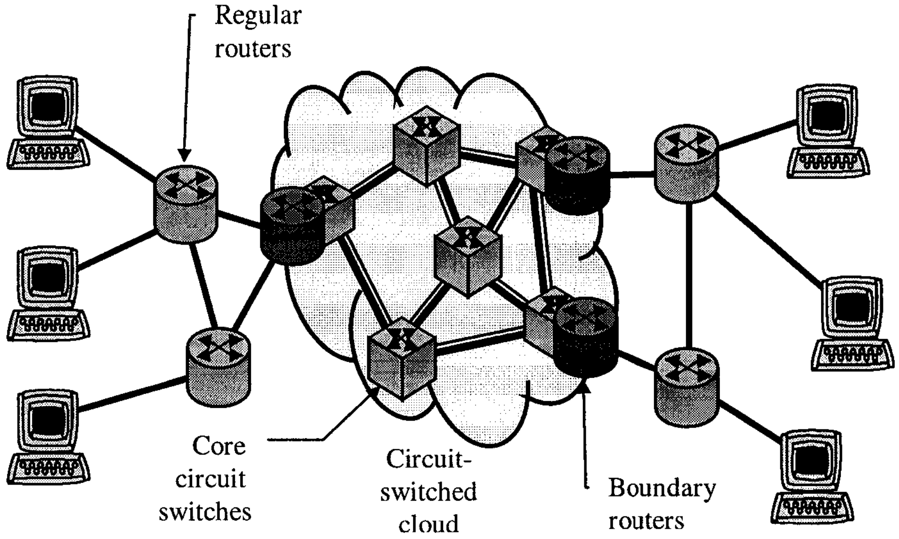

Figure 5.1: Network topology considered in this chapter.

5.2 Background and previous work

Consider the network architecture shown in Figure 5.1. The problem that arises now is how to accommodate traffic between boundary routers around the circuit-switched cloud. This issue can be decomposed in two parts: First, how much capacity is needed between boundary routers? (i.e., what is the traffic matrix between boundary routers?). Second, once we know this traffic matrix, how do we create the circuits that provide the required capacity?

The second question has been looked at by several researchers before. In some cases, it is regarded as a centralized optimization problem, in which the total throughput of the network needs to be maximized subject to constraints on link capacity and maximum flow rate. The problem is either solved using integer linear programming or some heuristic that achieves an approximate, but faster, solution [9, 163, 176]. The output of this centralized optimization problem is then distributed to the circuit switches. In other cases, researchers have treated the problem as an incremental problem in which each circuit request is routed individually as it arrives [169, 8, 13]. The circuit routing can be done at the source or hop-by-hop. Chapter 6 describes some of these signaling and routing protocols that have been proposed.

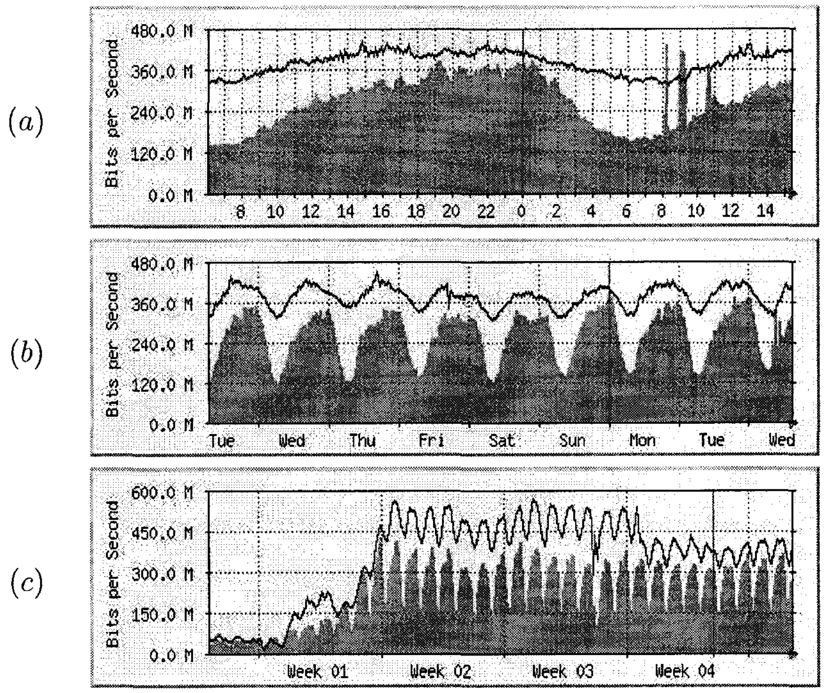

Figure 5.2: Daily (a), weekly (b) and monthly (c) average traffic traces in a 1-Gbit/s Ethernet link between Purdue University and the Indiana GigaPoP connecting to Internet-2 [63], as observed on February 5th, 2003. The dark line indicates outgoing traffic; the shaded area indicates incoming traffic. The low traffic in weeks 0 and 1 in graph (c) corresponds to the last week of December 2002 and the first week of January 2003, respectively, when the university was in recess.

This chapter focuses on the first question, how to estimate the traffic matrix between boundary routers and then use the estimate to provision coarse-granularity circuits. Some researchers [9, 163, 120] have suggested that future traffic matrices can be predicted off-line using past observations. These researchers point out that traffic in the core of the network is smoother than at the edges and that it follows strong hourly and daily patterns that are easy to characterize, as shown in Figure 5.2. For example, traffic is affected by human activity and scheduled tasks, and so peak traffic occurs during work hours on weekdays, whereas at night and during weekends there is less traffic.

Nonetheless, this off-line prediction of the traffic matrix fails to forecast sudden changes in traffic due to external events, such as breaking news that creates flash crowds, a fiber cut that diverts traffic or the new version of a popular program that generates many downloads. Only an on-line estimation of the traffic matrix would be able to accommodate these sudden and unpredictable changes in traffic patterns.

This on-line estimation of traffic could be done in several ways. One of them is to monitor the aggregate packet traffic [79], either by observing the instantaneous link bandwidth or the queue sizes in the routers. While this approach does not require much hardware support and is easy to implement, it does not provide good information about the traffic trends. Packet arrivals present many short- and long-range dependencies that make both the instantaneous arrival rates and queue sizes fluctuate wildly. In contrast, I propose using another way of estimating the current traffic usage by monitoring user flows. It requires more hardware support, but, in exchange, user flows provide a traffic estimation that is more predictive and has less variation, at least for the circuit-creation latencies under consideration (1 ms-1 s), as we will see below.

Figure 5.3 gives a clear example of the fluctuations in the instantaneous arrival rate. The dots in the background denote the instantaneous link bandwidth over 1-ms time intervals. With so much noise, it is difficult to see any trends in the data rates. Thus, one could apply filters to smooth the signal. For example, the dark gray line in Figure 5.3 shows the moving average $R(t) = (1 - \alpha)R(t - \Delta t) + \alpha r(t)$, where $r(t)$ is the instantaneous measure, Δt is 1 ms and α is 0.10. The figure also shows the instantaneous traffic rate over 100-ms intervals (light gray line) and the sum of the average bandwidth of the active flows (black line). The average bandwidth of a flow is the total number of bits that are transmitted divided by the flow duration. Of course, the flow average bandwidth is something that is not known when a flow starts, but I will explain below how to estimate an upper bound in the next section. One can see that the 100-ms bin size provides the signal with the least noise of all measures of traffic based on counting packets, but there are still many more fluctuations than in the measure based on the average bandwidth of active flows, from which the trends are much clearer. In brief, user flows provide more stable measurement than packets and queue sizes for time scales between 1 ms and 1 s.

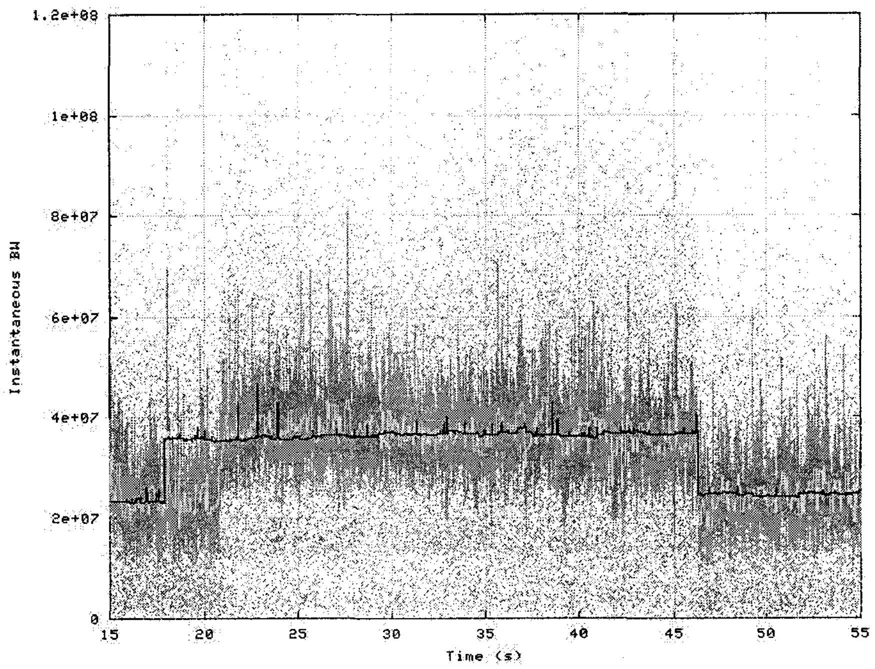

Figure 5.3: Time diagram showing the instantaneous link bandwidth over 1-ms intervals (dots), its time moving average (dark gray line), the instantaneous link bandwidth over 100-ms intervals (light gray line) and the sum of the average rate of all active flows (black line). The trace was taken from an OC-12 link on January 18th, 2003 [131].

User flows also provide an advance notice before major changes in bandwidth utilization occur. For example, we can see that in Figure 5.3 there is a sudden increase in traffic between times 21 s and 46 s (due to a single user flow whose only constraint was a 10-Mbit/s link). This traffic increase was predicted four seconds before it happened by observing the active flows. There are two reasons for this advance notice: first, an application may take some time to start sending data at full rate after it connected with its peer.[2] Second, it takes several round-trip times for TCP connections to ramp up their rate to the available throughput because of the slow-start mechanism. In contrast, when we monitor the instantaneous arrival

[2] This is the main reason for the advance notice in Figure 5.3. Unfortunately, it was impossible to know what application caused this behavior because the trace was "sanitized".

rate to estimate the traffic matrix, the filters that are applied to reduce the noise add some delay to the decision making of whether more capacity is needed. If the circuit creation mechanism already takes a long time, adding more delay in the traffic estimation makes the system react more slowly.

5.3 Monitoring user flows

In this chapter, I propose monitoring user flows to estimate the traffic matrix between boundary routers because they can provide a stable (i.e., with little variation) and predictive estimate of the actual traffic. The arrival process of these user flows has in general fewer long- and short-range dependencies. It has been reported that arrivals of user flows in the core behave as if they followed a Poisson process [78]. Feldmann reported that the interarrival times of HTTP flows follow a Weibull distribution [73], but as the link rate increases and more flows are multiplexed together, the interarrival times tend to the special case of the exponential distribution according to Cao et al. [33] and also Cleveland et al. [45]. The Poisson arrival process is well understood and there are many models in queueing theory that use it.

The approach I am proposing requires similar hardware support for monitoring active flows to what was described in Section 4.3, basically a fixed-length classifier that can detect new flows and that monitors the activity of the current flows. Such classifiers are already available for OC-192c link speeds [4, 71]. However, counting the number of active flows is not enough, as we need to know the average bandwidth that they use. Most of the time it is not possible to estimate the average bandwidth when the flow starts because it requires knowing the flow duration and the number of bits that will be transmitted. In contrast, it is possible to have an upper bound, which I will call peak bandwidth[3] of a flow. I consider this flow peak bandwidth to be constant throughout the lifetime of the flow, and it can be determined the same way as in Section 4.3.3 (through signaling or through an estimation of the access link bandwidth) even if one does not know *a priori* the flow average bandwidth.

[3] The only requirement for the flow peak bandwidth is that it has to be larger than the flow average bandwidth.

More formally, once we know the set of active flows between a pair of boundary routers at a given instant, $F(t)$, we need only to assign a peak bandwidth to each of the flows, C_f, where $C_f \geq average\,BW(f), \forall f \in F(t)$. Then, we need a circuit with a capacity, $K_{cct}(t)$, that is greater than or equal to the sum of the flow peak bandwidths, $C(t) = \sum_{f \in F(t)} C_f$ and $K_{cct}(t) \geq C(t)$.

Many circuits take time to be created because circuit management signaling, switch scheduling, and/or crossconnect reconfiguration may be slow. If we decide that we need to increase the circuit capacity between two boundary routers, $C(t)$, there will be some latency, T, before the changes take place. During this latency period, the circuit capacity will be insufficient, and queueing delays and potentially packet drops will occur at the circuit head. I call this a *circuit overflow.* More precisely, a circuit overflow happens at time t when:

$$\{\exists \tau \in [t, t+T), s.t.\ C(\tau) > C(t)\}$$

Obviously, the longer the circuit-creation latency, T, is, the more circuit overflows occur.

A way of avoiding circuit overflows is to provision extra capacity as safeguard. The size of this safeguard band depends on T and the dynamics of traffic, and it determines the probability of a circuit overflow.

I have used several user-flow traces from the Sprint Backbone [170] to analyze the sizes of the safeguard bands based on the signaling delays and the overflow probabilities. Figure 5.4 displays a sample path showing how the safeguard band varies with time for a circuit-creation latency of $T = 1\,\mathrm{s}$. Based on the sum of average bandwidths of the active flows (dashed line) and the sum of the corresponding peak bandwidths,[4] one can construct the instantaneous peak-bandwidth envelope, $C(t)$, depicted as a solid line. The dotted line shows the safeguard-band envelope, i.e., the maximum of the peak-bandwidth envelope in the next T period ($= 1$ s), $\hat{C}_T(t) = max\{C(\tau); \tau \in [t, t+T)\}$.

[4] In the analysis, the peak flow rate is defined as the minimum number of 56-Kbit/s circuits that are needed to carry the average flow rate. Over 97.5% of the flows fitted within a single 56-Kbit/s circuit. A tighter bound could have been used, as well.

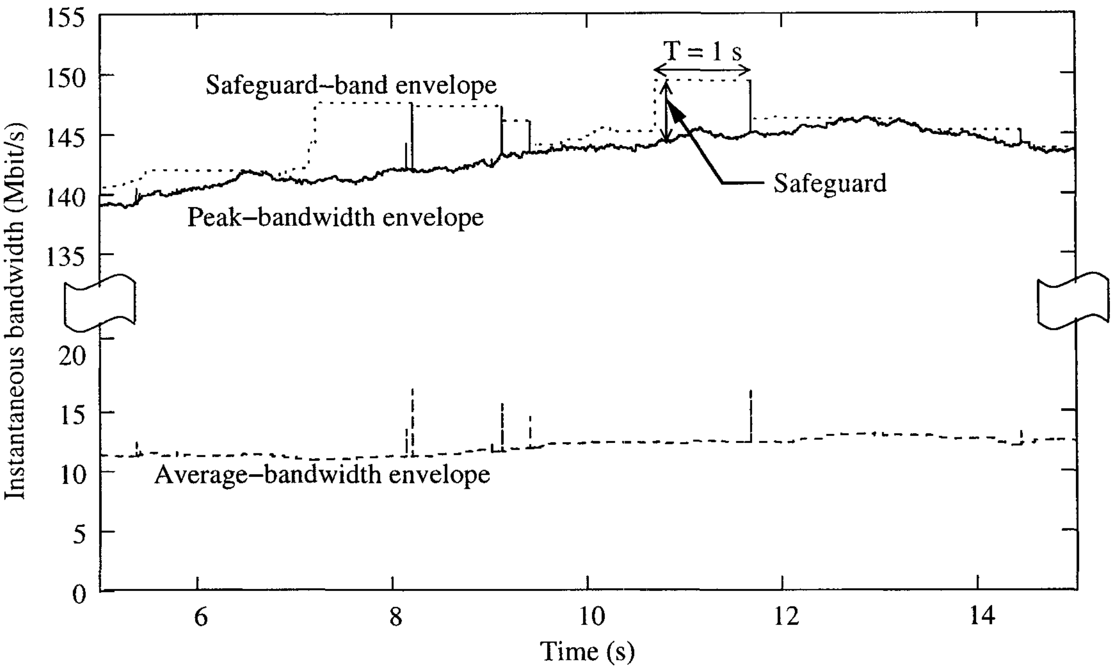

Figure 5.4: Time diagram showing how the safeguard-band envelope is calculated.

