Performance Measures and Application Performance

Kyunghan Lee
Networked Computing Lab (NXC Lab)
Department of Electrical and Computer Engineering
Seoul National University
https://nxc.snu.ac.kr
kyunghanlee@snu.ac.kr

Adapted from Prof. P. Stuedi's slides

What is network performance?

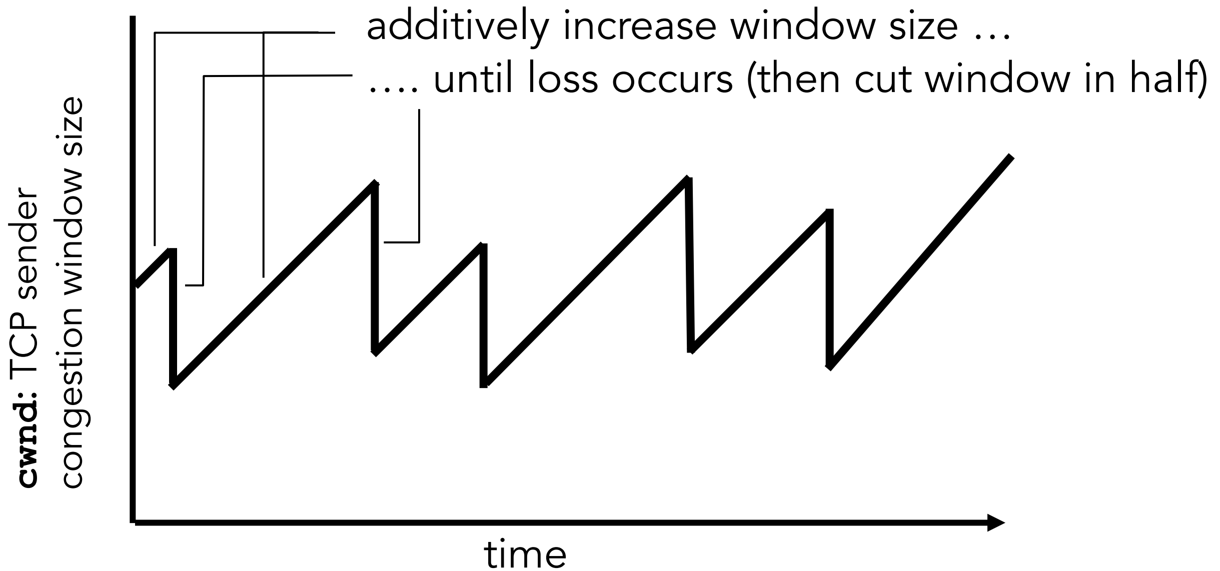
- Two fundamental measures:
  - 1. Bandwidth
    - Roughly: bits transferred in unit time
    - Not quite the Electrical Engineering definition
    - Also known as **Throughput**
  - 2. Latency
    - Time for 1 message to traverse the network
    - Half the **Round Trip Time** (RTT)
    - Also known as **Delay**

Kevin Fall and Steve McCanne, "You Don't Know Jack about Network Performance," ACM Queue, pp. 54-59, May 2005.

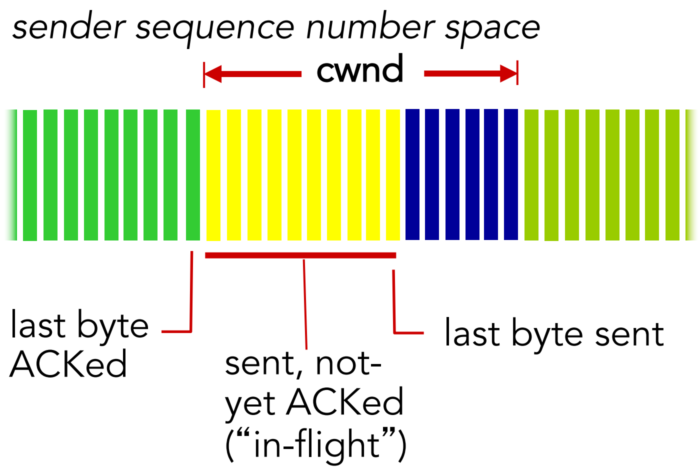
TCP congestion control: Additive Increase Multiplicative Decrease (AIMD)

