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.

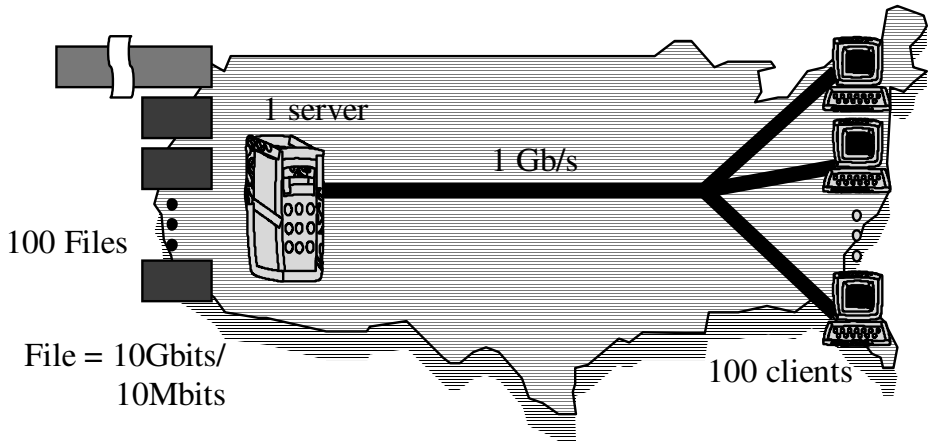

Figure 3.1: Network scenario for both motivating examples. The bottleneck link is a transcontinental link of 1 Gbit/s. In the first scenario, all files are of the same length (10 Mbits). In the second scenario, the first file that gets requested is 1000 times longer (with a size of 10 Gbits) than the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

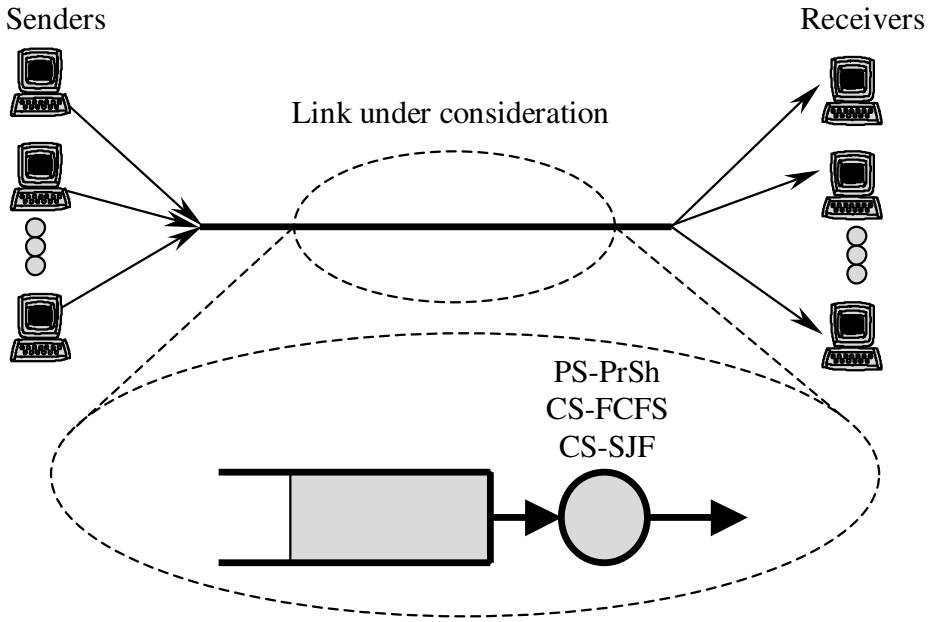

Figure 3.2: Queueing model used to analyze a bottleneck link using circuit and packet switching.

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.