Since we cannot predict the future, we can instead analyze $\hat{C}_T(t)$ to find the safeguard band, S_T^p, for a given overflow probability p. In other words, we calculate S_T^p such that $P(\hat{C}_T(t) - C(t) \leq S_T^p \cdot C(t)) \leq p$. In a real system, we would continuously estimate the instantaneous peak-bandwidth envelope, $C(t)$. If at any time the difference between the circuit capacity and $C(t)$ goes below the safeguard band, $K_{cct}(t) - C(t) < S_T^p \cdot C(t)$, then we would request an increase in the circuit capacity, so that the spare capacity remains above the safeguard band to avoid circuit overflows.

Figure 5.5 depicts the safeguard band relative to the peak-bandwidth envelope, $(\hat{C}_T(t) - C(t))/C(t)$, for various overflow probabilities and circuit-creation latencies for one OC-12 link in the Sprint backbone. There are some stair-case steps for a relative safeguard band between 0.04% and 0.3% because the peak-bandwidth envelope can only increase in multiples of 56 Kbit/s, which is around 0.04% of the peak-bandwidth envelope.

Figure 5.5 confirms our intuition that the longer the circuit-creation latency, the larger the safeguard band needs to be for a given overflow probability. For example,

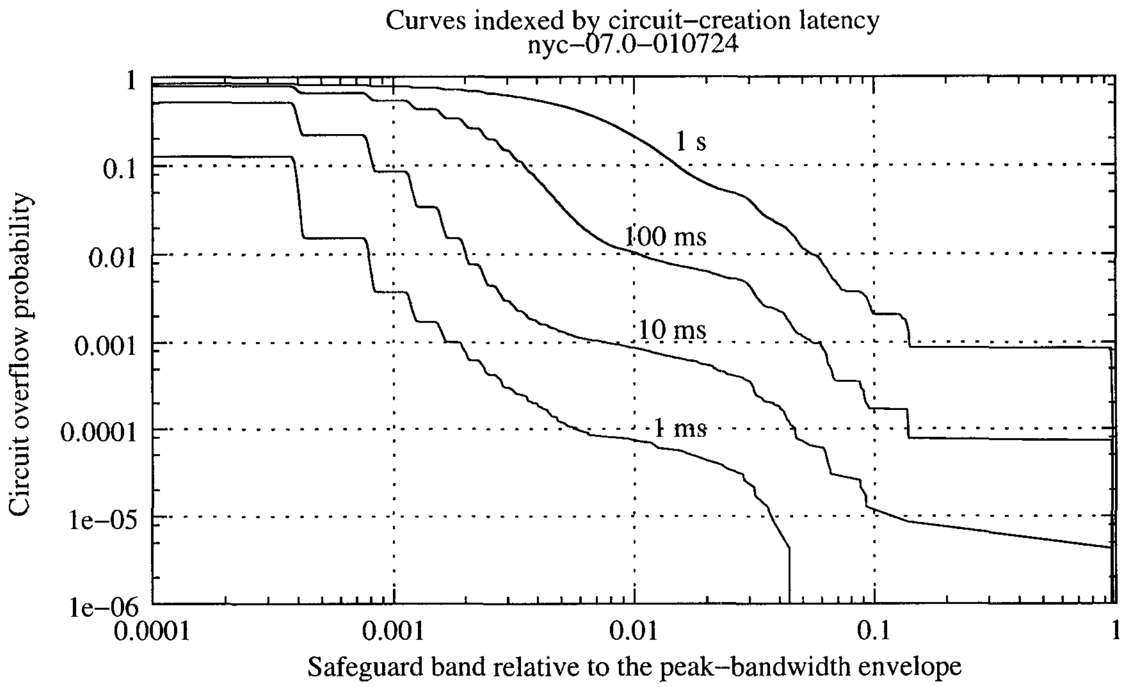

Figure 5.5: Safeguard band required for certain overflow probabilities and circuit-creation latencies.

for an overflow probability of 0.1%, one needs a safeguard of 0.15% times the current peak-bandwidth envelope for T=1 ms, 0.8% for T= 10 ms, 6% for T= 100 ms and 12% for T= 1 s. The faster the crossconnect and the signaling are, the more efficiently resources are used. This result indicates that we should use fast signaling and fast switching elements for the establishment of circuits.

It should also be noted that for safeguard bands greater than or equal to 0.1% of the peak-bandwidth envelope and latencies smaller than or equal to 100 ms, a ten-fold decrease of the circuit-creation latency corresponds to a ten-fold decrease in the overflow probability for the same safeguard band. These results were consistent among all traces that were studied. Figure 5.6 shows the cumulative histogram of the peak-bandwidth envelope for the different Sprint traces. The trace used for Figure 5.5 is the one centered around 150 Mbit/s (nyc-07.0-010724). The other traces yielded similar safety margins.

The liberation of resources is not as important as their reservation because their

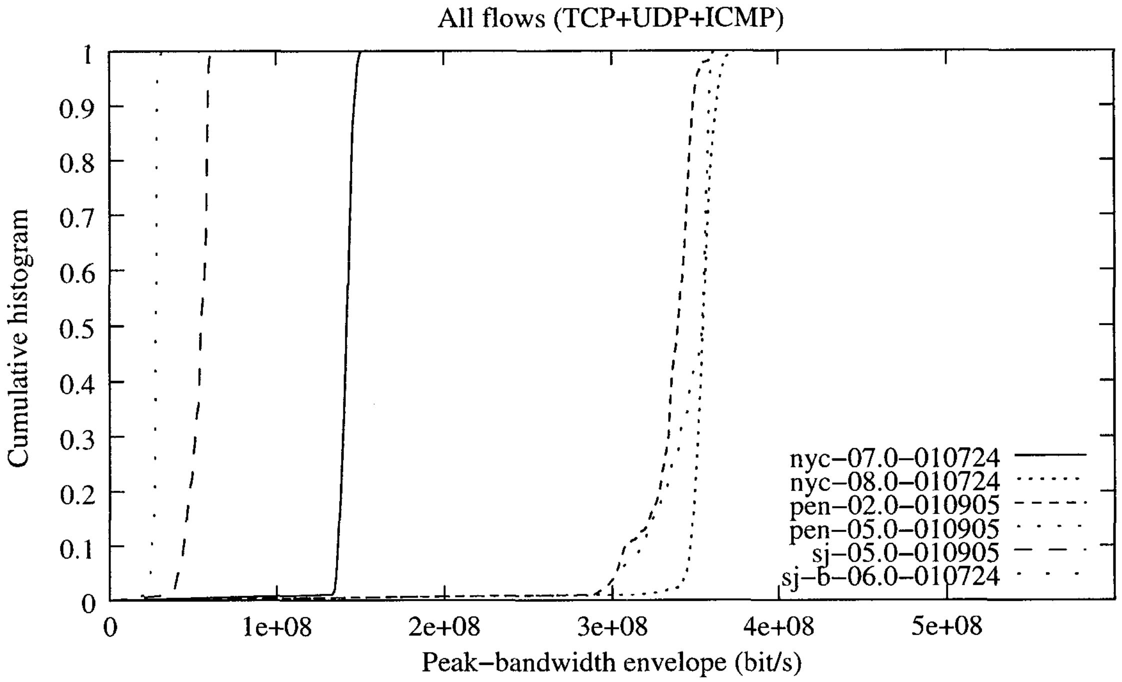

Figure 5.6: Cumulative histogram of the peak-bandwidth envelope for different Sprint traces.

release does not directly contribute to a circuit overflow (unless bandwidth is scarce, but as mentioned in Chapter 2, bandwidth is plentiful in the core). One can simplify the circuit management signaling with a scheme that uses soft state and an inactivity timeout. This simple scheme would retain the extra circuit bandwidth for a period of time that is at least as long as the circuit-creation latency to avoid oscillations in the resource allocation.

5.4 Modeling traffic to help identify the safeguard band

In the previous section, I used traces from real links in the network to predict the safeguard band that is required for a certain overflow probability. In most cases, it is not economical to have trace collecting equipment on every link, and so it may not be possible to obtain such detailed traces. For this reason, it is beneficial to have a simple model that requires less information to achieve the same goal. In addition, if the model is simple enough, one can also obtain formulae that predict the appropriate safeguard band based on a small number of network parameters.

I will now perform the same analysis as in the previous section on synthetic traffic traces that are generated using the distributions and statistics from the links under consideration. Notice that this stochastic information can be obtained with considerably less effort than a real trace because they can be estimated by sampling the traffic.

In a trace of active flows, one has three pieces of information per flow: the flow interarrival time, the flow duration and the flow average bandwidth.[5] Flow interarrival times are essentially independent of each other and closely follow a Poisson process, as shown by the nearly exponential interarrival times in Figure 5.7. In the traces, the average arrival rates were between 124 and 594 flow/s. This hypothesis of Poisson-like arrivals is further supported by the wavelet estimator described by Abry and Veitch [1]: the Hurst parameter of the interarrival times is very close to 1/2, which suggests independence. Similar results have been reported by Fredj et al. [78] and by Cleveland et al. [33, 45]. For this reason, for the synthetic trace, we can model flow arrivals as a Poisson process. Hence to parameterize the model we need only the average arrival rate of the flows.

The flow average bandwidth (shown in Figure 5.8a) and the flow duration (shown in Figure 5.8b) have empirical distributions that are harder to model. Furthermore, the values are not independent of each other. The correlation coefficient between them in the Sprint traces was between -0.134 and -0.299,[6] which is consistent with the work by Zhang et al. [189]. Figure 5.9 shows the joint histogram for the flow duration and average bandwidth, which makes their correlation clear. Jobs with more available bandwidth usually take less time to complete. In terms of successive arrivals, the autocorrelation function was almost zero, and so the arrivals can be considered

[5] One could use the number of bytes transferred by the flow instead of the flow average bandwidth, since the latter is equal to the former divided by the flow duration.

[6] For TCP traffic, the correlation coefficient was between -0.137 and -0.310, whereas for non-TCP traffic, it was between -0.089 and -0.391.

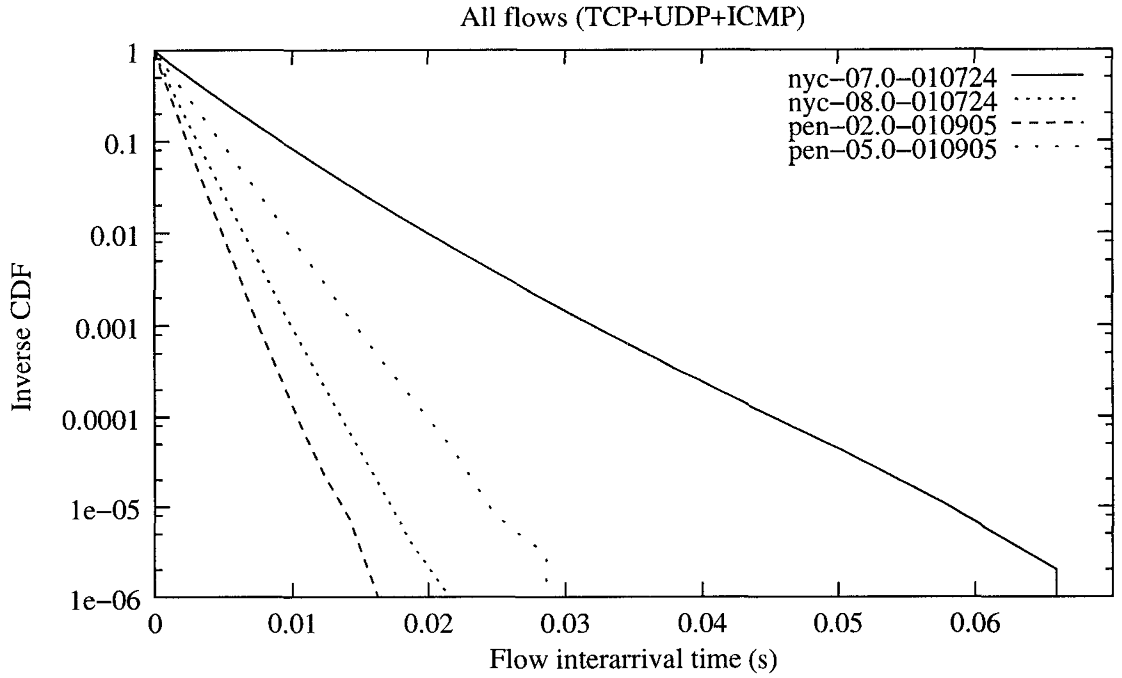

Figure 5.7: Inverse cumulative histogram of the flow interarrivals for both TCP and non-TCP traffic in the Sprint traces. An exponential interarrival time would be represented as a straight line in this graph.

independent. The flow average bandwidth and flow duration can then be modeled as a sequence of i.i.d. 2-dimensional random variables.

Even if the correlation between the flow average bandwidth and the flow duration is small, when the marginal distributions of the two magnitudes are used the results of the model and the traces diverge considerably for the low overflow probabilities. The reason is that short-duration and high-bandwidth flows occur more often in the synthetic traces created from the marginal distributions than in the real trace, and these flows can skew the results. Results are much closer to the trace-driven model when using Poisson arrivals and the empirical joint distribution for the flow duration and average rate. Figure 5.10 shows how the synthetic trace using the joint distribution produces results that are very close to those obtained with the real trace.

