

Exercise Session 2: Network Performance

COM-208: Computer Networks

Before you start

The goal of this exercise session is to **understand the meaning of throughput and delay**, as **well as the nature of different delay components**. Exercises consist of two types: paper-and-pencil + lab exercises.

Transmission rate and throughput

The **transmission rate** of a link is the **rate at which we can push bits into the link**, and we measure it in “bits per second” (bps), “kilobits per second” (kbps), “megabits per second” (mbps), and so on. Note that 1kbps = 1000bps (not 1024bps), 1mbps = 1 million bps, etc.

The **throughput** between two end-systems is **the rate at which destination receives data**, and we also measure it in “bits per second” (bps), “kilobits per second” (kbps), “megabits per second” (mbps), and so on.

The **maximum throughput** that can be achieved between two end-systems is the **transmission rate of the bottleneck link** between them; i.e, **the link with the slowest transmission rate**.

The different components of delay

A packet experiences different kinds of delay:

- The **transmission delay** experienced by a **packet on a link** is the amount of time it takes to push the bits of the packet onto the link, and it is equal to the packet length divided by the link transmission rate.
- The **propagation delay of a link** is the amount of time it takes for one bit to go from one end of the link to the other, and it is equal to the link length divided by the link propagation speed.

- The **queuing delay** experienced by a packet at a switch is the **amount of time the packet sits inside a queue at the switch**, waiting for other packets to be processed and transmitted. Queuing delay depends on the other traffic that traverses the same links and shares the same queues as the packet.
- The **processing delay** experienced by a packet at a switch is the amount of **time it takes for the switch to process the packet** (after it removes it from the queue and before starting to transmit it). Processing delay depends on the switch's processing capabilities and is typically independent of packet size, at least in most of the scenarios we will discuss in this class.

Delay estimation over a single link

Two end-systems, A and B , are connected by a single link of transmission rate R , length l , and propagation speed c . $R = 100\text{mbps}$, $l = 200\text{km}$, $c = 2 \cdot 10^8\text{m/s}$.

- What is the propagation delay of the link?

A sends two back-to-back packets to B , the first one of size $L_1 = 1\text{kb}$, the second one of size $L_2 = 10\text{kb}$.

- What is the transmission delay experienced by each packet?
- What is the total transfer time, i.e. the time that elapses from the moment A starts transmitting the first bit of the first packet until the moment B receives the last bit of the second packet?
- What is the packet inter-arrival time at B , i.e., the amount of time that elapses from the moment the last bit of the first packet arrives until the moment the last bit of the second packet arrives?

Circuit switching

Let's examine how transfer time changes as we start introducing packet switches between source and destination. Suppose end-systems A and B are connected by not one, but N links and $N - 1$ packet switches. All the links have the same properties as the link in the previous problem.

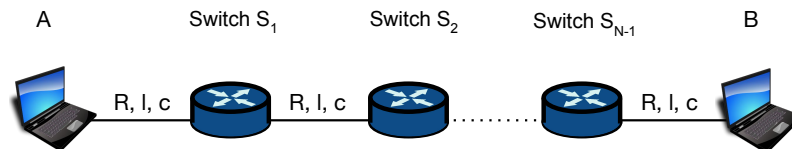


Figure 2: Two end-systems connected through multiple links and packet switches.

Suppose all these packet switches perform *circuit switching*, as in a traditional telephone network: before A starts “talking” to B , a physical circuit is established between them; this enables a packet switch to start transmitting each bit as soon as it receives it. A simple way to think of this scenario is that the packet switches “disappear”: if every switch transmits each bit as soon as it receives it, then it is, in a way, as if the switch becomes part of the link, as if A is directly connected to B .

A sends two back-to-back packets to B , each of size $L = 1\text{kb}$. Assume zero processing delay.

- What is the total transfer time when N is equal to 2, then 3 links?
- What is the total transfer time as a function of an arbitrary number of links N ?

Estimating total delay using timing diagrams

This is a good moment to introduce what we call **timing diagrams**: pictures that represent the various delays that a packet experiences as it travels through a network path. We have started such a diagram right below (Figure 5):

- There is a continuous vertical line for each end-system and packet switch; these lines represent time axes.
- A dashed horizontal line, that crosses the time axes, represents a point in time. For example, on the timing diagram below, the first dashed horizontal line represents $t = 0$, the second one represents $t = d_{prop}$, and so on.
- A continuous line that connects two time axes represents a bit as it travels between the corresponding devices; the beginning of this line represents the moment when the bit was transmitted, while the end of the line represents the moment when the bit arrived at the corresponding device. For example, on the timing diagram below, the first continuous line between the A and Switch1 time axes represents the first bit of the first packet as it travels from A to Switch1; the second continuous line between the same time axes represents the last bit of the first packet.