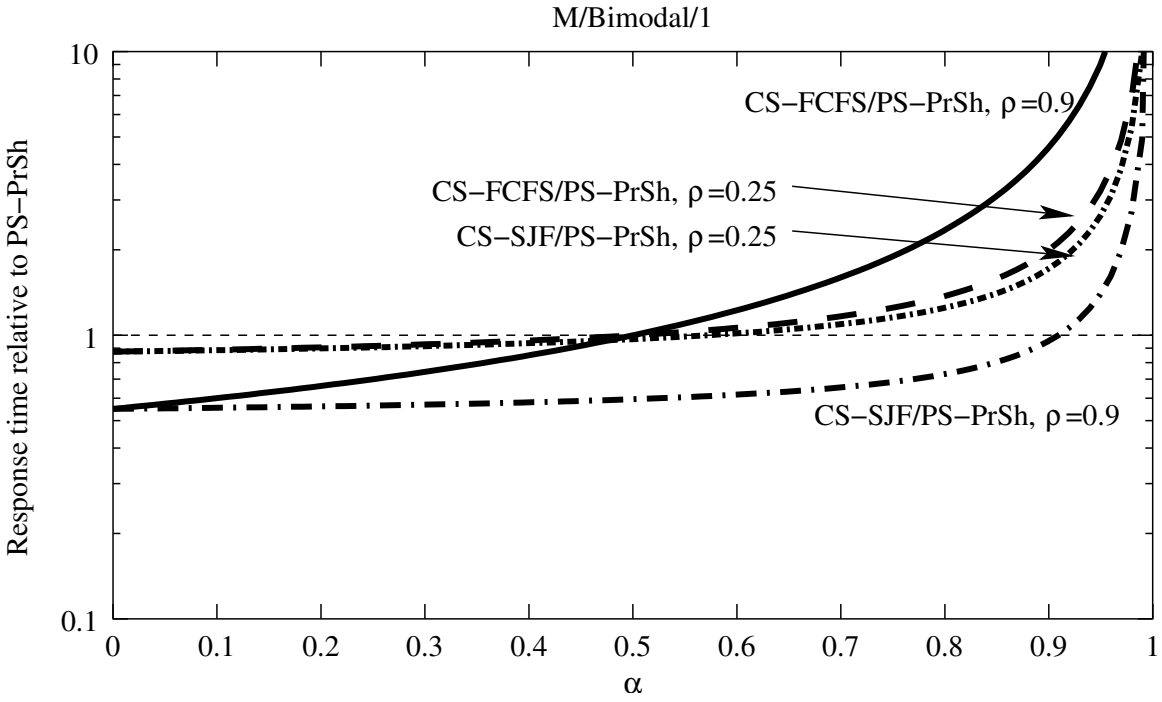

Figure 3.3: Relative average response time of CS-FCFS and CS-SJF with respect to PS-PrSh for a single server. Arrivals are Poisson and flow sizes are bimodal with parameter α. Link loads are $\rho = 0.25$ and $\rho = 0.9$.