This model corresponds to an $MB/G/\infty$ system, where there are infinite parallel servers, arrivals are batched Poisson and service times are correlated with the

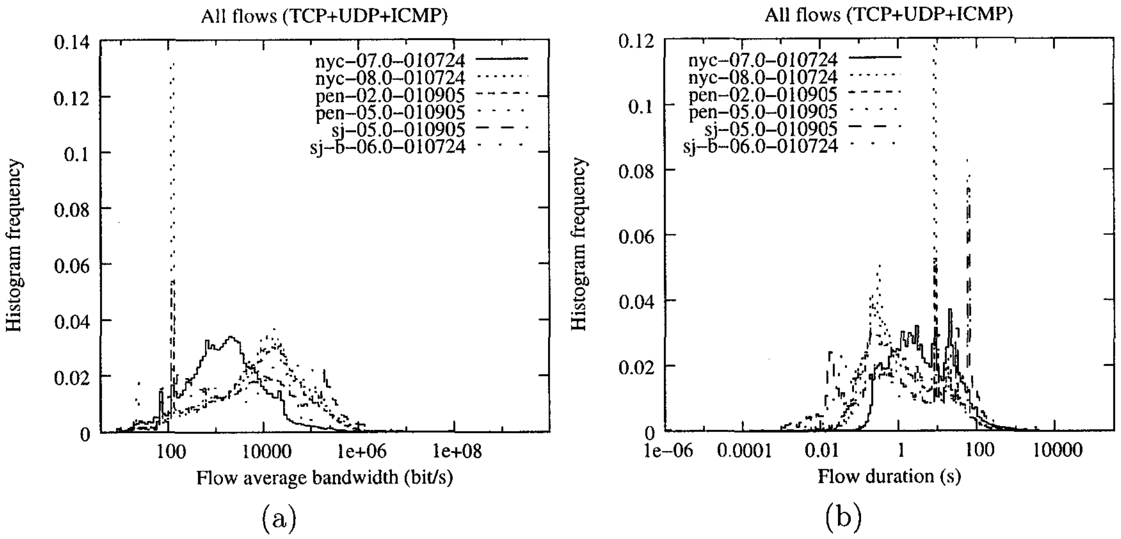

Figure 5.8: Histograms of (a) the flow average bandwidth and (b) the flow duration for both TCP and non-TCP traffic in the Sprint traces. Single-packet flows have not been considered.

batch size. As far as I know, there is no closed-form expression for the transition probabilities:

$$\begin{aligned} p &= 1 - P[N(t) - N(0) < S_T^p \cdot N(0), \forall t \in [0,T)] \\ &= P[max\{N(t) - N(0), \forall t \in [0,T)\} \geq S_T^p \cdot N(0)] \end{aligned}$$

where $N(t)$ is the number of clients in the system at time t.

In summary, we can estimate the safeguard band that is required to avoid circuit overflows just by using the average flow rate and the joint distribution of the flow average bandwidth and the flow duration. This information can then be used to construct a set of curves like the one in Figure 5.10.

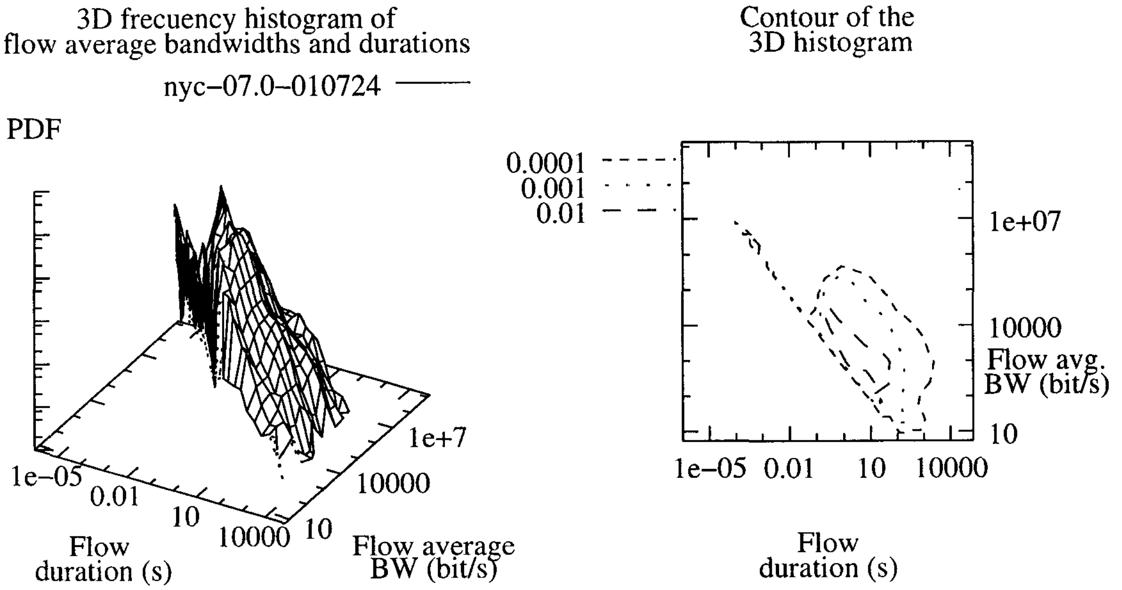

Figure 5.9: Joint histogram of flow durations and average bandwidths for both TCP and non-TCP traffic in the Sprint traces.

5.5 Discussion

This chapter considers circuits between boundary routers. If we need to increase the capacity of an existing circuit, it might be that the current circuit path cannot accommodate this increase, while an alternate path can. In this case, one option is to reroute the whole circuit through a path that has the required capacity (if there is one), but this option might be too costly in terms of signaling and resource consumption. One alternative is to create a separate circuit with a capacity equal to the additional capacity that is needed. However, one problem is that this parallel circuit will have a propagation latency that is different from the original path. If data is injected into the combined circuit, it may happen that a packet is split into two parts that travel through different paths, and so a complex realignment buffer will be required at the egress point to realign the two parts of the packet. Such a mechanism has already been proposed for SONET/SDH, and it is known as virtual concatenation of channels [46, 166].

One way of eliminating this realignment is to avoid splitting packets over parallel

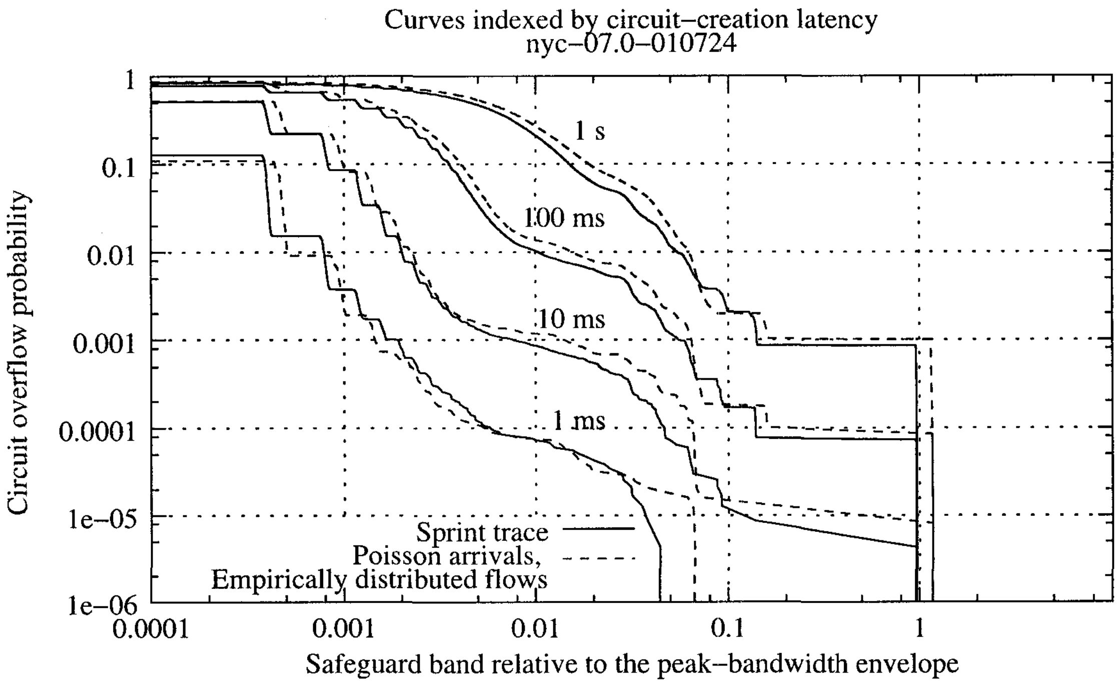

Figure 5.10: Safeguard band required for certain overflow probabilities and circuit-creation latencies for real traffic traces (solid line) and a simple traffic model (dashed line) with Poisson arrivals and flow characteristics that are drawn from an empirical distribution.

paths. Packets can then be recovered integrally at the tail end of each backbone circuit and injected directly into the packet-switched part of the network. This method can create some packet reordering within a user flow, which TCP may interpret as packet drops due to congestion. Yet, reordering would be rare if the difference in propagation delays between the parallel paths is smaller than the interarrival time imposed by the access link to consecutive packets of the same flow (for 1500-byte packets, it is 214 ms for 56-Kbit/s access links, and 8 ms for 1.5-Mbit/s access links). One possible solution is to equalize the delay using a fixed-size buffer at the end of one of the sub-circuits. However, this buffer may not be necessary because, as reported recently [17], TCP is not significantly affected by occasional packet reordering in a network.

It should be pointed out that the definition used here for circuit overflow is rather strict, and it represents an upper bound on the packet drop rate. In general, the ingress boundary router will have buffers at the head end of each backbone circuit, which will absorb short fluctuations in the flow rate between boundary routers. For this reason, in the measurements in Figures 5.5 and 5.10, all single-packet flows were ignored. The buffer at the head end will also allow the system to achieve some statistical multiplexing between active flows; something that TCP Switching in Chapter 4 could not achieve. However, as mentioned in Chapter 3, this statistical multiplexing will not necessarily lead to a smaller response time because the flow peak rate will still be capped by the access link.

The approach presented in this chapter does not specify any signaling mechanism and does not impose any requirements on it. One could use existing mechanisms such as the ones envisioned by GMPLS [7] or OIF [13], which will be described in Chapter 6. This method can also be used in conjunction with TCP Switching to control an optical backbone with an electronic outer core and an optical inner core. TCP Switching would control the outer fine-grain electronic circuit switches and would provide the information that is used to control the inner coarse-grain optical circuit switches.

5.6 Conclusions and summary of contributions

This Chapter has discussed how to monitor user flows to predict when more bandwidth is needed between boundary routers of a circuit-switched cloud in the Internet. It has also developed a simple model that can be used to estimate safeguard bands that compensate for slow circuit-creation mechanisms.

The most important recommendation of this chapter is that the signaling used to establish a circuit should be as fast as possible. Otherwise, the safeguard band becomes very large. An alternative reading of this recommendation is that the establishment of a circuit should be done simultaneously on all nodes along the path, without having to wait for any confirmation from the upstream or downstream nodes. Moreover, slow crossconnect technologies (such as MEMS mirrors) should only be used if they can provide a very large switching capacity at a low cost, such that it can accommodate the additional safeguard band.

Chapter 6

Related work

6.1 Introduction

In this chapter, I will briefly summarize work that is related to the topic of this thesis.

6.1.1 Organization of the chapter

One key aspect of this thesis is the exploration of how to integrate high-capacity, all-optical circuit switches in the core of the network with a packet-switched access network. This integration was achieved by monitoring user-flows. Section 6.2 summarizes other approaches that also integrate circuit switching in the Internet. In contrast, Section 6.3 presents some approaches that try to extend the packet switching paradigm to an all-optical core. Since this approach differs significantly from the rest of this thesis, Section 6.3 has a discussion of its performance. Finally, Section 6.4 reviews other proposals for monitoring user flows.

6.2 Circuit switching in the Internet

As mentioned in Chapter 1, it is becoming increasingly difficult to build high-performance packet-switched routers. This is due to several reasons, but the primary reason is because traffic is growing faster than electronic technology in general, and memory access speeds in particular. This calls for research into alternatives to packet switching. One of these alternatives, which has also been explored by other researchers, is to integrate very high-capacity optical circuit switches in the core of an otherwise packet-switched Internet. Four main dynamic signaling mechanisms have been proposed to manage circuits in SONET/SDH and DWDM networks: Generalized Multiprotocol Label Switching — GMPLS — (Section 6.2.1), Automatic Switched Transport Network — ASTN — (Section 6.2.2), Optical Internetworking Forum — OIF — (Section 6.2.3), and Optical Domain Service Interconnect — ODSI — (Section 6.2.4). For each of these four approaches, a working group has defined signaling mechanisms for managing circuits, but leave it to vendors to define how to monitor traffic, when to trigger a new circuit establishment, and how much bandwidth to allocate.

Two architectures have been proposed to help decide when to create a circuit and how much bandwidth to give to it. The first is Optical Burst Switching — OBS — (Section 6.3), in which a router at the edge of the network queues packets up to a threshold and then establishes a circuit with an explicit or implicit connection release time (also known as a burst). The second technique, proposed by Veeraraghavan et al. (Section 6.2.6), defines an end-to-end, circuit-switched network that is parallel to the packet-switched Internet. In their scheme only large files are transmitted across the circuit-switched network.

The approach proposed in Chapter 4, TCP Switching, differs from both approaches above in that it (usually) piggybacks the creation of a circuit on the setup phase of a TCP connection. In this respect, TCP Switching is similar to IP Switching (Section 6.2.7), in which flows trigger the establishment of ATM virtual circuits. In contrast, Chapter 5 focuses on the coarse circuits that interconnect boundary routers around the core. It monitors user flows to estimate the required capacity for those circuits.

6.2.1 Generalized Multi-Protocol Label Switching (GMPLS)

Multi-Protocol Label Switching (MPLS) [165] is a packet-switching technique proposed by the Internet Engineering Task Force (IETF) for traffic engineering that uses labels to identify flows. These flows may be of any granularity, ranging from fine user flows to coarse inter-router flows. Each flow follows a different label-switched path. Labels are identifiers that are local to each link, and so a flow label has to be swapped at each node with the local label for the next link.

GMPLS [7, 8] has been proposed within the Common Control and Measurement Plane (ccamp) work group in IETF as a way to extend MPLS to incorporate circuit switching in the time, frequency and space domains. Label-switched paths now may consist of a chain of SONET/SDH channels, wavelengths or fibers with a minimum capacity of at least 51 Mbit/s. The extensions of GMPLS define the signaling for the establishment, routing, protection, restoration, deletion and management of coarse label-switched paths that are circuit switched. As of April 2003, there are three published Requests For Comments (RFC's) on the standards track (one for the signaling functional description and two for the signaling protocols — CR-LDP and RSVP-TE —, which will be briefly described below). In addition, there are 20 Internet Drafts in progress.

GMPLS uses the same mechanisms as MPLS to decide when to create or destroy a circuit. GMPLS relies on either a User-to-Network Interface (UNI) or an MPLS traffic-engineering server (TE server) to issue requests for new label-switched paths (LSP's) or to modify the characteristics of existing LSP's. This traffic-engineering server is vendor specific, and it is usually at the ingress of the packet-switched MPLS network, where it collects traffic information to make its decisions. Alternatively, one could use an approach similar to the one described in Chapter 5 to manage the LSP's.

The differences between pure MPLS and the extensions of GMPLS come from the nature of the circuit-switched channels that GMPLS uses. The two major differences are, first, that in GMPLS the channel ID of the circuit-switched channels (e.g., the slot number in a TDM frame or the wavelength ID) can be used as an explicit path label, and, second, that the data and control channels may be completely decoupled in GMPLS (control information may be sent out-of-band, as opposed to an in-band MPLS shim header). In addition, GMPLS can only allocate bandwidth in discrete and coarse amounts, and there are usually many parallel data channels between two adjacent nodes (which was not originally considered in the IP or MPLS control planes). Finally, in GMPLS, nodes may have restrictions on what labels can be chosen (e.g., because of limited wavelength conversion capabilities).

The GMPLS extensions take all these differences into account. More precisely, these extensions consist of:

- a new Link Management Protocol (LMP) that monitors the connectivity of the data and control channels, and that localizes link or node failures [8, 103].
- enhancements to the link state advertisement of Open Shortest Path First (OSPF) and Intermediate System-to-Intermediate System (IS-IS) routing protocols to advertise the availability of circuit-switched resources in the network [8, 103].
- enhancements to Resource Reservation Protocol with Traffic Engineering (RSVP-TE) and Constraint-Based Routing Label Distribution Protocol (CR-LDP) to allow an LSP across a circuit-switched core to be requested with certain bandwidth and protection characteristics [138, 7, 104].