Using these diagrams, we can easily identify the various delay components that a packet experiences on each link and at each switch. We have marked these for the first packet sent from A to B , when $N = 2$ links. If you add the second packet, you will have solved the problem for $N = 2$. You can draw a new diagram, with 4 time axes, to solve the problem for $N = 3$. You cannot draw a diagram for an arbitrary number of links N , but 2 and 3 links are usually enough to give you the right intuition.

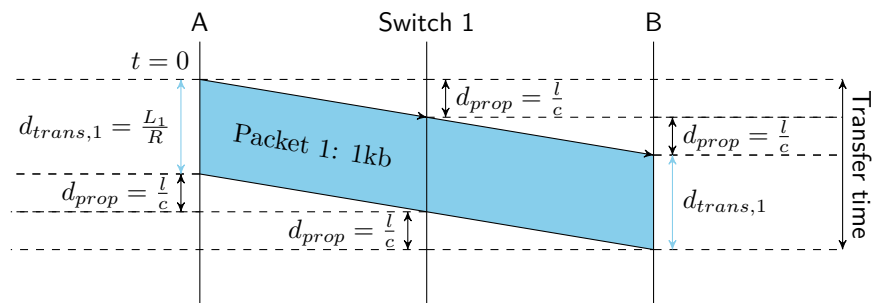


Figure 5: Timing diagram for the transmission of a single packet over two links connected by a packet switch performing circuit switching.

Store and forward

But, Internet packet switches do not typically perform circuit switching; many of them perform store-and-forward: when a switch receives a bit, it must store that bit in a buffer until the last bit of the corresponding packet has arrived; only then can the switch process the packet and start transmitting it over the next link.

Consider the same scenario as in the previous problem, except now, all the packet switches perform store-and-forward. Assume that the processing delay at each switch is 0. Still, the fact that every switch must buffer every bit until it has received the corresponding packet means that there will be some extra delay relative to the circuit switching scenario. (Picture a car rally where every time the first car arrives at an intermediate stop, it must wait for all the cars behind it to arrive before it can start again. Now replace the cars with bits of a packet.)

A sends two back-to-back packets to B , each of size $L = 1\text{kb}$.

- What is the total transfer time when N is equal to 2, then 3 links?
- What is the total transfer time as a function of an arbitrary number of links N ?
- How do your answers change when the processing delay at each switch is $d_{proc} = 1\mu\text{s}$ per packet? Assume that the switch must finish processing and transmitting a packet before it starts processing the next packet.
- Does the second packet experience any queuing delay at the first packet switch? To answer, focus on the first packet switch and compute the difference between: (a) when the switch transmits the last bit of the first packet and (b) when the switch receives the last bit of the second packet. If this is greater than 0, it means that the second packet must wait for the switch to finish transmitting the first packet, i.e., it experiences queuing delay.
- Does the second packet experience any queuing delay at any other packet switch?

The easiest way to solve this problem—and most delay problems—is through timing diagrams. We have started one right below that represents the first packet sent from A to B when $N = 2$ links. Notice what is happening at the switch: because it is a store-and-forward switch, it cannot transmit the first bit of the first packet as soon as it receives it, it must wait for the last bit of the first packet to arrive; only then it can process and transmit the first packet.

To solve the problem, add the second packet, then draw another timing diagram for $N = 3$.

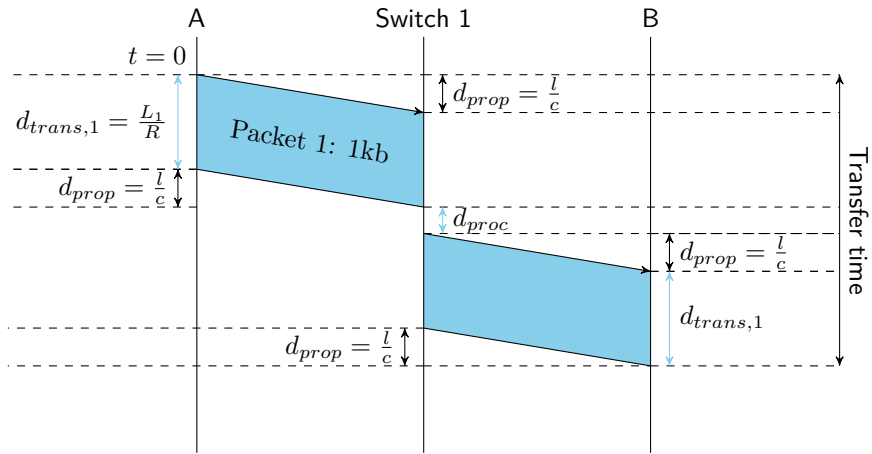


Figure 8: Timing diagram for a transmission over two links connected by a store-and-forward switch.