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

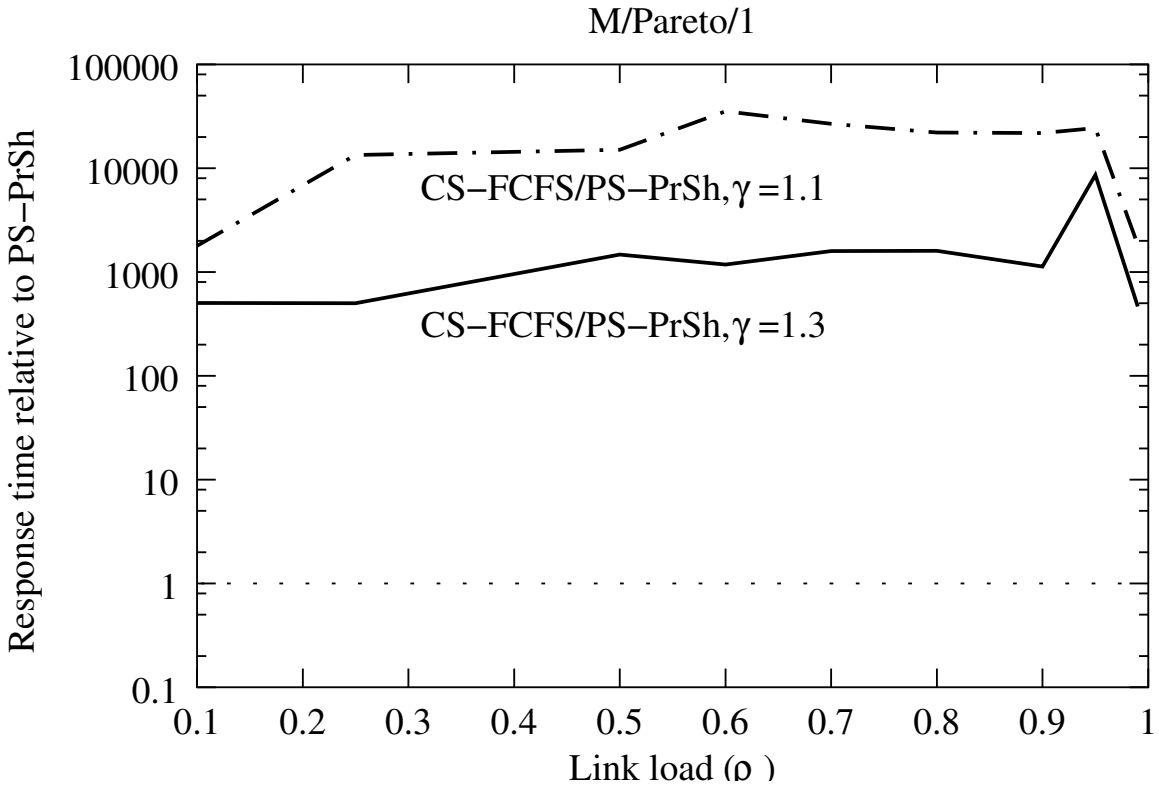

Figure 3.4: Relative average response time of CS-FCFS with respect to PS-PrSh for a single server. Arrivals are Poisson and flow sizes are bounded Pareto with parameter $\gamma = 1.1$ and $\gamma = 1.3$. Link loads are $\rho = 0.25$ and $\rho = 0.9$.

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.

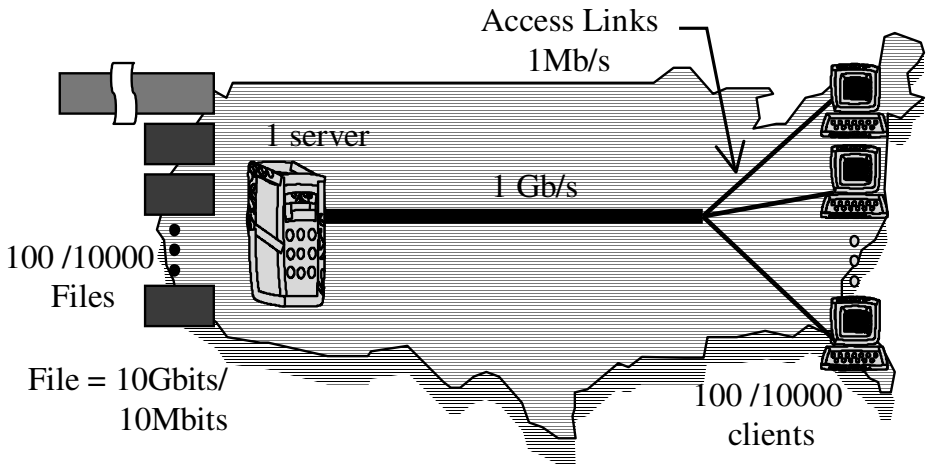

Figure 3.5: Network scenario for motivating examples 3.3.1 and 3.3.2. Access links of 1Mbit/s have been added, while the transcontinental link of 1 Gbit/s is kept the same. 1% of the files are long files of 10 Gbits, and the rest are only 10-Mbit long. In Example 3.4.1, there are only 100 clients, and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