When a GMPLS node decides to establish a new LSP, it sends downstream an RSVP-TE `PATH` message (or a `Label Request` message if it uses CR-LDP) towards the destination. This message contains a generalized-label request with the desired bandwidth and (optionally) the desired protection level. The message is routed using a constrained-based shortest-path-first algorithm that uses the link state information flooded using OSPF or IS-IS, unless the `PATH/Label Request` message contains an explicit route. The downstream node sends back an RSVP-TE `Resv` message (or a `Label Mapping` message for CR-LDP) that includes the generalized label[1] that identifies the LSP.

[1] If the LSP is composed of several parallel channels, the downstream node may return one label for each channel.

GMPLS does not specify whether RSVP-TE or CR-LDP should be used, and it leaves to the vendors and carriers to decide. The main difference between RSVP-TE and CR-LDP is that RSVP-TE uses "soft state" to manage the paths (circuits are timed out unless the reservation is refreshed periodically), whereas CR-LDP uses "hard state" (an explicit message is required to destroy active circuits). Soft state has a higher signaling overhead and a looser control over resources, but it has a simpler recovery strategy under complex failure scenarios. GMPLS has also extended RSVP-TE to provide prompt notification of faults in the path.

Let us compare the signaling of GMPLS with that of TCP Switching (Chapter 4). In both RSVP-TE and CR-LDP, the ingress has to wait for the round-trip time of a two-way handshake to start sending data. In TCP Switching, the first packet in the flow is used to establish the circuit, and, consequently, there is no delay in sending the data. In addition, TCP Switching uses soft state without paying a penalty in signaling overhead since any activity in the data channel automatically refreshes the state of the circuit. In a sense, TCP Switching assumes semitransparent switches that can understand whether a channel is being used or not. This hardware support is not assumed to be present in GMPLS because many of its nodes switch information transparently.

GMPLS can create both uni- and bi-directional LSP's with a single `PATH/Label Request` message. In contrast, TCP Switching (like traditional MPLS) only works with purely unidirectional circuits. These bidirectional LSP's are useful for several important applications, such as telephony and private lines, and they also simplify path protection by having the two directions share their fate.

In GMPLS, like in MPLS, LSP's can be nested, and so a hierarchy of LSP's can be built to exploit the higher capacity of optical circuit switches, which have coarse channel granularities. The hierarchy is composed of packet-switched LSP's, TDM circuits, wavelengths and fibers, as shown in Figure 6.1. This use of a hierarchy of circuits is similar to the one proposed in Chapter 5.

Failure recovery is a very important requirement for carriers in GMPLS. GMPLS

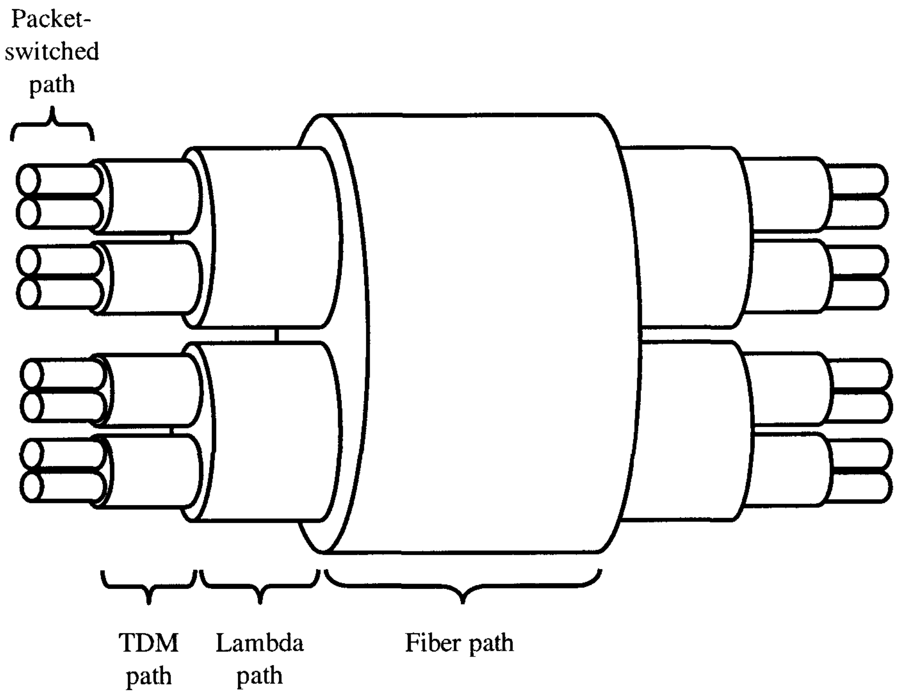

Figure 6.1: Hierarchy of label-switched paths in GMPLS.

can specify a different level of protection and restoration[2] for each LSP. There are different levels of failure recovery depending on the provisioning of additional resources (these resources can be pre-computed, pre-allocated or allocated on demand) and on the level of overbooking (protection resources can be dedicated, shared or best effort).

In summary, GMPLS proposes another way of integrating circuit switching in the core and packet switching in the edges. It focuses on the management of coarse circuits between core routers (like Chapter 5). However, its scope is slightly different than the contents of this thesis because it does not specify a control algorithm to decide when to create circuits and with what capacity. GMPLS also deals with many aspects, such as routing and path protection, that are out of the scope of this thesis.

[2] Protection refers to the extremely fast recovery from a failure (such as the 50 ms recovery time of SONET/SDH rings), whereas restoration is a slower failure recovery that relies on the regular signaling and routing mechanisms to re-establish the service.

6.2.2 ASTN: Automatic Switched Transport Network

ASTN (Automatic Switched Transport Network) [168] and ASON (Automatic Switched Optical Network) [167] are a set of Recommendations by Study Group 15 of the International Telecommunications Union — Telecommunication Standardization Sector (ITU-T) that specify the network architecture and the requirements for the signaling and routing in automatic switched transport networks. The network architecture is shown in Figure 6.2. The optical network has three planes: management, control and data transport.

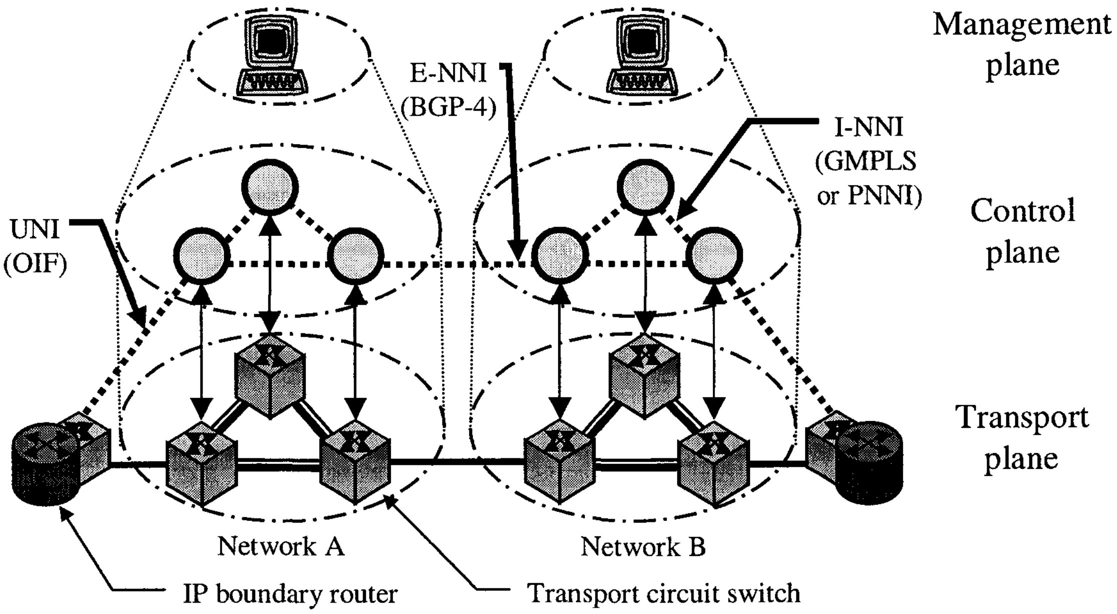

Figure 6.2: Network architecture of the Automatic Switched Transport Network (ASTN). UNI = User-to-Network Interface. I/E-NNI = Internal/External Network-to-Network Interface.

ASTN and ASON define the requirements in the control plane for the dynamic circuit provisioning (within minutes) and for the network survivability, protection and restoration. The goal is to specify a common control plane across multiple transport technologies that provides quality of service and equipment interoperability across domains and carriers.

ASTN and ASON do not develop new protocols when existing ones will do. Consequently, ASTN and ASON can either make use of GMPLS or PNNI [50, 117] as the signaling protocol. Although eleven standards have already been produced (detailing the architecture and the signaling requirements), the work of ASTN/ASON has not been completed except for the general framework.

6.2.3 OIF: Optical Internetworking Forum

The Optical Internetworking Forum (OIF) [13] is an industry forum composed of over 250 service providers and equipment vendors. It has defined a User-to-Network Interface (UNI) that allows user devices (i.e. edge routers and ATM switches) to dynamically request circuits between boundary devices through the circuit-switched optical network core. These circuits are provisioned rapidly with various levels of circuit protection and restoration. The OIF UNI also specifies signaling for automatic neighbor and service discovery, and for fault detection, localization and notification. For the moment, the work on OIF's UNI and IETF's GMPLS remain very complementary. In the future, OIF plans to specify a Network-to-Network Interface (NNI) that allows the direct interconnection of optical switches and networks from different vendors. OIF has already produced version 1.0 of its UNI and also several specifications for the electrical and very-short-reach optical interfaces between chips and between system elements. The OIF is not a formal standards body, but produces detailed specifications that are presented to traditional standards bodies (IETF and ITU-T) for adoption.

6.2.4 ODSI: Optical Domain Service Interconnect

The Optical Domain Service Interconnect (ODSI) is another industry forum that was started by several startups almost at the same time as OIF. However, ODSI lacked the participation by the large, established networking vendors and carriers, and so after merging its efforts with the OIF signaling workgroup, ODSI ceased to exist in late 2000. Like OIF, ODSI had also defined an optical UNI for edge routers and switches to request circuits from the core. The key difference between the two specifications was that ODSI developed a TCP-based signaling protocol, whereas OIF uses RSVP-TE or CR-LDP.

GMPLS, ASTN, OIF and ODSI share the same goal: to allow more dynamic, automated and standardized optical networks. However, they address different issues: ASTN has a top-down approach and focuses on the network architecture and requirements. The other three proposals define the components of the architecture: GMPLS specifies the routing, the topology and link state dissemination, and the NNI signaling, and OIF and ODSI define the UNI signaling and work on the equipment interoperability. The four efforts are aware of each other work and try to coordinate their efforts. For more information, Clavenna [44] has written a good overview of the differences between GMPLS, ASTN, OIF and ODSI.

6.2.5 Grid computing and *CA*Net 4*

Grid computing is a network of computation; i.e., a set of tools and protocols for coordinated problem solving and resource sharing among pooled assets. These pooled assets are known as virtual organizations, and they can be distributed around the world. The shared resources are heterogeneous and autonomous (they may belong different organizations), and their relation is temporary. The Global Grid Forum is a research forum in distributed computing that mirrors IETF and that is trying to standardize the grid-computing protocols and architectures under the Open Grid Services Architecture (OGSA) [76]. Globus Toolkit [80] is an open-source reference implementation of OGSA based on open standards from the web services world.

An example of a network designed for grid computing is CA*Net 4 [5]. It is part of the Canadian national research network, and it is targeted towards universities, research institutes and companies that need to exchange a good amount of data among different locations either regularly (e.g., a company with multiple sites) or for limited periods of time (e.g., for the duration of a joint project). CA*Net 4 is composed of a set of unorganized, point-to-point wavelengths[3] that are forwarded transparently by DWDM circuit switches. The network clients are big and sophisticated; they

[3] These wavelengths carry either SONET/SDH channels of 2.5-Gbit/s or 10-Gbit/s, or Ethernet channels of 1 Gbit/s or 10 Gbit/s.

either lease or own a subset of the unorganized wavelengths, and they operate the equipment that interconnects, translates and grooms those wavelengths to create their own private network. The network client has complete control over its own wavelengths and its network equipment, and it decides what gets added/dropped at the different exchange points.

The most interesting part of CA*Net 4 design is the business model. Clients only have to pay the capital cost of the dark fiber or wavelengths and the switching equipment, instead of the usual monthly service charge paid to traditional ISPs. Network connectivity is treated as a capital asset rather than a service as it is today. In addition, clients can sublease part of the bandwidth (at the STS-1 or Gbit-Ethernet granularity) in its own private circuit network through automated procedures.

In contrast, the proposals of Chapters 4 and 5 are for a public Internet infrastructure where resources are shared by all users. In addition, these two proposals are geared towards the unsophisticated end user who wants a service similar to the current public Internet without having to worry about the internals of the network, or having to hold a lease on parts of the network.

6.2.6 Proposal by Veeraraghavan et al.

Veeraraghavan et al. [181] define a circuit-switched network that reaches the end hosts and that runs in parallel to the packet-switched Internet. All traffic is sent through the packet-switched network, except when a long file needs to be transferred. Then, the end host creates a new end-to-end circuit in the circuit-switched network that is used for the long trasfer. Since this end-to-end circuit is a constant bandwidth channel that is solely reserved for that transfer, the transmission sees no losses due to queueing or contention, no packet reordering, and no delay jitter. As a result, a new transport protocol, called Zing, is proposed. This protocol has very simple error and flow control mechanisms. Another characteristic of the system is that the circuit-switching signaling is simple enough to be implemented in hardware.

The use of an end-to-end circuit-switched network has two problems: first and most importantly, as shown in Chapter 3, circuit switching in the access network yields very bad response times for end users since large file transfers eventually monopolize the link for long periods of time. Second, the cost of a second network with the corresponding links and switches is very large, and so it is unlikely that this solution will be widely deployed in the near future. This barrier to its deployment limits its attractiveness, since one can only use Zing to exchange files with the few nodes that are connected to the circuit-switched network. In contrast, the two approaches presented in Chapters 4 and 5 do not require any flag days, in which all network elements have to be upgraded or changed. Consequently, these two approaches can be deployed incrementally without any changes in either the access networks or the end hosts.

6.2.7 IP Switching

TCP Switching is most similar to IP Switching [129], in which user flows trigger the establishment of ATM virtual circuits. The main difference is that TCP Switching uses true circuits, as opposed to the use of the connection-oriented packet switching of ATM [117]. Consequently, TCP Switching can benefit from the much higher capacity of circuit switches.

IP Switching uses ATM virtual circuits, which is a packet-switching technique. With virtual circuits, resources are not necessarily reserved as with true circuits. Consequently, bandwidth is not wasted if the ATM virtual circuit remains active after the associated flow has ended. With TCP Switching, bandwidth is reserved, and it is wasted when unused. This wastage of bandwidth is relevant since typical flows in the Internet last only a few seconds. For this reason, the recommended inactivity timeouts of IP switching are above 30 seconds [108]; in contrast, TCP Switching uses a timeout that is only slightly larger than the RTT (0.25-1 s).

6.3 Packet switching in the optical domain

Chapters 4 and 5 and Section 6.2 have described two ways of using high-capacity all-optical circuit switches by integrating circuit-switched clouds with the rest of the

Internet that uses packet switching. Several researchers have proposed all-optical packet-switched routers instead.