Now with lots of packets!

When two end-systems communicate over the Internet, they often exchange $P > 2$ packets. Intuitively, that should increase the total transfer time, but by how much? A factor of P ? Does it depend on N ?

Consider the same scenario as in the previous problem: A and B are connected through $N - 1$ store-and-forward switches, each introducing processing delay 0 . Except now, A sends P back-to-back packets to B , all with the same length L . Assume zero propagation delays.

- What is the total transfer time?

Queuing and bottlenecks

You may have noticed the artificial uniformity in the previous problems: all packets were of the same length, all links were of the same transmission rate... Reality is not like this, of course. Packets have different lengths, and links have different transmission rates, and both of these things lead to queuing. Even if A and B are alone in the Internet—there is no other traffic to interfere with theirs—when A sends back-to-back packets to B , a packet may have to wait at a packet switch for the previous packet to be transmitted, either because the previous packet is longer, or because the next link is slower.

End-systems A and B are connected through two links with one store-and-forward packet switch in the middle, introducing processing delay 0. Both links have propagation delay d_{prop} . The first link has transmission rate R_1 , while the second one has transmission rate R_2 .

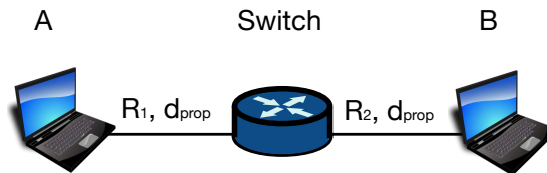


Figure 9: Two end-systems connected through two links.

Assume that $R_1 = R_2$. A sends two back-to-back packets to B , the first one of size L_1 , the second one of size $L_2 \neq L_1$.

- In which scenario does the second packet experience queuing delay at the switch? How much?
- What is the total transfer time (in this scenario)?

[This one is if you want to challenge yourself.] A and B are connected through N links and $N - 1$ store-and-forward packet switches, each introducing processing delay 0, all links have propagation delay d_{prop} and transmission rate R , and A sends P back-to-back packets to B , of different lengths, L_1, L_2, \dots, L_P . Assume infinite packet-switch buffers (no packet drops).

- What is the total transfer time?

Now let's go back to the simpler scenario where A and B are connected through two links with one store-and-forward packet switch in the middle, introducing processing delay 0. Assume that $R_2 \neq R_1$. A sends two back-to-back packets to B , each of size

L. Is it possible that the second packet experiences queuing delay at the packet switch? Let's consider all the possibilities:

- Suppose $R_1 < R_2$, i.e., the first link has a lower transmission rate than the second one, in other words, the first link is the bottleneck. (Picture a narrow local street, followed by a wide highway. Would you expect cars to queue up at the highway entrance?)
 - How much queuing delay does the second packet experience at the packet switch?
 - What is the packet inter-arrival time at *B*?
- Now suppose $R_1 > R_2$, i.e., the second link is the bottleneck. (Picture a wide highway, followed by a narrow local street. Would you expect cars to queue up at the highway exit?)
 - How much queuing delay does the second packet experience at the packet switch?
 - Suppose that *A* sends the second packet *T* seconds after sending the first one. How large must *T* be to ensure no queuing at the switch?

Lab: Measure delay for different targets

Tools for the lab exercises

For the lab exercises, you will use “ping” (which you discovered in the previous exercise session) to experimentally estimate throughput and delay between your computer and various targets around the world. You will see that there are quantities that we cannot directly measure, yet we can reason about them if we understand what they are.

Download from Moodle `runping.sh` and `plot.sh`. If they are not already executable, make them, e.g., by running `chmod u+x runping.sh` and `chmod u+x plot.sh`. If you are doing the lab on your own computer, make sure you have the `gnuplot` application installed (for Ubuntu `sudo apt install gnuplot`).