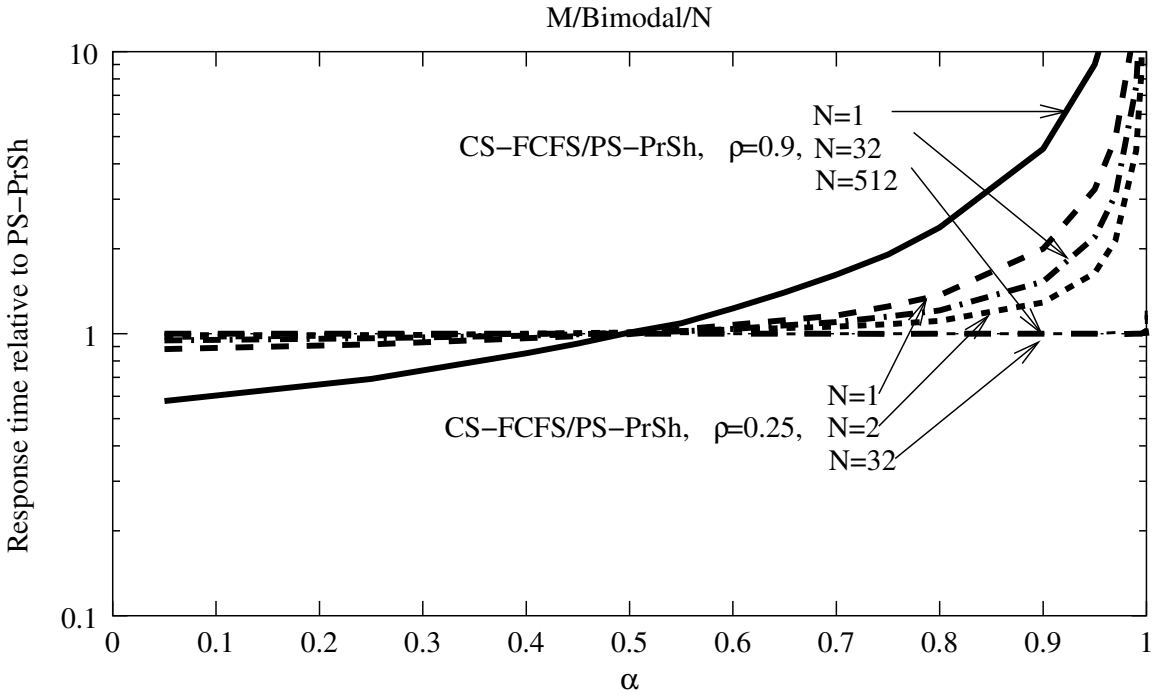

Figure 3.6: Relative average response time of CS-FCFS and CS-SJF with respect to PS-PrSh for an increasing core-to-access link-capacity ratio, N. Arrivals are Poisson and flow sizes are bimodal with parameter α. Link loads are $\rho = 0.25$ and $\rho = 0.9$. The value of N at which the curve flattens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

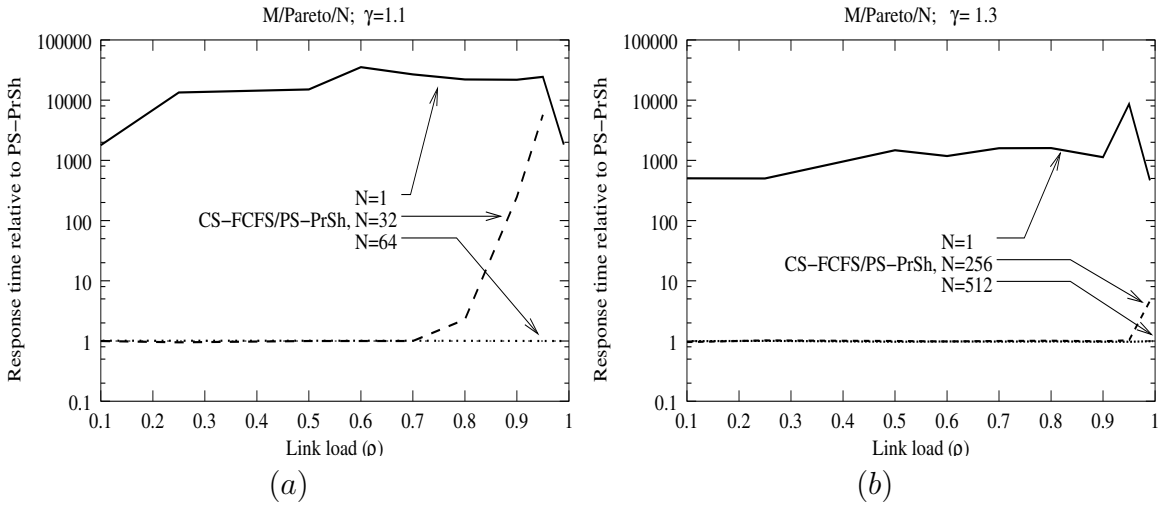

Figure 3.7: Relative average response time for CS-FCFS with respect to PS-PrSh for an increasing core-to-access link-capacity ratio, N. Arrivals are Poisson and flow sizes follow a bounded Pareto distribution with $\gamma = 1.1$ (a) and $\gamma = 1.3$ (b). The value of N at which the curves flatten is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