El-Bawab and Shin [68] give an overview of the state of the art in the underlying technologies that used for all-optical packet switching, such as technologies for 3R[4] regeneration (SOA-based[5] Mach-Zehnder interferometers, soliton transmission, and self-pulsating distributed feedback lasers), packet delineation and synchronization (fiber delay lines), packet header processing (O/E[6] conversion, subcarrier multiplexing, and Michelson interferometers), optical buffering (fiber delay lines), optical space switching (SOAs, and $LiNbO_3$ crossconnects), and wavelength conversion (SOAs with cross-phase or cross-gain modulation, O/E/O conversion, and wave mixing).

El-Bawab and Shin state that major technological challenges need to be overcome before optical packet switching is viable. Many of the enabling technologies are still in the research and exploration stages, and so it is premature to build a commercial all-optical router. Buffering and per-packet processing are the basis for packet switching, and they remain the most important challenge to the implementation of an optical router. Through reflections, refractions and diffractions, we know how to bend, multiplex and demultiplex light, but we (still) do not know how to store as much information in optics as with an electronic DRAM, or how to process information in optics as fast as with an electronic ASIC. Current efforts in high-speed optical storage and processing [109, 151, 178] are still too crude and complex to be usable. With current optical storage approaches, information degrades fairly rapidly (the longest holding times are around 1 ms), and these approaches can only be tuned for specific wavelengths. In other areas, such as signal regeneration, packet synchronization, space crossconnects and wavelength conversion, progress has been made, but scalability, reliability and cost are still issues that need to be solved. In any case, even if some of the technology on which optical packet switching depends is not here yet, one can still study its performance to see what one can achieve once the technology has been developed.

[4] Reamplification, Reshaping and Retiming.

[5] Semiconductor Optical Amplifier.

[6] Electronics-to-Optics.

The family of solutions that does packet switching in optics can be further subdivided into two based on the size of the switching units: Optical Packet Switching (OPS) switches regular IP packets, whereas Optical Burst Switching (OBS) deals with "bursts", units that are larger and encapsulate several IP packets.

6.3.1 Optical Packet Switching (OPS)

Optical Packet Switching (OPS) [186, 185] is the simplest and most natural extension of packet switching over optics. It consists of sending IP packets directly over an all-optical backbone. The biggest challenge that packets face in an optical switch is the lack of large buffers for times of contention. As a rule of thumb, routers have $RTT \times bandwidth$ worth of buffering [182], so that TCP congestion control works well. For an OC-192c link and an average packet length of 500 bytes, this is equivalent to a buffer space of 625,000 packets. In contrast, existing optical buffering techniques based on fiber delay lines can accommodate at most a few tens of packets. With such small buffers, the packet drop rate of an optical packet switch is quite high even for moderate loads.

OPS tries to overcome the lack of buffers by combining two other techniques to solve contention: wavelength conversion and deflection routing. If two packets arrive simultaneously, and there are no local buffers left, the optical packet switch first tries to find another free wavelength in the same fiber, and if it cannot find it, it will try another fiber that does not have contention. The number of wavelengths is expected to be between 4 and 512, and the number of neighboring nodes fewer than 10.

OPS has some shortcomings: one is that we do not have much room to solve the contention. If we multiply the options given by the three dimensions (fiber delay lines, wavelength conversion and path deflection), we have less than (10−50 packets/FDL)× $(4 - 512$ wavelengths/fiber$) \times (2 - 10$ neighbors$) = 80 - 256,000$ options. It may seem to be close to the number of choices that we get from the electrical buffers in a router (625,000 packets for a 10-Gbit/s link), but the number of degrees of freedom is in fact much less since there are numerous dependencies that limit the choice. Moreover, packets that are bounced to different paths may cause congestion in other wavelengths or other parts of the network, spreading local congestion across larger areas of network. In addition, packets no longer follow the same path, and so they may arrive out of order, which may be interpreted by TCP as losses due to congestion, and TCP may thus throttle back its rate. Packet reordering within a TCP session also causes unnecessary retransmissions, prevents the congestion window from growing properly and degrades the quality of the RTT estimator in TCP [12, 17].

A problem that is perceived with OPS is that IP packet sizes are very short for some optical crossconnects to be rescheduled. A 40-byte packet takes 32 ns to be received on an OC-192c link, and only 8 ns on an OC-768c link. By contrast, MEMS mirrors have tilting times of over 1 ms. For this reason, several researchers have proposed using bigger switching units, called *bursts*, in an architecture called Optical Burst Switching.

6.3.2 Optical Burst Switching (OBS)

Optical Burst Switching (OBS) was proposed in [155, 177], and it is a hybrid between packet switching and circuit switching. OBS pushes buffers to the edges of the network, where electronic switches are, leaving no buffers in the optical core. OBS gathers bursts of data at the ingress nodes of the backbone using large electronic buffers until the node has enough data or a burst formation timeout occurs. At this point, the burst is sent through the all-optical core. In general, the burst is preceded by an out-of-band signaling message that creates a lightweight circuit with an explicit or implicit teardown time, through which the burst is sent, as shown in Figure 6.3. If the circuit is successfully created, the burst traverses the circuit, and then the circuit is destroyed once the burst has finished.

If during the circuit establishment there is no bandwidth left for the burst, the node can either temporarily buffer the burst using the limited space of local fiber delay lines or it can try to deflect the burst circuit to another wavelength or another fiber. If none of these three options is available the incoming burst is then dropped at that node. From the point of view of the user flows, the behavior of OBS is closer to OPS than to traditional circuit switching techniques. If there is contention,

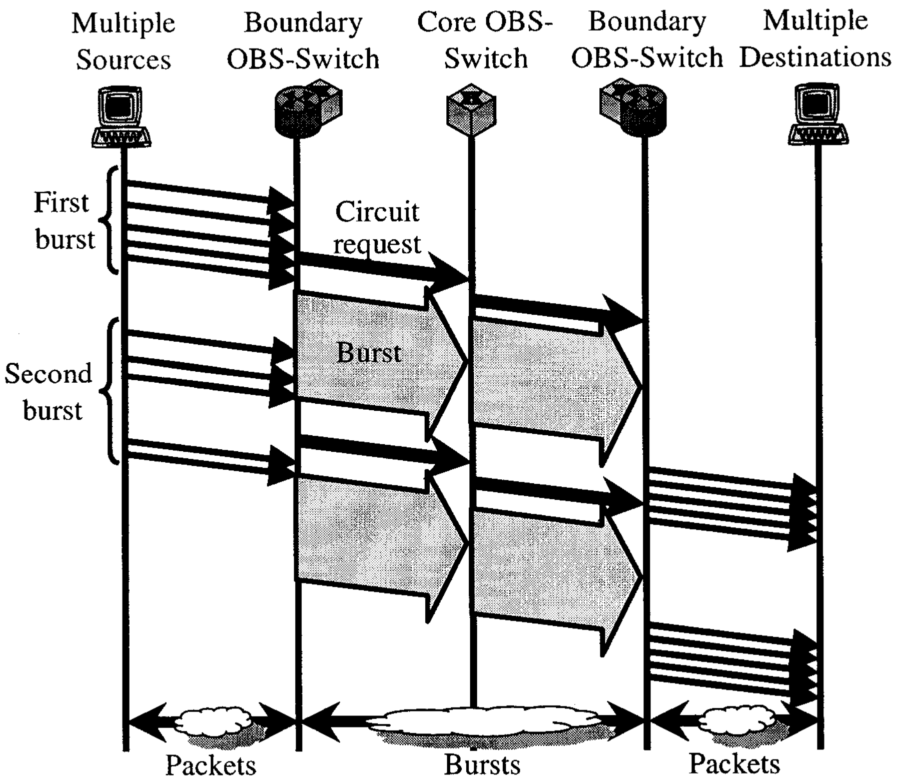

Figure 6.3: Sample time diagram of a network using Optical Burst Switching.

information from at least one active flow is dropped at the intermediate nodes in OPS; with traditional circuit switching, new flows are blocked (buffered) at the ingress, but old, active flows are unaffected. In traditional circuit switching, once a flow has been accepted, it is guaranteed a data rate and no contention. For this reason, the end user does not perceive OBS as a circuit switched network, but rather as a packet-switched one that switches large packets, as shown in Figure 6.3.

There are different types of OBS, essentially with different degrees of signaling complexity. The high rate of burst formation in the core makes the proposals with the simplest signaling the most interesting (i.e., those with "best-effort" reservation that do not wait for confirmations). The two most popular flavors of OBS are called Just-In-Time (JIT), which uses circuits with an open-ended duration and that are closed by an explicit "release" message from the ingress node, and Just-Enough-Time (JET), which explicitly specifies the circuit duration when the circuit is created [6].

With OBS, data is sent in batches as opposed to streamed as with regular IP or traditional circuit switching, such as the proposals of Chapters 4 and 5. This has an effect on TCP, since it relies on the packet timing to pace its transmissions. With OBS, delivery is best effort, and so the burst may be lost. Since TCP considers the loss of three consecutive packets as a sign of congestion, when burst sizes are long, the loss of a burst is expensive because it makes TCP sources throttle back their transmission rate. The effect of the burst loss rate is amplified by TCP. These two interactions of OBS with TCP are only noticeable when bursts are very long, when there are several packets belonging to the same user flow in each burst. TCP's flow and error control, thus, will set a limit on the maximum burst size that will depend on the rates under consideration.

OBS uses electrical buffers at the ingress to aggregate regular IP packets destined to the same egress node into bursts. The aggregation reduces the number of forwarding decisions that have to be done by the OBS so that they can be done electronically. The trade-off for this is that OBS requires more buffering at the ingress of the optical backbone than the optical circuit switching solutions because IP packets in OBS have to wait until the next burst departs, whereas with circuit switching, packets belonging to active circuits are sent as soon as they arrive. Furthermore, in TCP Switching, the circuits have the same capacity as the access link, hence they are not the bottleneck in the flow path. Consequently, queueing at the circuit head is unusual.

6.3.3 Performance of OPS/OBS

We can use the "end-user response time" to compare the performance of these two related techniques. Let me start with OBS. According to [139], If we ignore retransmission timeouts and operate in the absence of window-size limitations, we can write the average throughput of TCP as:

Average throughput $\propto \frac{1}{RTT \cdot \sqrt{p \cdot b}}$

where RTT is the round-trip time, p is the packet drop probability and b is the number of packets acknowledged per ACK message. The first thing to notice is that the longer the burst size is, the more TCP data and acknowledgement packets get bundled together in bursts of OBS, which makes the value of b increase. In addition, the small amount of buffers in OBS is not enough to solve the contention among bursts, and so the drop rate is larger than with regular packet switching in electronic form. For example, for a system load of 50% and four wavelengths per link, the drop rates for open-loop traffic with OBS are between 2% and 0.1% [186, 188], whereas the drop rates of electronic packet switching are typically several orders of magnitude lower. Using an $M/M/k/k + d$ model, where k is the number of wavelengths per link and d the number of fiber delay lines, Yoo et al. [188] show that the drop rate decreases exponentially with the number of wavelengths, k.

Furthermore, the burst-formation time in OBS increases the RTT, which reduces the average throughput of TCP and, thus, increases the user response time.[7] Simulations using ns-2 suggest that even when we use a long burst formation latency of 50 ms, OBS leads to response times that are only about 10% slower than electronic packet switching, and so one can conclude that their user response time performance is comparable.

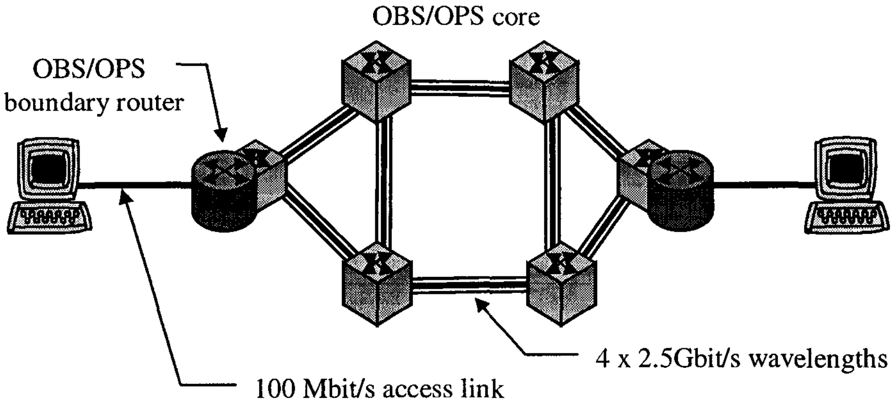

Figure 6.4: Topology used in [186] to simulate the effect of Optical Packet and Burst Switching on TCP. The core wavelengths were carrying bursty IP traffic in the background.

The previous arguements about the burst/packet losses in OBS/OPS seem to

[7] Remember that TCP Switching also had an increase of the RTT because of the transmission times over thin circuits. As the access-link rate increases in TCP Switching, the RTT increase becomes negligible.

question the end-user performance of OBS/OPS even under moderate loads because of the high losses in the unscheduled optical cloud. However, some authors [86, 188] have analyzed and performed open-loop simulations of OPS/OBS with unscheduled optical clouds, and they have found that the losses of the system are acceptable if enough wavelengths were available. For example, with a system load of 50% when the number of wavelengths per link went from 4 to 32, the packet loss rate when from 2% to $4 \cdot 10^{-5}$.

However, the close-loop, multiplicative-decrease-additive-increase congestion control algorithm of TCP can overreact to the clustered losses of OBS/OPS, and it can make TCP cut its transmission rate very aggressively. Moreover, the burst formation time has an important impact on the TCP throughput if it increases the connection RTT [64]. Yao et al. [186] have simulated what happens when an FTP session contends in an OPS/OBS, unscheduled optical core, such as the one shown Figure 6.4. Figure 6.5 shows the response time of file transfers of 1.6 Mbytes. One can see how the response time starts degrading with backbone loads of only 30%, and how, with backbone loads of only 50%, the response times of those FTP sessions using OPS is between 12 to 20 times worse than that of an unloaded network. Figure 6.5 also shows how OBS can achieve a better performance by aggregating packets into bursts, but the performance improvement is not enough to make the system usable under reasonable link loads. However, something should be said about these results; the system under consideration had only four wavelengths per link, so there is still room for improving the performance by adding more wavelengths per link. Today it is possible to switch over 320 wavelengths [173].