`runping.sh` is a shell script that takes as an argument a target (DNS name or IP address) and pings the target many times with different packet sizes. For example, `./runping.sh www.epfl.ch` executes:

```
ping -s 22 -c 50 -i 1 www.epfl.ch > www.epfl.ch-p50
...
ping -s 1472 -c 50 -i 1 www.epfl.ch > www.epfl.ch-p1500
```

The first line pings `www.epfl.ch` 50 times, once per second, using 50-byte packets, and writes the output to a file called `www.epfl.ch-p50`. The last line pings `www.epfl.ch` 50 times, once per second, using 1500-byte packets, and writes the output to a file called `www.epfl.ch-p1500`.

`plot.sh` takes as an argument a list of file names (that were produced by ‘`runping.sh`’) and produces three new files:

1. `destination_delay.png` shows delay as a function of time, for different packet sizes (different colors correspond to different packet sizes).
2. `destination_scatter.png` shows delay as a function of packet size as a scatter plot.
3. `destination_avg.txt` contains the average (2nd column) and minimum (3rd column) delay values for each packet size (1st column). For example, `./plot.sh www.epfl.ch-p*` produces the above three files for the target `www.epfl.ch`.

Measure and process

Run `runping.sh` with target `www.epfl.ch`, then with target `www.yahoo.com`, and then with target `www.gov.kg`. Given that you are sending one ping every second, these will take several seconds to complete.

When they finish, process the results by running:

```
./plot.sh www.epfl.ch-p*
./plot.sh www.yahoo.com-p*
./plot.sh www.gov.kg-p*
```

Think about delay

Look at the figures you have produced for each target and answer the following questions:

- Does delay change significantly over time? Whether you answer yes or no, explain why you think this is happening. E.g., if you answered that it does not change, consider each delay component and explain why it doesn't change over time. If you answered that it does change, which delay component(s) are the ones that change?

Note: If delay has very large spikes, this may be because you are on WiFi. Try connecting to a VDI VM to do the lab.

- Do you expect delay to change with the size of the ping packets? Look at the latency vs ping size graph that you produced. Does latency change? How do you explain the result?
- Suppose that, for a given target, there is no clear dependence pattern between delay and packet size, i.e., packet size does not affect delay significantly. What does that say about the path from your computer to that target?

Estimate delay components

Now we will explore the kind of "rough" estimations that network engineers often need to do.

Ping tells you the overall delay experienced by different packets of different sizes, but it does not (directly) tell you anything about individual delay components, like transmission delay or queuing delay. This is not surprising, because ping operates at the end-systems, it has no access inside the network to see, e.g., how long it takes to transmit each packet on each link, or how long each packet has to wait in a queue at each switch.

Still, if we know the overall delay experienced by *many* different packets and of *many* different sizes, then we *can* infer information about individual delay components.

Study the content of the file `www.yahoo.com_avg.txt`. Estimate the following components of the delay between your computer and `www.yahoo.com`:

- The propagation+processing delay.
- The average queuing delay.

You don't have enough information to do any precise estimation. Do the best you can given the information you have.

Now study the content of the file `www.gov.kg_avg.txt` and estimate the same delay components between your computer and `www.gov.kg`. Observe the differences in the results relative to `www.yahoo.com`. What might explain these differences?

Parallel paths and throughput

Usually, multiple network paths exist between end-systems. Intuitively, adding multiple paths between two end-systems should improve network performance, but which metric exactly? Clearly, adding paths (of the same type) cannot reduce the propagation delay between two end-systems, nor the transmission delay experienced by each single packet. What it *can* change is the overall rate at which A can send data to B : the throughput.

End-systems A and B are connected over M parallel network paths; in this context, “parallel” means that they do not share any links between them. Each path $k \in [1, 2, \dots, M]$ consists of N links with transmission rates $R_1^k, R_2^k, \dots, R_N^k$.

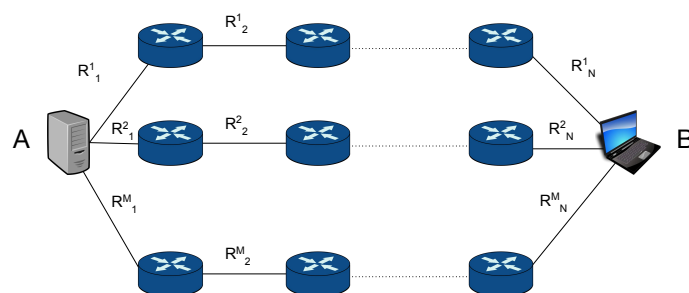


Figure 19: A network topology with multiple parallel paths.

- What is the maximum possible throughput from A to B , when they can use only one path at a time?
- How does your answer change when A and B can use all M paths simultaneously?