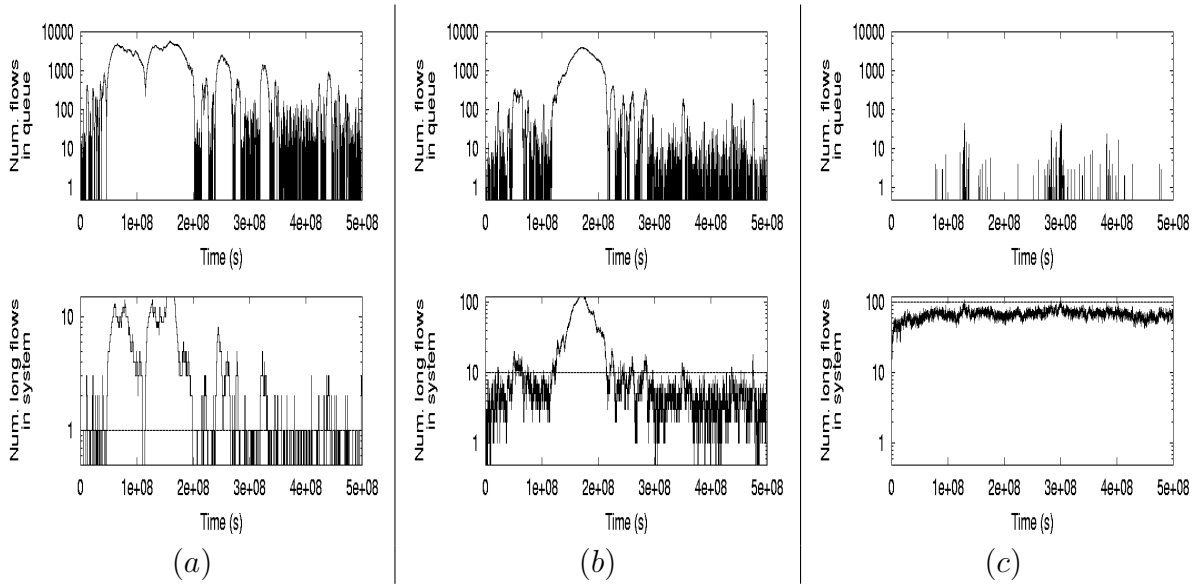

Figure 3.8: Time diagram of three M/Pareto/N/CS-FCFS systems. The top three graphs show the total number of jobs in the queue for (a) $N = 1$, (b) $N = 10$, and (c) $N = 100$. The bottom graphs only show the number of long jobs in the system (both in the queue and in the N servers). Whenever there are 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, $P(at\ least\ N\ arrivals\ during\ short\ flow) = 1 - F_{\hat{\lambda}}(N-1) \to 0$ as $N \to \infty$, 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, $P(at\ least\ N\ arrivals\ during\ short\ flow) \approx 0$.