There have been several proposals [188, 186] to improve the dismal performance of OSP/OBS by creating several traffic classes with strict priorities or by giving priority to through traffic when it is contending with inbound traffic. The end result is that the high-priority class sees a network load that is much smaller than the total link load. It is as if all traffic of lower priority did not exist for the high priority class. For eight wavelengths per link, the high-priority class gets an acceptable performance (open-loop loss rate $\approx 4 \cdot 10^{-5}$) at the cost of heavily hurting the low-priority class, which gets an unacceptable performance, with loss rates of 20% for a total network

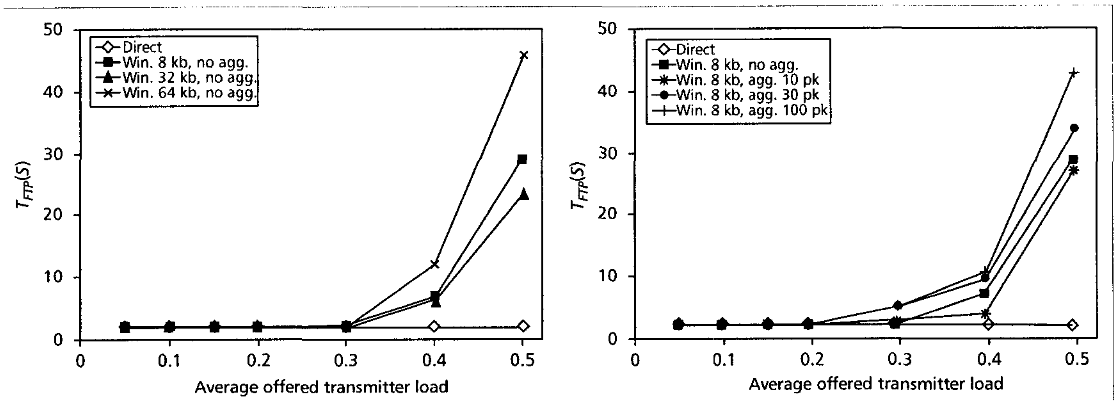

Figure 6.5: Response time of FTP sessions in Optical Packet and Burst Switching using TCP, as shown in Figure 7 in [186]. The diagram on the left studies the effect of the TCP receiver window size (8, 32 and 64 Kbytes), and the diagram on the right the effect of the burst size (1, 10, 30 and 100 packets). The "direct" curve uses regular packet switching with large electronic buffers in all nodes. The other curves use OBS/OPS with fiber loops, wavelength conversion and fiber deflection to resolve contention.

load of 60%.

Even if, on average, link loads are low in the core of the network, it is not a reasonable assumption on certain links (near hot spots) and at certain moments (e.g., after rerouting traffic following a link failure). Furthermore, hotspots and failures happen in unpredictable locations at unpredictable times [90]. OBS/OPS would not be able to provide the maximum performance where it is needed the most, unless the OBS/OPS is extremely overprovisioned by having many wavelengths per link.

6.4 Flow Measurement

A key component of Chapters 4 and 5 is the monitoring of flows and the study of their characteristics. This monitoring of flows has been done both off-line and on-line. The off-line profiling of user flows has been used, first, to see whether the proposed solutions make sense in the face of current Internet workloads and, second, to feed the model in Section 5.4. On the other hand, the on-line monitoring of active flows has been used to control the circuit-switched backbone in real time. Next, I will describe two approaches that can be used to study user flows.

6.4.1 RFC 2722 and NetFlow

RFC 2722 [22] provides a general framework for describing network traffic flows and presents an architecture for traffic flow measurement and reporting. The purpose of such flow-measurement system is to understand network usage and performance, which is in general done off-line, rather than to control the network in real time. Namely, such a flow measurement system can be used for network planning, performance and QoS estimation, and per-user billing.

There are two related tools that use sampling of packets to study flows. Cisco offers a feature in its routers called NetFlow [41] that logs in memory one packet out of every N packet arrivals[8] and later dumps the log to a permanent storage. There are numerous commercial and open-source programs that analyze off-line the traces sampled by NetFlow [171]. Duffield et al. [65] have proposed sampling flows with a frequency that is the inverse of the flow size to decrease the number of samples without introducing measurement errors.

6.4.2 Proposal by Estan and Varghese

Estan and Varghese [70] propose two methods that sample large flows (those that take a non-negligible amount of the link capacity) more precisely. One method samples packets at fixed arrival intervals, and it creates a filter for the flow of each sampled packet. All subsequent packets will try that filter. Large flows are more likely to have a filter in place when their packets arrive, and so they are more likely to be matched and sampled. The other method hashes each arriving packet using multiple hash functions. The value of each of the hash entries is increased with the packet size. A packet belonging to a large flow finds that the values of all its hash entries are large, whereas short flows most likely have some entry with a small value. These two methods use less memory than Cisco's NetFlow, and they accurately sample large flows, but they ignore many small flows.

[8] A recommended sampling rate is 1 packet out of 100.

As with the method listed in Section 4.3.3, the two methods described above require the observation of every single packet in the link. The difference between the two approaches is that Estan's methods use fewer filters by focusing on big flows, whereas the method of Section 4.3.3 uses many more filters because it measures how many flows are currently active, whether they are large or small. This latter information is then used to calculate the total flow capacity to properly size the circuit in the core in real-time. However, small flows typically take less than 20% of the aggregate rate, and so Estan's two methods can provide a rough estimate of the envelope of the total flow bandwidth with less state, but, as mentioned in Section 4.3.3, the amount of state related to all active flows (big or small) is not a big problem.

6.5 Conclusions

This chapter discusses other proposals that are related to this thesis. Some of these proposals try to integrate circuit switching in the core within a packet-switched Internet in a way that is similar to the one explored in this thesis: These approaches map flows between boundary routers to circuits. They develop other aspects of this integration that have not been addressed or elaborated in detail in this thesis, such as the protection, restoration and routing of circuits, or the statistical monitoring of flows. As such, these approaches could make use of the ideas developed in Chapters 4 and 5, and vice versa.

Other approaches, such as Optical Burst Switching and Optical Packet Switching, propose extending packet switching to all-optical switches.[9] They require an extremely overprovisioned network with hundreds of wavelengths per link to achieve performances that are comparable to those of electronic packet switching. Even if all the technological challenges that remain to get there are solved, the end user will not see a better response time from the network than with a traditional circuit switching

[9] Even if OBS uses circuit switching to forward the bursts, from the point of view of performance it behaves like a packet switching technology that switches very large packets (the bursts) using cut-trough techniques.

solution, as pointed in Chapter 3.

Chapter 7

Conclusions

As researchers in networking, we are continuously trying to eliminate any bottlenecks in the Internet by proposing and evaluating alternative protocols, algorithms or techniques. Frequently, we simply consider the functions in the current router architecture (classification, route lookup, per-packet processing, buffering and scheduling) in isolation. This dissertation looks at the router as a whole and it asks the following question: Can the underlying technology (electronics in Silicon) keep up with the pace of traffic growth? Figure 1.3 shows that the answer is clearly no. In 10 years time, there will be a five-fold gap between information forwarding in electronics and the backbone traffic volume.

There are already several architectures [36, 92, 93] that try to overcome the limitations of electronics by using load balancing and massive parallelism. However, this thesis takes a different approach, and it explores what would happen if we used optical switching elements, which are known to scale to capacities that are unimaginable with electronics. Optics can, indeed, overcome the gap between traffic growth and switching capacity. However, we cannot use the traditional packet-switch design for optical switches because we (still) do not know how to buffer light in large amounts.

One switching technique that is not affected by this drawback of optics is circuit switching because circuit switching moves all contention away from the data path, and, thus, it eliminates the need for buffering in the forwarding path. But, it is worth asking: What is the price to pay to use this technique? How will the efficiency, complexity and performance be affected? The first contribution of this thesis is a comparison of circuit and packet switching in the Internet, whether in electronics or optics. From analytical models, simulation and evidence from real networks, the conclusion is twofold:

- On one hand, circuit switching yields a very poor response time in access networks and LAN's with respect to packet switching. This is because of the blocking created by large file transfers when using circuits.

- On the other hand, in the core, circuit switching provides high reliability and scales better in capacity than packet switching without deteriorating the end-user response time or quality of service. The reason for this is that, first, circuit switches have a simpler data path and, second, the end-user response time is largely determined by the access links, which limits the maximum user-flow rate.

If we look at the backbone today, there is a lot of circuit switching in the form of SONET/SDH and DWDM switches. This thesis sustains that rather than disappear, these circuit switches will play a more relevant role in the future Internet. Currently, these core circuit switches are not integrated with the rest of the Internet, and IP treats the circuits as mere fixed-bandwidth, layer-2 paths between edge routers. In addition, these circuit switches are manually provisioned, and so it takes hours and even days to reconfigure them. They do react very slowly, and so they are vastly overprovisioned to account for any unexpected changes (for example, SONET/SDH provisions a parallel and disjoint path in a ring to accommodate for any sudden failure in the network). We would be better off if we had a circuit-switched system that reacts to the current network conditions in real-time.

The second contribution of this dissertation are two evolutionary approaches that integrate a circuit-switched core with the rest of an Internet that uses packet switching. The first approach (called TCP Switching) maps user flows to fine-grain, lightweight circuits in the core. The second approach monitors user flows to estimate the right size of the coarse-grain, heavyweight circuits that interconnect boundary routers around the core. This thesis uses user flows extensively to control the circuit switches in the backbone. The amount of per-flow state these techniques require is quite manageable with current technology, and it does not limit the performance of the switch.

A word of caution: The introduction of any dynamic algorithm for circuit management may be slow. Many carriers are reluctant to fully automate the provisioning of their backbone and to let some edge routers (potentially belonging to their clients) make decisions involving millions of dollars. These carriers would prefer to start with an automatic network management software that gives recommendations to network operators, who in turn use a point-and-click interface to rapidly reconfigure the network. Only when carriers feel confident enough with the decision-making algorithms will they let these algorithms run the network. I believe this last step is inevitable because, as networks grow and become more complex, it will be increasingly more difficult for human operators to react fast enough to changes in the network.

This thesis proposes only two of many possible ways of scaling the backbone to accommodate the growth of Internet traffic. Other related techniques that also use circuit switching in the core are the proposals by Veeraraghavan et al., GMPLS, ASTN/ASON, ODSI and OIF. A different set of techniques are Optical Burst Switching and Optical Packet Switching. They introduce optical switches in the backbone that perform packet switching of either large bursts of data or regular IP packets. OBS and OPS represent a big departure from the switching techniques that operators of the large transport networks currently use for the core (SONET/SDH and DWDM). It will not be easy for OBS/OPS to convince operators to adopt their network model, especially since these two approaches will not improve the performance seen by the end user, as discussed in Chapter 2.[1]

[1] It is interesting to note that despite the almost simultaneous coming of age of IP and SONET/SDH in the late 80's, current IP routers have not been able to displace electronic circuit switching in the core, as shown in Table 2.1. This could be an indication of what can happen with OBS/OPS.

7.1 Future directions

One important aspect of circuit switching that has not been addressed in this thesis is the routing of circuits in the backbone. Routing has important implications in the performance and scalability of a circuit-switched network. For example, a network can increase its throughput without increasing the total capacity if the traffic load is balanced across multiple parallel paths. Even if the set of routes is not optimal, the throughput can be much higher than the trivial shortest-path-first solution. Routing decisions need to be fast so as to be reactive to changes in the network traffic. Routing also performs an important role in the robustness of the network because in case of a failure, the routing mechanism has to restore the broken paths as soon as possible. One can speed up recovery if a disjoint backup path has been pre-computed and perhaps even pre-provisioned before any failure occurs.

Finally, the analysis of circuit and packet switching can have many other applications; especially, when comparing preemptive and non-preemptive systems with several parallel channels or servers. A short list of applications include:

- Router and switch crossconnects with few or no buffers. These crossconnects resemble a bufferless optical network where most or all buffers are at the ingress and egress points. The approaches that have been presented in this thesis could be applicable in this situation.

- Wireless access networks, in which orthogonal channels are used to increase the capacity of the access network.

- HTTP 1.1, where a client pipelines its requests through several parallel connections to a proxy server.

- Computer clusters, in which workloads are so large that it becomes very expensive to switch contexts, and so tasks need to be executed to completion.

7.2 Final words

Hopefully, the ideas in this dissertation will serve as a useful foundation for the design and architecture of future networks, and they will encourage further research on the integration of circuit and packet switching. This approach will allow us to use all-optical switches that scale and can cope with the rapid growth of Internet traffic.

Glossary

σ^2	Variance
$E[.]$	Expected value. Synonym for mean or average value
ASIC	Application Specific Integrated Circuits
ATM	Asynchronous Transfer Mode
CDF	Cumulative Probability Function
CMOS	Complementary Metal Oxide Semiconductor
DiffServ	Differentiated Services. RFC 2475
DRAM	Dynamic Random Access Memory
DSL	Digital Subscriber Line. A broadband access technology that works over local phone loops
DWDM	Dense Wavelength Division Multiplexing
FCFS	First Come, Fist Served, a scheduling discipline
FPGA	Field Programmable Gate Array
GMPLS	Generalized Multi-Protocol Label Swapping. Extension to MPLS that includes the use of circuits and wavelengths as paths
ICMP	Internet Control Message Protocol. RFC 792

IntServ	Integrated Services. RFC 1633
IP	Internet Protocol. A network protocol based on packet switching that is the basis for the Internet. RFC 791
ISP	Internet Service Provider
LAN	Local Area Network
M/GI/N	Queueing system with one queue and N servers. Arrivals are Poisson, and service times are independent and follow any generic distribution
M/M/N	Queueing system with one queue and N servers. Arrivals are Poisson, and service times are independent and follow an exponential distribution
MAN	Metropolitan Area Network
MEMS	Micro-Electro-Mechanical System
MPLS	Multi-Protocol Label Swapping. Protocol that associates paths to IP flows based on a label that is pre-pended to the packet. RFC 3031
OBS	Optical Burst Switching
OC-X	Optical Carrier. Specifies the SONET channel bandwidth in the optical domain. OC-1 = 51.85 Mbit/s
OC-Xc	Optical Carrier concatenated (as opposed to channelized)
OPS	Optical Packet Switching
PDF	Probability Density Function
PoP	Point of Presence

PrSh	Processor Sharing, a scheduling discipline
QoS	Quality of Service. Probabilistic measure that indicates whether the average delay, the delay jitter, the packet loss or the flow bandwidth are within a certain bounds
RAM	Random-Access Memory
RTT	Round-Trip Time. Time it takes for a packet to go from the source to the destination and back
SDH	Synchronous Digital Hierarchy
SJF	Shortest Job First, a scheduling discipline
SONET	Synchronous Optical NETwork
SRAM	Static Random-Access Memory
STS-X	Synchronous Transmission Structure. Specifies the SONET channel bandwidth in the electric domain. STS-1 = 51.85 Mbit/s
TCP	Transmission Control Protocol. End-to-end transport protocol that is responsible for verifying the reliable delivery of data. RFC 793
TDM	Time Division Multiplexing
TTL	Time To Live. Field in the IP header that eliminates routing loops by limiting the life of a packet in the network
UDP	User Datagram Protocol. RFC 768
WAN	Wide Area Network. Synonym for the backbone or Internet core
WDM	Wavelength Division Multiplexing
WFQ	Weighted Fair Queueing, a scheduling discipline

Bibliography

[1] Patrice Abry and Darryl Veitch. Wavelet analysis of long range dependent traffic. *IEEE Transactions on Information Theory*, 44(1):2–15, January 1998.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, August 2002.

[3] Alan Allan, Don Edenfeld, William H. Joyner, Jr., Andrew B. Kahng, Mike Rodgers, and Yervant Zorian. 2001 technology roadmap for semiconductors. *IEEE Computer Magazine*, 35(1):42–53, January 2002.

[4] Applied Micro Circuits Corporation, AMCC. *nPX5700, 10 Gbps Traffic Manager / Switch Fabric*, 2003. `http://www.amcc.com/cardiff/docManagement/displayProductSummary.jsp?prodId=nPX5700`.

[5] Bill St. Arnaud, Jing Wu, and Bahman Kalali. Customer controlled and managed networks. Technical report, Canarie, 2003.

[6] Ilia Baldine, George N. Rouskas, Harry H. Perros, and Dan Stevenson. Jumpstart: a just-in-time signaling architecture for WDM burst-switched networks. *IEEE Communications Magazine*, 40(2):82–89, February 2002.

[7] A. Banerjee, J. Drake, J. Lang, B. Turner, D. Awduche, L. Berger, K. Kompella, and Y. Rekhter. Generalized Multiprotocol Label Switching: An overview of signaling enhancements and recovery techniques. *IEEE Communications Magazine*, 39(1):144–150, January 2001.