- *Approach:* sender increases transmission rate (window size), probing for usable bandwidth, until loss occurs
  - *additive increase:* increase **cwnd** by 1 MSS (max segment size) every RTT until loss detected
  - *multiplicative decrease*: cut **cwnd** in half after loss

AIMD's saw tooth behavior: probing for bandwidth



TCP Congestion Control



TCP sending rate:

- *roughly*: send **cwnd** bytes, wait RTT for ACKs, then send more bytes

$$\textbf{Rate} \approx \frac{\textbf{cwnd}}{\text{RTT}} \text{ bytes/sec}$$

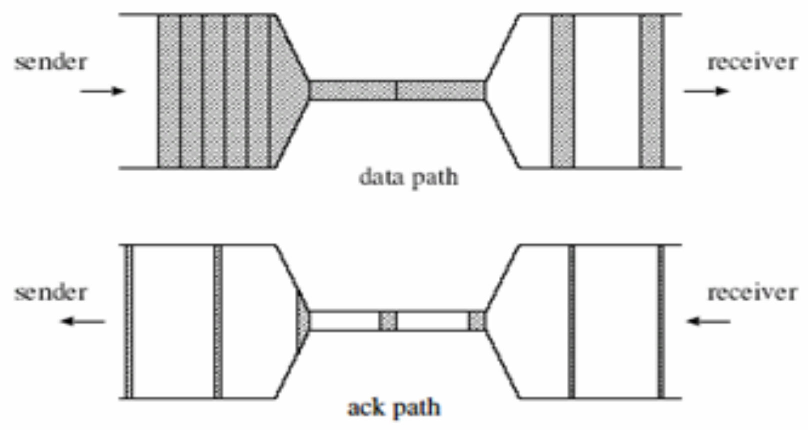
- sender limits transmission:

$$\textbf{LastByteSent-LastByteAcked} \leq \textbf{cwnd}$$

- **cwnd** is dynamic, function of perceived network congestion

Control Loop in TCP

- **`cwnd`** and ACK
  - "ACK clocking": the sender transmits a data packet upon an ACK reception