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

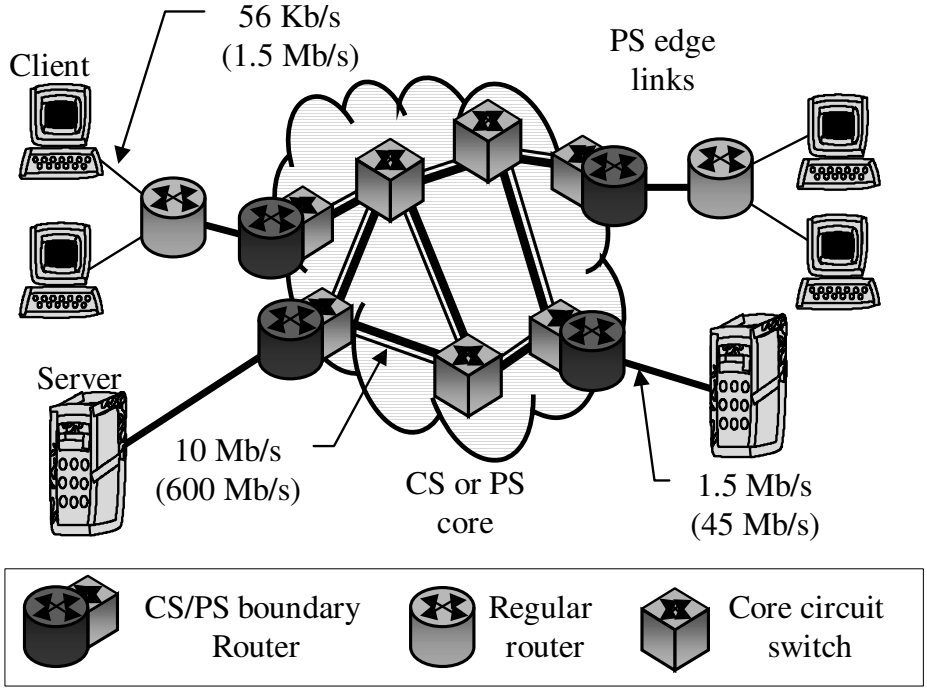

Figure 3.9: Topology used in the ns-2 simulation. Two separate access link speeds of 56 Kbit/s (phone modem) and 1.5 Mbit/s (DSL, cable modem) were considered for the clients. The capacities of server and core links were scaled accordingly.

[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).

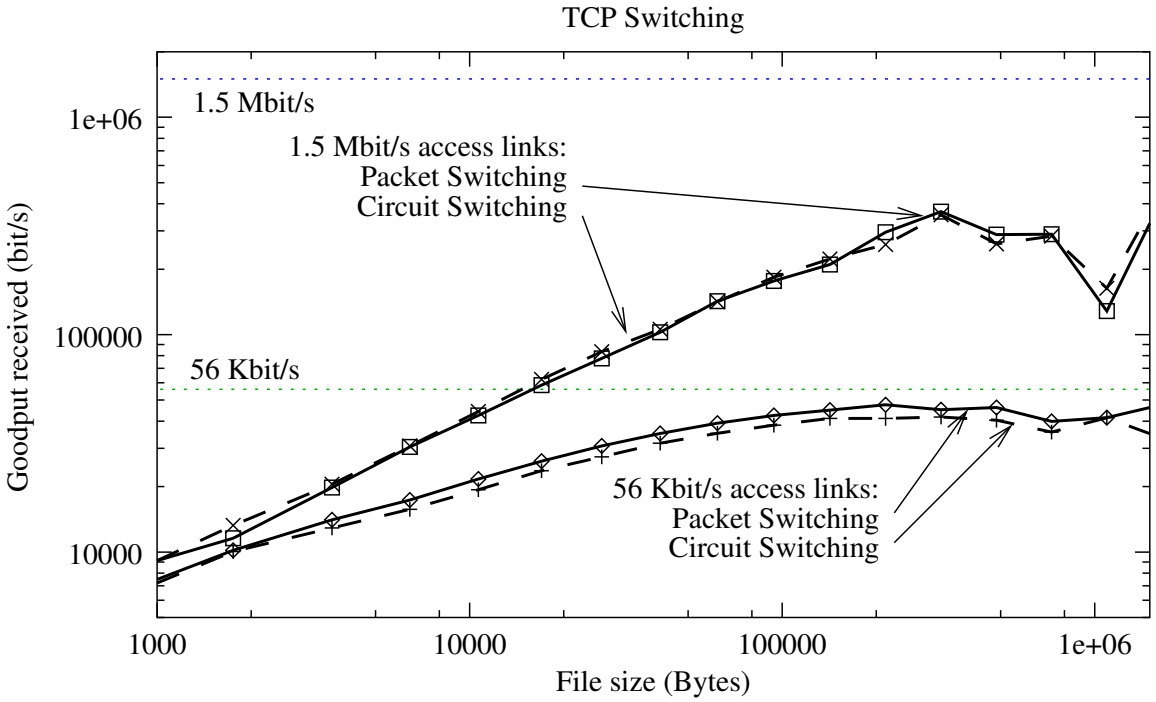

Figure 3.10: Average goodput as a function of the size of the transferred file for access links of 56 Kbit/s and 1.5 Mbit/s.

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.

