Thinking about networks

Ankit Singla
Learning objectives

• What do we want from the network?
• How do we achieve these goals?

• Readings for next week
  • Reading 1: Rethinking the design of the Internet
  • Reading 2: Why the Internet only just works
What do we want from the network?

- Performance: latency? bandwidth?
- Reliability / availability / security?
- Flexibility, manageability?
- …
Network performance
Performance = bandwidth?
To get many megabits-per-second ...
To get many megabits-per-second ...

4400 km
80 Km / hour
1 TB USB stick
= 40 Mbps

... but 55 hours
Performance = latency?
LATENCY LAGS BANDWIDTH

Recognizing the chronic imbalance between bandwidth and latency, and how to cope with it.
Bandwidth and latency

Transfer time

Network round-trip time

Large object

Small object
Bandwidth and latency

Transfer time

Network round-trip time

Large object

Small object

10 GBps, 1 millisecond
- 1 B?
- 100 KB?
- 10 GB?

10 GBps, 100 millisecond
- 1 B?
- 100 KB?
- 10 GB?
Bandwidth and latency

Transfer time

Large object

Small object

Network round-trip time

10 GBps, 1 millisecond
- 1 B?
- 100 KB?

Note: you are expected to know how TCP works, e.g.: what is AIMD? bandwidth-delay product?

With TCP?

100 KB $\approx 1500$ B $\times (0 + 2 + 4 + 8 + 16 + 32 + 5)$

$\approx 7$ milliseconds in the best case;
0.0143 GBps
Why Flow-Completion Time is the Right Metric for Congestion Control

Nandita Dukkipati  
Computer Systems Laboratory  
Stanford University  
Stanford, CA 94305-9025  
nanditad@stanford.edu

Nick McKeown  
Computer Systems Laboratory  
Stanford University  
Stanford, CA 94305-9025  
nickm@stanford.edu

Plot inspired by Patterson's illustration of how latency lags bandwidth in computer systems [2].

Worse still, in a real network flows come and go unpredictably, even if their arrival times and durations are known [3].

Users typically want their flows to complete as quickly as possible. This makes Flow Completion Time (FCT) an important - arguably the most important - performance metric.

Proposition 1: For typical downloads, while propagation delay will always increase in bandwidth, FCT has reduced by only 50%.

Proposition 2: They care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Why is this? The intuition is that as network bandwidth increases, high throughput and efficiency will reduce FCT.

But we believe - and it is the main argument of this paper - that this is not the case. Algorithms have been developed to improve FCT, even if they are heuristic. It is intractable to minimize FCT.

Proposition 3: Congestion control mechanisms which make flows last multiple round trip times (RTT) are focussed on efficiency - maximizing link throughput, utilization - almost entirely on maximizing link throughput, utilization - an important - arguably the most important - performance metric.

Proposition 4: They represent a tiny fraction of traffic. Certainly, fairness is not necessarily in the user's best interest. Certainly, control mechanisms which make flows last multiple RTTs are not focussed on fairness, which matter more to the operator than the user.

Proposition 5: Users just want their flow to finish quickly. They care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 6: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 7: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 8: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 9: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Why is this? The intuition is that as network bandwidth increases, high throughput and efficiency will reduce FCT.

But we believe - and it is the main argument of this paper - that this is not the case. Algorithms have been developed to improve FCT, even if they are heuristic. It is intractable to minimize FCT.

Proposition 10: Congestion control mechanisms which make flows last multiple RTT are focussed on efficiency - maximizing link throughput, utilization - almost entirely on maximizing link throughput, utilization - an important - arguably the most important - performance metric.

Proposition 11: They represent a tiny fraction of traffic. Certainly, fairness is not necessarily in the user's best interest. Certainly, control mechanisms which make flows last multiple RTTs are not focussed on fairness, which matter more to the operator than the user.

Proposition 12: Users just want their flow to finish quickly. They care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 13: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 14: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 15: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Why is this? The intuition is that as network bandwidth increases, high throughput and efficiency will reduce FCT.

But we believe - and it is the main argument of this paper - that this is not the case. Algorithms have been developed to improve FCT, even if they are heuristic. It is intractable to minimize FCT.

Proposition 16: Congestion control mechanisms which make flows last multiple RTT are focussed on efficiency - maximizing link throughput, utilization - almost entirely on maximizing link throughput, utilization - an important - arguably the most important - performance metric.

Proposition 17: They represent a tiny fraction of traffic. Certainly, fairness is not necessarily in the user's best interest. Certainly, control mechanisms which make flows last multiple RTTs are not focussed on fairness, which matter more to the operator than the user.

Proposition 18: Users just want their flow to finish quickly. They care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 19: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 20: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.

Proposition 21: Users care less about the throughput of the network, how efficient network utilization is, or fairness, which matter more to the operator than the user; in fact, high throughput or excellent network utilization has the potential to increase FCT.
Beyond flow completion time?

- How long does https://www.google.ch/?q=cool+stuff take?
- What’s the best video quality I can watch, without the “buffering”?
- How long does my Hadoop job take?
- ...

Also, we want consistent, predictable performance!
But what about fairness?!

Suppose a network is flow fair. How useful is that?