[8] A. Banerjee, J. Drake, J. Lang, B. Turner, K. Kompella, and Y. Rekhter. Generalized Multiprotocol Label Switching: An overview of routing and management enhancements. *IEEE Communications Magazine*, 39(6):144–150, June 2001.

[9] Dhritiman Banerjee and Biswanath Mukherjee. Wavelength-routed optical networks: linear formulation, resource budgeting tradeoffs, and a reconfiguration study. *IEEE/ACM Transactions on Networking*, 8(5):598–607, 2000.

[10] Paul Baran. Introduction to distributed communications networks. Memorandum RM-3420-PR, Rand Corporation, August 1964.

[11] Paul Baran. On distributed communication networks. *IEEE Transactions on Communications*, 12(1):1–9, March 1964.

[12] Jon C. R. Bennett, Craig Partridge, and Nicholas Shectman. Packet reordering is not pathological network behavior. *IEEE/ACM Transactions on Networking (TON)*, 7(6):789–798, 1999.

[13] G. Bernstein, B. Rajagopalan, and D. Spears. OIF UNI 1.0 — controlling optical networks. White paper, Optical Internetworking Forum, 2001.

[14] Dimitri Bertsekas and Robert Gallager. *Data networks (2nd ed.)*. Prentice-Hall, Inc., 1992.

[15] David J. Bishop, C. Randy Giles, and Gary P. Austin. The Lucent LambdaRouter: MEMS technology of the future here today. *IEEE Communications Magazine*, 40(3):75–79, March 2002.

[16] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. RFC 2475: An architecture for differentiated services, December 1998.

[17] E. Blanton and M. Allman. On making TCP more robust to packet reordering. *ACM Computer Communication Review*, 32(1), January 2002.

[18] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. RFC 2205: Resource ReSerVation Protocol (RSVP) — version 1 functional specification, September 1997.

[19] Robert Braden. RFC 1122: Requirements for Internet hosts — communication layers, October 1989.

[20] Robert Braden, David Clark, and Scott Shenker. RFC 1633: Integrated Services in the Internet architecture: an overview, June 1994.

[21] Ira Brodsky. How IP-based networks will conquer telecom. *Network World*, May 1999.

[22] N. Brownlee, C. Mills, and G. Ruth. RFC 2722: Traffic flow measurement: Architecture, October 1999.

[23] Nevil Brownlee and kc claffy. Understanding Internet traffic streams: Dragonflies and tortoises. *IEEE Communications Magazine*, 40(10):110–117, October 2002.

[24] BT. BT world communications report 1998/9. Technical report, British Telecommunications, plc., May 1998.

[25] BTexact. Carrier requirements of core IP routers 2002. White paper, British Telecommunications, plc., February 2002. `http://www.btexact.com/white_papers/downloads/WP113.pdf`.

[26] International Bureau. Report on international telecommunications markets: 1999. Technical report, US Federal Communications Commission, January 2000.

[27] D. M. Burns, V. M Bright, S. C. Gustafson, and E. A. Watson. Optical beam steering using surface micromachined gratings and optical phase arrays. In *Proceedings of the SPIE*, pages 99–110, San Diego, CA, July 1999.

[28] Cahners. 2001 business ISPs: Service, size, and share. advanced carrier business report. Technical report, Cahners, October 2001.

[29] Cahners. Information alert newsletter. Volume #23. Technical report, Cahners, July 2001.

[30] CAIDA, Cooperative Association for Internet Data Analysis. *Mapnet: Macroscopic Internet Visualization and Measurement*, 2002. http://www.caida.org/tools/visualization/mapnet/.

[31] CAIDA, Cooperative Association for Internet Data Analysis. *OC48 analysis summary: Distributions of traffic stratified by application*, 2002. http://www.caida.org/analysis/workload/oc48/stats_20020109/apps_index.xml.

[32] Calient Networks. *The DiamondWave Family of Photonic Switches*, 2002. http://www.calient.net/files/DATA_SHEET.pdf.

[33] Jin Cao, William S. Cleveland, Dong Lin, and Don X. Sun. On the nonstationarity of internet traffic. In *Proceedings of ACM SIGMETRICS*, pages 102–112, 2001.

[34] CellStream. The "unofficial" MPLS service provider list. Technical report, CellStream, Inc., Global Consulting Services, August 2002. http://www.cellstream.com/MPLS_List.htm.

[35] C. David Chaffee. SONET vs. IP over photons: Debate and reality. *Business Communications Review*, pages 14–16, March 1999.

[36] Cheng-Shang Chang, Duan-Shin Lee, and Yi-Shean Jou. Load balanced birkhoff-von neumann switches, part i: one-stage buffering. *Computer Communications*, 25:611–622, 2002.

[37] H. Jonathan Chao, Cheuk H. Lam, and Eiji Oki. *Broadband Packet Switching Technologies: A Practical Guide to ATM Switches and IP Routers.* John Wiley & Sons, October 2001. ISBN: 0471004545.

[38] G. A. Chidi. VoIP taking 6 percent of international calls. *ITworld.com*, November 2001. http://www.itworld.com/Net/3303/IDG011106VoIPvolume/.

[39] Byung-Gon Chun. TCP Switch implementation. Cs344 class project report, Stanford University, 2001. http://klamath.stanford.edu/TCPSwitching/TCPSwitchImplementation_ByungGon.pdf.

[40] Ciena. *CIENA MultiWave CoreDirector*, 2001. `http://www.ciena.com/downloads/products/coredirector.pdf`.

[41] Cisco Systems. *Cisco IOS NetFlow Technology Data Sheet*, 2000. `http://www.cisco.com/warp/public/cc/pd/iosw/prodlit/iosnf_ds.pdf`.

[42] Cisco Systems. *Cisco 12416 Internet Router: Data Sheet*, 2001. `http://www.cisco.com/warp/public/cc/pd/rt/12000/12416/prodlit/itro_ds.htm`.

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

[44] Scott Clavenna. Optical signaling systems. *Light Reading*, January 2002. `http://www.lightreading.com/document.asp?site=lightreading&doc_id=7098`.

[45] William S. Cleveland, Dong Lin, and Don X. Sun. IP packet generation: statistical models for TCP start times based on connection-rate superposition. In *Proceedings of ACM SIGMETRICS*, pages 166–177, 2000.

[46] Matthew Coakeley. Virtual concatenation: Knowing the details. *ComDesign, an EE Times community*, November 2002. `http://www.commsdesign.com/design_corner/OEG20021112S0006`.

[47] Kerry Coffman and Andrew Odlyzko. *Handbook of Massive Data Sets*, chapter Internet growth: Is there a "Moore's Law" for data traffic? J. Abello, P. M. Pardalos, and M. G. C. Resende editors, Kluwer, 2001.

[48] Kerry Coffman and Andrew Odlyzko. *Optical Fiber Telecommunications IV B: Systems and Impairments*, chapter Growth of the Internet. I. P. Kaminow and T. Li, eds., Academic Press, 2002.

[49] J. W. Cohen. The multiple phase service network with generalized processor sharing. *Acta Informatica*, 12:245–284, February 1979.

[50] ATM Forum Technical Committee. Private network-network interface specification v.1.1. Technical report, ATM Forum, April 2002.

[51] T1 Committee. Synchronous Optical Network (SONET) - automatic protection switching. Technical Report T1.105.01-2000, ANSI Standard, March 2000.

[52] R. W. Conway, W. L. Maxwell, and L. W. Miller. *Theory of Scheduling.* Addison Wesley, Reading, MA, 1967.

[53] A. Copley. Optical Domain Service Interconnect (ODSI): Defining mechanisms for enabling on-demand highspeed capacity from the optical domain. *IEEE Communications Magazine*, 38(10):168–174, October 2000.

[54] Corvis. *Corvis ON (All-Optical switch)*, 2002. `http://www.corvis.com/Corvis/media/rl/ON_LR.pdf`.

[55] J. Cowie, A. Ogielski, B. Premore, and Y. Yuan. Global routing instabilities during Code Red II and Nimda worm propagation. `http://www.renesys.com/projects/bgp_instability`, September 2001.

[56] Mark Crovella and Azer Bestavros. Self-similarity in World Wide Web traffic: Evidence and possible causes. In *Proceedings of SIGMETRICS'96: The ACM International Conference on Measurement and Modeling of Computer Systems.*, Philadelphia, Pennsylvania, May 1996.

[57] Mark Crovella, Murad Taqqu, and Azer Bestavros. *A Practical Guide To Heavy Tails: Statistical Techniques and Applications*, chapter Heavy-Tailed Probability Distributions in the World Wide Web, pages 3–26. R. Adler, R. Feldman, and M. Taqqu, editors. Birkhäuser Verlag, Boston, 1998.

[58] Vinodh Cuppu, Bruce L. Jacob, Brian Davis, and Trevor N. Mudge. A performance comparison of contemporary DRAM architectures. In *Proceedings of ACM/IEEE ISCA*, pages 222–233, 1999.

[59] A. Daum. Broadband: The revolution's on hold in Europe just now. Technical report, GartnerG2, Inc., 2001.

[60] Dell'Oro. DSL's road to recovery begins in 2003, according to Dell'Oro Group 5 year forecast. Press release, Dell'Oro Group, July 2002.

[61] Dell'Oro. Ethernet switch market grew 16% in 4Q01, according to Dell'Oro Group. Press release, Dell'Oro Group, February 2002.

[62] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair-queueing algorithm. In *Proceedings of ACM SIGCOMM*, pages 1–12, Austin, TX, September 1989.

[63] Information Technology Dept. *Traffic between Purdue University and the Indiana GigaPoP*. Purdue University, 2003. `http://mrtg.noc.purdue.edu/data/math-g190-c6509-01/math-g190-c6509-01-ge-gigapop.html`.

[64] Andrea Detti and Marco Listanti. Impact of segments aggregation on TCP Reno flows in optical burst switching networks. In *Proceedings of IEEE Infocom*, pages 1803–1812, 2002.

[65] N. Duffield, C. Lund, and M. Thorup. Charging from sampled network usage. In *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, November 2001.

[66] Chris Edwards. Panel weighs hardware, software design options. *EE Times*, June 2000. `http://www.eetimes.com/story/OEG20000607S0043`.

[67] S. G. Eick, P. Schuster, A. Mockus, T. L. Graves, and A. F. Karr. Visualizing software changes. Technical Report 113, National Institute of Statistical Sciences, December 2000.

[68] Tarek S. El-Bawab and Jong-Dug Shin. Optical packet switching in core networks: Between vision and reality. *IEEE Communications Magazine*, 40(9):60–65, September 2002.

[69] David Emberley, Sterling Perrin, and Thomas S. Valovic. More is not enough: Bandwidth end use forecast and analysis, 2000-2005. Analyst brief, IDC, February 2002.

[70] C. Estan and G. Varghese. New directions in traffic measurement and accounting. In *Proceedings of ACM SIGCOMM*, pages 323–336, 2002.

[71] EZchip, Technologies. *NP-1, OC-192 Network Processor*, 2003. `http://www.ezchip.com/html/pr_np-1.html`.

[72] M. D. Fagen. *A History of Engineering and Science in the Bell System: The Early Years (1875-1925)*. Bell Telephone Laboratories, New York, 1975.

[73] Anja Feldmann. *Self-similar Network Trac and Performance Evaluation*, chapter Characteristics of TCP connection arrivals, pages 367–399. K. Park and W. Willinger, editors. John Wiley and Sons, 2000.

[74] Anja Feldmann, Jennifer Rexford, and Ramon Cáceres. Reducing overhead in flow-switched networks: An empirical study of web traffic. In *Proceedings of IEEE Infocom*, pages 1205–1213, 1998.

[75] Domenico Ferrari. Real-time communication in an internetwork. *Journal of High Speed Networks IOS Press*, 1(1):79–103, 1992.

[76] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The physiology of the grid: An open grid services architecture for distributed systems integration. Technical report, Open Grid Service Infrastructure WG, Global Grid Forum, June 2002.

[77] Charles Fraleigh. *Provisioning IP Backbone Networks to Support Delay Sensitive Traffic.* PhD thesis, Electrical Engineering Dept., Stanford University, 2002.

[78] S. Ben Fredj, T. Bonald, A. Proutiere, G. Regnie, and J. Roberts. Statistical bandwidth sharing: A study of congestion at flow level. In *Proceedings of ACM SIGCOMM*, pages 111–122, 2001.

[79] Aysegul Gençata and Biswanath Mukherjee. Virtual-topology adaptation for WDM mesh networks under dynamic traffic. In *Proceedings of IEEE Infocom*, volume 1, pages 48–56, June 2002.

[80] The Globus Project. *Globus Toolkit*, 2003. `http://www.globus.org/toolkit/`.

[81] Walter J. Goralski. *SONET, 2nd Edition.* McGraw-Hill Professional, 2000.

[82] Pakaj Gupta. *Algorithms for Routing Lookups and Packet Classification.* PhD thesis, Computer Science Dept., Stanford University, 2001.

[83] P. M. Hagelin, U. Krishnamoorthy, J. P. Heritage, and O. Solgaard. Cross-connect switch using micromachined mirrors. *IEEE Photonics Technology Letters*, 12(7):882–885, July 2000.

[84] Mor Harchol-Balter and Allen B. Downey. Exploiting process lifetime distributions for dynamic load balancing. *ACM Transactions on Computer Systems*, 15(3):253–285, 1997.

[85] Martin Hoffmann, Peter Kopka, and Edgar Voges. Low-loss fiber-matched low-temperature PECVD waveguides with small-core dimensions for optical communication systems. *IEEE Photonic Technology Letters*, 9(9):1238–1240, 1997.

[86] Ching-Fang Hsu, Te-Lung Liu, and Nen-Fu Huang. Performance analysis of deflection routing in optical burst-switched networks. In *Proceedings of IEEE Infocom*, pages 66–74, 2002.

[87] Frank Ianna. *Ianna Outlines Plan to Evolve the AT&T Network.* AT&T, March 1999. `http://www.att.com/technology/ip/iannaplan.html`.

[88] Industry Analysis Division, Common Carriers Bureau. Trends in telephone service. Report, US Federal Communications Commission, August 2001.

[89] Information Sciences Institute. *The Network Simulator, ns-2*, 2002. `http://www.isi.edu/nsnam/ns/`.

[90] Sundar Iyer, Supratik Bhattacharyya, Nina Taft, Nick McKeown, and Christophe Diot. An approach to alleviate link overload as observed on an IP backbone. In *Proceedings of IEEE Infocom*, San Francisco, California, April 2003.

[91] Sundar Iyer, Ramana Rao Kompella, and Nick McKeown. Analysis of a memory architecture for fast packet buffers. In *IEEE Workshop on High Performance Switching and Routing*, Dallas, Texas, May 2001. IEEE Workshop on High Performance Switching and Routing.

[92] Sundar Iyer and Nick McKeown. Making parallel packet switches practical. In *Proceedings of IEEE Infocom*, pages 1680–1687, Ancorage, Alaska, March 2001.

[93] Sundar Iyer and Nick McKeown. Maintaining packet order in two-stage switches. In *Proceedings of IEEE Infocom*, pages 1032–1042, New York, NY, June 2002.

[94] Juniper Networks. *T640 Internet Routing Node: Datasheet*, 2002. `http://www.juniper.net/products/dsheet/100051.html`.