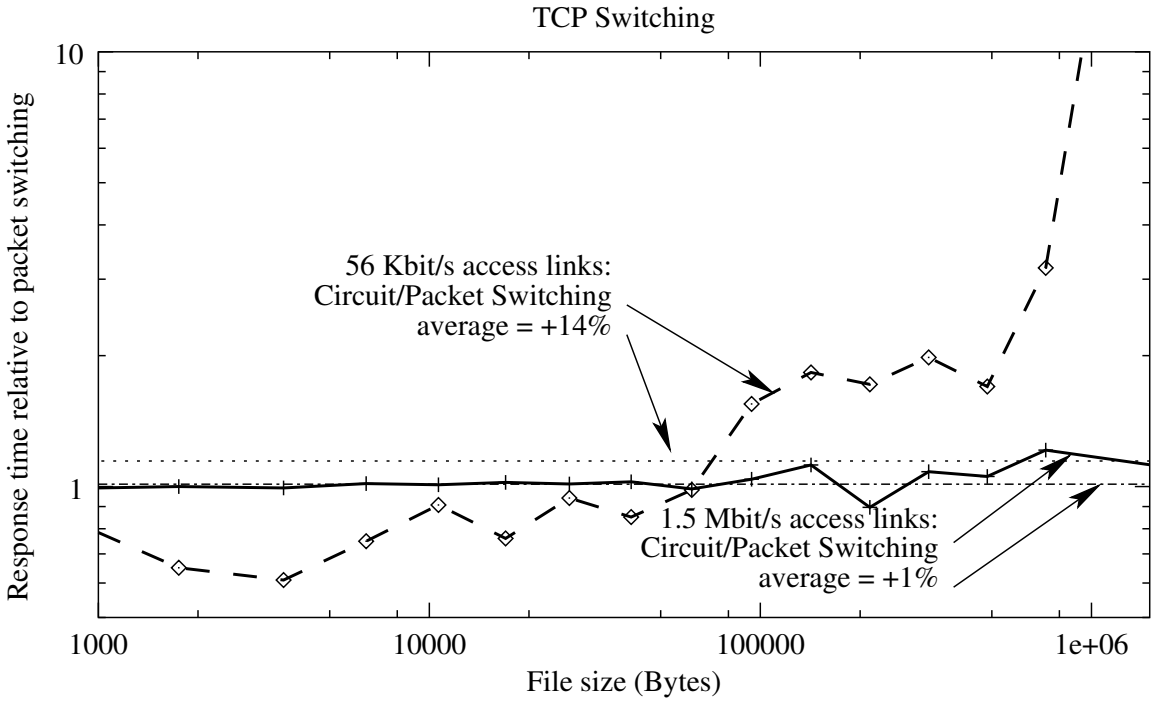

Figure 3.11: Average relative response time as a function of the size of the transferred file for access links of 56 Kbit/s and 1.5 Mbit/s. The average response time of TCP Switching over all file transfers is 14% greater than that of packet switching for 56 Kbit/s access links, and only 1% greater for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

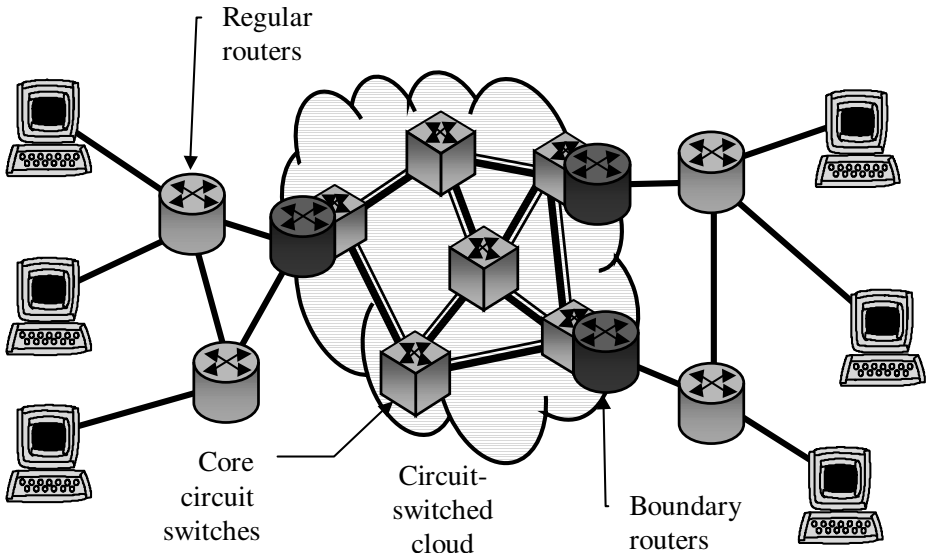

Figure 3.12: Hybrid network architecture using circuit switching in the core and packet switching in the edges that is recommended in this thesis.

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.