“Both the thing being allocated (rate) and what it is allocated among (flows) are completely daft—both unrealistic and impractical.”
Food for thought …

How to …

• … translate micro-benchmarks to app-level metrics?
• … make service providers accountable?
• … improve the performance of systems?
• … approach fairness?
Reliability
Three particular considerations with reliability …

- The end-to-end argument
- The fate-sharing principle
- Packet vs. circuit switching
Three particular considerations with reliability ...

- The end-to-end argument
- The fate-sharing principle
- Packet vs. circuit switching
TCP/IP vs. Everything/IP

Initially, TCP/IP was a monolith. Why did we split?
What if no reliable transport is provided?

Every application that needs reliability has to engineer it from scratch

- Programmer burden
- Much higher likelihood of bugs
- Wasteful effort
What if the network layer tried to provide reliable delivery?

Reliable (or unreliable) transport
...built on...

Best-effort global packet delivery
Reliable

#1: My voice call wants speedy delivery, even if it’s lossy …
What if the network layer tried to provide reliable delivery?

Reliable (or unreliable) transport
...built on...

Best-effort global packet delivery
Reliable

#2: Can the network even achieve this at all?
Example: reliably transfer file from host A to B

Check reliability at every step (involving network layer)

Problem: Bugs, failures are a truth of life
Example: reliably transfer file from host A to B

It’s not *reliable transfer*, if A-B *depend on* network that can fail
Example: reliably transfer file from host A to B

Allow unreliable steps (network layer is best-effort). B checks correctness. On failure, B tells A to retry. 

Can still fail, but only if A / B themselves fail.

Depends only on what end-points themselves control
Three particular considerations with reliability ...

- The end-to-end argument
- The fate-sharing principle
- Packet vs. circuit switching
The fate-sharing principle

When storing state in a distributed system, co-locate it with entities that rely on that state.

State is lost only if those entities fail; then it doesn’t matter.

Example: network connection state at end hosts.
Three particular considerations with reliability ...

- The end-to-end argument
- The fate-sharing principle
- Packet vs. circuit switching
Circuit switching relies on the Resource Reservation Protocol
Pros and cons of circuit switching

Predictable performance

Wasteful, if traffic is bursty or short

Large latency for small messages, as they wait for circuits

Require new circuit setup upon failure
Pros and cons of packet switching

Efficient use of resources

Automatic, in-network rerouting on failures

Unpredictable performance

Requires buffer management and congestion control
Packet switching beats Circuit switching with respect to resilience and efficiency.
Rule of thumb for circuits v. packets

Large Peak / Average ⇒ Packets
>100 for Web
What about Internet video?

Small Peak / Average ⇒ Circuits
~3 for voice

Video share of global Internet traffic
~3 for voice
>100 for Web
Chunked video transfer today

Packets / sec

Wireshark I/O graphs
Course topics
Big, ongoing / future changes to the Internet

**Algorithms and protocols**
- Congestion control
- Routing protocols
- Video streaming

**Network design & management**
- Satellite networks
- Software-defined networking
- Data centers
Weekly reading guide
More on the end-end argument

This “rethink” was in 2001

What has changed?
Why the Internet only just works

M Handley

1. Introduction

The Internet only just works. I want to make it clear though, right from the start, that this is not a forecast of imminent doom and disaster. My reasons for making this assertion are twofold. Firstly, I believe that this has historically been the natural state of the Internet and it is likely to remain so in future. Unless this is understood, then it is hard to understand which problems are really cause for concern, and which we can safely ignore or put off solving till some later date. Secondly, I want to discuss some problems that should be cause for concern, at least in the medium term.

2. The natural state of affairs

If we look back at the history of the Internet, the story is one of constant change. Indeed the phrase 'Internet time' is often used to express just how fast things change. But if we look at the core protocols that comprise the Internet at the lower levels of the stack, change has been comparatively slow and carefully considered.

2.1 1970-1993 — a history of change

The first large-scale packet switching network was the ARPAnet, which was used to come to grips with the main architectural issues that would go on to be the basis of the Internet. The basic protocol that underlay the ARPAnet was NCP [1], which combined addressing and transport into a single protocol. Many of the higher-level protocols that would go on to become common on the Internet were first deployed on the ARPAnet. The most obvious are remote log-in, e-mail, and file transfer, but there were also ARPAnet experiments with packet voice, which predate common usage of voice-over-IP by over twenty years.

The ARPAnet was very successful, but it was also clear that flexibility should be of prime importance in the design of a general-purpose successor [2], and as a result reliability was separated from addressing and packet transfer in the design of the Internet protocol suite, with IP being separated from TCP. The switchover to TCP/IP culminated in a flag-day on 1 January 1983 when all remaining ARPAnet nodes switched. There were approximately four hundred nodes; this was probably the last time such a flag-day was possible, and every change since then has needed to be incrementally deployable.

Changing a large network is very difficult. It is much easier to deploy a novel new protocol that fills a void than it is to replace an existing protocol that more or less works. Change is, however, possible when the motivation is sufficient. In 1982 the domain name system (DNS) was deployed, replacing the original hosts.txt file [3] for naming Internet systems. This was a clear response to a scaling problem, but the necessity for change was obvious, and the DNS not only solved the basic issue of distributing files of host names, but also allowed the change to decentralised administration of the namespace. Decentralised administration is one of the basic enablers of the rapid...