[95] R.E. Kahn, S.A. Gronemeyer, J. Burchfiel, and R.C. Kunzelman. Advances in packet radio technology. *Proceedings of the IEEE*, 66(11):1468–1496, November 1978.

[96] Leonard Kleinrock. *Queuing Systems, Volume I: Theory*, volume 1. Wiley-Interscience, New York, 1975.

[97] Leonard Kleinrock. Principles and lessons in packet communications. *Proceedings of the IEEE*, 66(11):1320–1329, November 1978.

[98] Paul Korzeniowski. VoIP-still only a drop in the bucket. *Voice 2001, Business Communications Review*, pages 78–80, February 2001.

[99] Eric Krapf. Can they really rebuild the PSTN? *Business Communications Review*, pages 36–44, May 2000.

[100] Jason Krause. How low can they go? *The Industry Standard*, September 1999.

[101] U. Krishnamoorthy, K. Li, K. Yu, D. Lee, J.P. Heritage, and O. Solgaard. Dual mode micromirrors for optical phased array applications. *Sensors and Actuators: A. Physical*, 97-98C:22–26, May 2002.

[102] R. Kuhn. Sources of failure in the public switched telephone network. *IEEE Computer*, 30(4):31–36, April 1997.

[103] Editor L. Berger. RFC 3471: Generalized Multi-Protocol Label Switching (GMPLS), Signaling Functional Description, January 2003.

[104] Editor L. Berger. RFC 3473: Generalized Multi-Protocol Label Switching (GMPLS) Signaling, Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions, January 2003.

[105] C. Labovitz, R. Wattenhofer, S. Venkatachary, and A. Ahuja. Resilience characteristics of the Internet backbone routing infrastructure. In *Proceedings of the Third Information Survivability Workshop*, Boston, MA, October 2000.

[106] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. Delayed Internet routing convergence. *IEEE/ACM Transactions On Networking*, 9(3):293–306, June 2001.

[107] Craig Labovitz, Abha Ahuja, and Farnam Jahanian. Experimental study of Internet stability and wide-area backbone failures. In *Proceedings of FTCS*, Madison, WI, June 1999.

[108] Steve Lin and Nick McKeown. A simulation study of ip switching. In *Proceedings of ACM SIGCOMM*, pages 15–24, Cannes, France, September 1997.

[109] C. Liu, Z. Dutton, C. H. Behroozi, and L. V. Hau. Observation of coherent optical information storage in an atomic medium using halted light pulses. *Nature*, 409:490–493, January 2001.

[110] Peter Lothberg. A view of the future: The IP-only Internet. NANOG meeting #22, May 2001. `http://www.nanog.org/mtg-0105/lothberg.html`.

[111] Lucent Technologies. *WaveStar OLS 1.6T Brochure*, 2001. `http://www.lucent.com/livelink/152114_Brochure.pdf`.

[112] Lucent Technologies. *LambdaXtreme Transport*, 2002. `http://www.lucent.com/livelink/0900940380004c3f_Brochure_datasheet.pdf`.

[113] G. Varghese M. Shreedar. Efficient fair queuing using deficit round robin. In *Proceedings of ACM SIGCOMM*, pages 231–242, Cambridge, MA, September 1995.

[114] Ratul Mahajan, David Wetherall, and Tom Anderson. Understanding BGP misconfiguration. In *Proceedings of ACM SIGCOMM*, 2002.

[115] Matrix.net. *Internet Ratings*, 2002. `http://ratings.miq.net/`.

[116] N. Maxemchuk, I. Ouveysi, and M. Zukerman. A quantitative measure of topology lifetime for telecommunications networks. In *Proceedings of the Conference on Global Communications (GLOBECOM)*, volume 1, pages 690–694, December 2000.

[117] David E. McDysan and Darren L. Spohn. *ATM Theory and Applications.* McGraw-Hill Osborne Media, 1998.

[118] McKinsey&Company and Goldman Sachs. *US communications infrastructure at a crossroads: oportunities among gloom*, August 2001.

[119] Lee McKnight and Brett Leida. Internet telephony: Costs, pricing, and policy. In *Twenty-fifth Annual Telecommunications Policy Research Conference, Alexandria, VA*, September 1997.

[120] Alberto Medina, Nina Taft, Kave Salamatian, Supratik Bhattacharyya, and Christophe Diot. Traffic matrix estimation: Existing techniques compared and new directions. In *Proceedings of ACM SIGCOMM*, pages 161–174, August 2002.

[121] Merrill Lynch. *Optical Systems*, August 2002. Technical Report.

[122] Dejan Milojicic, Erik Brewer, Fred Douglis, Peter Druschel, Gary Herman, and Munindar Singh. Internet technology. *IEEE Concurrency*, 8(1), January - March 2000.

[123] Cyriel Minkenberg. *On packet switch design.* PhD thesis, Eindhoven University of Technology, 2001.

[124] Partha P. Mitra and Jason B. Stark. Nonlinear limits to the information capacity of optical fibre communications. *Nature*, 411:1027–1030, June 2001.

[125] Pablo Molinero-Fernández. *Ns-2 models used in the TCP Switching simulations*, 2002. `http://klamath.stanford.edu/TCPSwitching`.

[126] C. Siva Ram Murthy and Mohan Gurusamy. *WDM Optical Networks: Concepts, Design, and Algorithms.* Prentice Hall, 2001.

[127] National Telecommunications and Information Administration. Falling through the net: Defining the digital divide. Technical report, US Department of Commerce, 1999.

[128] NetEconomy. Circuit vs. packet: the debate intensifies. *the NetEconomy*, October 2001.

[129] Peter Newman, Greg Minshall, and Thomas L. Lyon. IP Switching — ATM under IP. *IEEE/ACM Transactions on Networking*, 6(2):117–129, 1998.

[130] T. H. Ning. Why BiCMOS and SOI BiCMOS? *IBM Journal of Research and Development*, 46(2/3), 2002.

[131] NLANR. *NLANR network traffic packet header traces*, 2001. `http://moat.nlanr.net/Traces/`.

[132] Nortel Networks. *OPTera Connect HDX optical switch*, 2002. `http://www.nortelnetworks.com/products/01/optera/connect/hdx/techspec.html`.

[133] Nua Internet Surveys. *How Many On-line?*, April 2002. `http://www.nua.ie/surveys/how_many_online/n_america.html`.

[134] Mike O'Dell. Keynote speech. ACM SIGCOMM conference, August 2002.

[135] Andrew Odlyzko. Data networks are mostly empty and for good reason. *IT Professional*, 1(2):67–69, Mar/Apr 1999.

[136] Andrew Odlyzko. Content is not king. *First Monday*, 6(2), February 2001. `http://www.firstmonday.dk/issues/issue6_2/odlyzko/`.

[137] Opnix, Inc. *The Internet Traffic Report*, 2002. `http://www.internettrafficreport.com/`.

[138] Editor P. Ashwood-Smith and Editor L. Berger. RFC 3472: Generalized Multi-Protocol Label Switching (GMPLS) Signaling, Constraint-based Routed Label Distribution Protocol (CR-LDP) Extensions, January 2003.

[139] Jitedra Padhye, Victor Firoiu, Don Towsley, and Jim Krusoe. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of ACM SIGCOMM*, pages 303–314, Vancouver, CA, September 1998.

[140] Konstantina Papagiannaki, Nina taft, Zhi-Li Zhang, and Christophe Diot. Long-term forecasting of Internet backbone traffic: Observations and initial models. In *Proceedings of IEEE Infocom*, San Francisco, California, April 2003.

[141] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.

[142] David Passmore. Why convergence will succeed. *Business Communications Review*, pages 16–18, November 1999. `http://www.bcr.com/bcrmag/1999/11/p16.asp`.

[143] David Patterson, Thomas Anderson, Neal Cardwell, Richard Fromm, Kimberly Keeton, Christoforos Kozyrakis, Randi Thomas, and Katherine Yelick. A case for intelligent RAM. *IEEE Micro Magazine*, 17(2):34–44, April 1997.

[144] David Patterson and John Henessy. *Computer Architecture. A Quantitative Approach.* Morgan Kaufmann Publishers, second edition edition, 1996.

[145] Vern Paxson and Mark Allman. RFC 2988: Computing TCP's retransmission timer, November 2000.

[146] Vern Paxson and Sally Floyd. Wide area traffic: the failure of Poisson modling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, 1995.

[147] A. L. Penenberg. The war for the poor. *Forbes Magazine*, September 1997.

[148] Radia Perlman. *Interconnections: Bridges, Routers, Switches, and Internetworking Protocols (2nd edition).* Addison-Wesley, 1999. ISBN: 0201634481.

[149] B. Pesach, G. Bartal, E. Refaeli, A. J. Agranat, J. Krupnik, and D. Sadot. Free-space optical cross-connect switch by use of electroholography. *Applied Optics*, 39(5):746–758, February 2000.

[150] Christian Schmutzer Peter T́omsu. *Next Generation Optical Networks: The Convergence of IP Intelligence and Optical Technologies.* Prentice Hall, 1st edition, 2001. ISBN: 013028226X.

[151] D. F. Phillips, A. Fleischhauer, A. Mair, R. L. Walsworth, and M. D. Lukin. Storage of light in atomic vapor. *Physical Review Letters*, 86(5):783–786, 2001.

[152] Jon Postel. RFC 791: IP: Internet Protocol, September 1981.

[153] Jon Postel. RFC 793: TCP: Transmission Control Protocol, September 1981.

[154] Private communication. Source requested not to be identified, April 2002.

[155] C. Qiao and M. Yoo. Optical burst switching (OBS) - a new paradigm for an optical Internet. *IEEE Journal of High Speed Networks*, 8(1), 1999.

[156] Light Reading. Optical provisioning: Light years ahead. *Light Reading*, August 2000.

[157] RHK. Diversification paid off as ON market shrank in 2001. Report #1079, RHK, Telecommunication Industry Analysis, May 2002.

[158] RHK. IP edge: Cisco slips in 2001, Juniper triples its share. Report #1101, RHK, Telecommunication Industry Analysis, May 2002.

[159] RHK. Nortel claims top spot in race for MSS market share. Report #1102, RHK, Telecommunication Industry Analysis, May 2002.

[160] RHK. North American telecom capex to turn up in 2004. Press release #154, RHK, April 2002.

[161] RHK. Special report: Juniper makes gains in core and edge router markets. Industry News #114, RHK, Telecommunication Industry Analysis, February 2002.

[162] RHK. United States Internet traffic experiences annual growth of 100%, but just 17% revenue growth. Press release #157, RHK, Telecommunication Industry Analysis, May 2002.

[163] Fabio Ricciato, Stefano Salsano, Angelo Belmonte, and Marco Listanti. Off-line configuration of a MPLS over WDM network under time-varying offered traffic. In *Proceedings of IEEE Infocom*, volume 1, pages 57–65, June 2002.

[164] L. G. Roberts. The evolution of packet switching. *Proceedings of the IEEE*, 66(11):1307–1313, November 1978.

[165] E. Rose, A. Viswanathan, and R. Callon. RFC 3031: Multiprotocol Label Switching Architecture, January 2001.

[166] ITU Telecommunication Standardization Sector. *Network node interface for the synchronous digital hierarchy (SDH).* International Telecommunication Union, Recommendation G.707/Y.1322 edition, 2000.

[167] ITU Telecommunication Standardization Sector. *Architecture for the Automatic Switched Optical Network (ASON).* International Telecommunication Union, Recommendation G.8080/Y.1304 edition, November 2001.

[168] ITU Telecommunication Standardization Sector. *Architecture for the Automatic Switched Transport Network (ASTN).* International Telecommunication Union, Recommendation G.807/Y.1302 edition, November 2001.

[169] ITU Telecommunication Standardization Sector. *Link capacity adjustment scheme (LCAS) for virtual concatenated signals.* International Telecommunication Union, Recommendation G.7042/Y.1305 edition, February 2003.

[170] Sprint ATL. *Sprint network traffic flow traces*, 2002. `http://www.sprintlabs.com/Department/IP-Interworking/Monitor/`.

[171] SWITCH, Swiss Education and Reseach Network. Floma: Pointers and software. `http://www.switch.ch/tf-tant/floma/software.html`.

[172] Andrew S. Tanenbaum. *Computer networks (3rd ed.).* Prentice-Hall, Inc., 1996.

[173] Lucent Technologies. Lucent Technologies announces record-breaking 320-channel optical networking system. Press release, Lucent Technologies, April 2000.

[174] Tellium. *Aurora optical switch*, 2002. `http://www.tellium.com/documents/brochures/aos.pdf`.

[175] F. A. Tobagi et al. Modeling and measurements techniques in packet communication networks. *Proceedings of the IEEE*, 66(11):1423–1447, November 1978.

[176] Massimo Tornatore, Guido Maier, and Achille Pattavina. WDM network optimization by ILP based on source formulation. In *Proceedings of IEEE Infocom*, volume 1, June 2002.

[177] J. Turner. Terabit burst switching. *IEEE Journal of High Speed Networks*, 8(1), 1999.

[178] A. V. Turukhin, V. S. Sudarshanam, M. S. Shahriar, J. A. Musser, B. S. Ham, and P. R. Hemmer. Observation of ultraslow and stored light pulses in a solid. *Physical Review Letters*, 88(023602), 2002.

[179] University of Maryland. *The Code Decay Project.* `http://www.cs.umd.edu/~aporter/html/evolution.html`.

[180] US Census. Industry quick report. Technical report, US Department of Commerce, 1997. `http://factfinder.census.gov/servlet/IQRBrowseServlet`.

[181] M. Veeraraghavan, M. Karol, R. Karri, R. Grobler, and T. Moors. Architectures and protocols that enable new applications on optical networks. *IEEE Communications Magazine*, 39(3):118–127, March 2001.

[182] C. Villamizar and C. Song. High performance TCP in ANSNET. *ACM Computer Communication Review*, 24(5):45–60, 1994.

[183] Walter Willinger, Vern Paxson, and Murad Taqqu. *A Practical Guide to Heavy Tails: Statistical Techniques and Applications*, chapter Self-similarity and Heavy Tails: Structural Modeling of Network Traffic, pages 27–54. R. Adler, R. Feldman, and M. S. Taqqu, editors. Birkhäuser Verlag, Boston, 1998.

[184] The Yankee Group. *Operational Costs of IP networks; Service Providers' experiences and requirements*, 2001.

[185] Shun Yao, Biswanath Mukherjee, S. J. Ben Yoo, and Sudhir Dixit. A unified study of contention-resolution schemes in optical packet-switched networks. *accepted for publication in Journal of Lightwave Technology*, April 2003.

[186] Shun Yao, Fei Xue, Biswanath Mukherjee, S. J. Ben Yoo, and Sudhir Dixit. Electrical ingress buffering and traffic aggregation for optical packet switching and its effect on TCP-level performance in optical mesh networks. *IEEE Communications Magazine*, 40(9):66–72, September 2002.

[187] H. Yasaka, H. Sanjoh, H. Ishii, Y. Yoshikuni, and K. Oe. Repeated wavelength conversion of 10 Gb/s signals and converted signal gating using wavelength-tunable semiconductor lasers. *IEEE Journal of Lightwave Technology*, 14(6):1042–1047, June 1997.

[188] M. Yoo, C. Qiao, and S. Dixit. Optical burst switching for service differentiation in the next-generation optical Internet. *IEEE Communications Magazine*, 39(2):98–104, February 2001.

[189] Yin Zhang, Lee Breslau, Vern Paxson, and Scott Shenker. On the characteristics and origins of Internet flow rates. In *Proceedings of ACM SIGCOMM*, pages 161–174, August 2002.