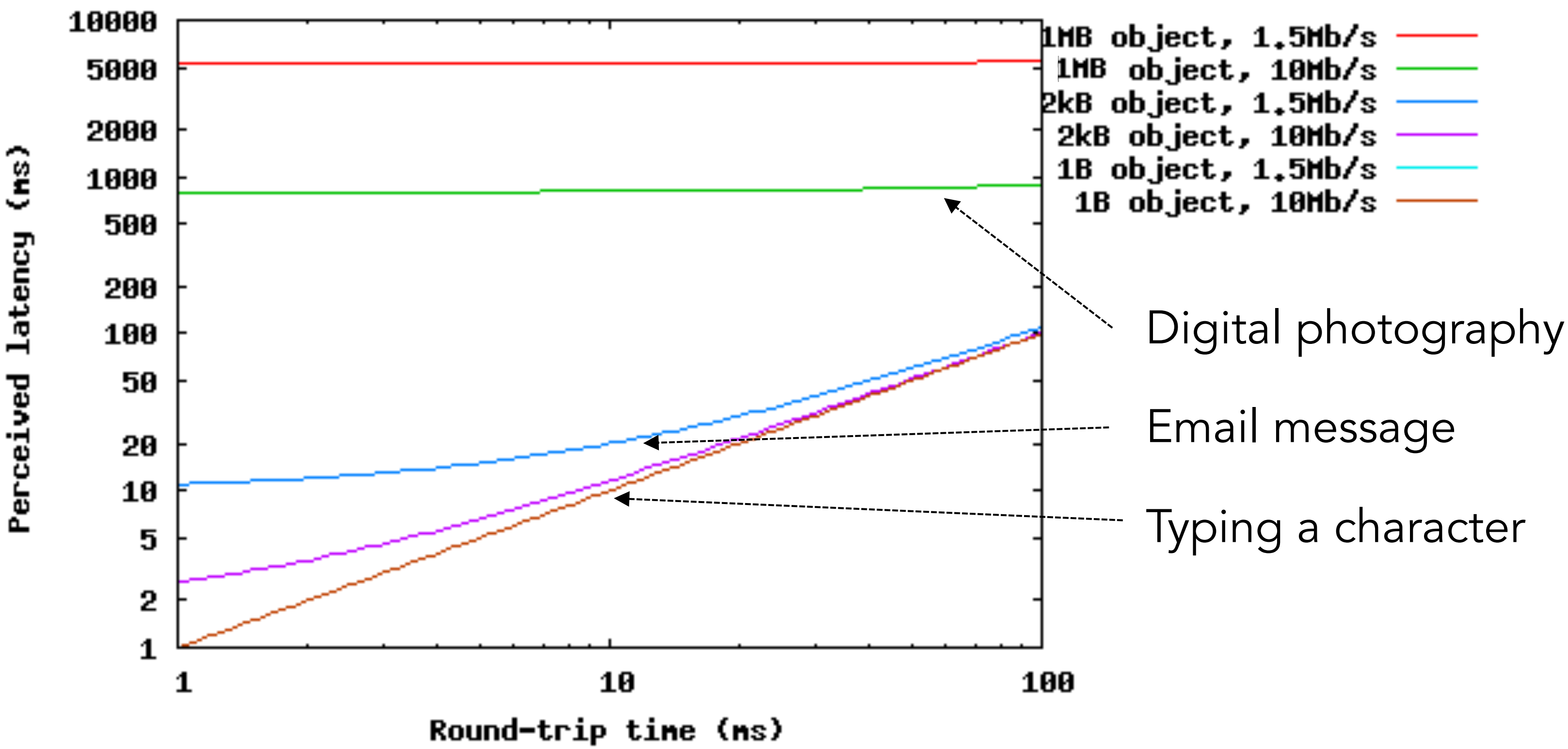


Source: http://www.soi.wide.ad.jp/class/20020032/

- Control the size of the sliding window
  - Suppose that receiver buffer size is sufficiently large (i.e., large **`rwnd`**)
  - Controlling the window size will determine the TX rate (➔ congestion control).

Time to transfer an object

(*Ignoring* queuing delay, processing, etc.)



Example: request 1MB file over a 1Gb/s link, with 200ms RTT

- What's the *throughput*?

Throughput = (Transfer size) / (Transfer time)

8Mb / 0.208s = ~38.5Mb/s ??

- What's the *transfer time*?

Request + first byte delay (→ RTT)

Transfer time = RTT + (Transfer size) / Bandwidth

0.2s + 8Mb/1Gbs = 0.208s

What's gone wrong here?

- File is too small?
- Round-trip time is too high?

- You can't reduce the latency (propagation delay).
  - Adding bandwidth won't really help.

Bandwidth-Delay Product (BDP)

- Example: Latency = 200ms, Bandwidth = 40Gb/s
  - ⇒ "channel memory" = 8Gb, or 1 gigabyte

- BDP: What the sender can send before receiver sees anything
  - Or *must* send to keep the network pipe full...

TCP Window Size

- Recall TCP keeps data around for transmission!
  - e.g., consider sliding window of w bytes
  - TCP moves w bytes every RTT (needs to wait for ACK)
    $\Rightarrow$ throughput = w / RTT
- What's the optimal window size w?
  - Need to keep the pipe completely full
    $\Rightarrow w =$ RTT × pipe bandwidth
  - Example: 10Gb/s, 200ms RTT
    - Gives: $w \sim$ 200MB per connection…
- Protocol limits:
  - TCP window size without scaling ≤ 64kB
  - TCP window size with RFC1323 scaling ≤ 1GB

What's gone wrong here?

- File is too small?
- Round-trip time is too high?

- You can't reduce the latency.
  - Adding bandwidth won't really help.

- **Lesson 1**: effectively using high bandwidth-delay product networks is hard.
- **Lesson 2**: as bandwidth increases, **latency** is more important for performance.

What are these high-BDP networks?

- Where is high bandwidth?
  - Datacenters! 100Gbps or More (Ethernet or InfiniBand)
  - 5G! Targeting 10Gbps as its peak data rate

- Where is high delay?
  - Between datacenters: 200ms not unusual
  - 3G or 4G networks: higher than 50ms, 500ms not unusual
  - Wireless networks: many reasons, as we will see

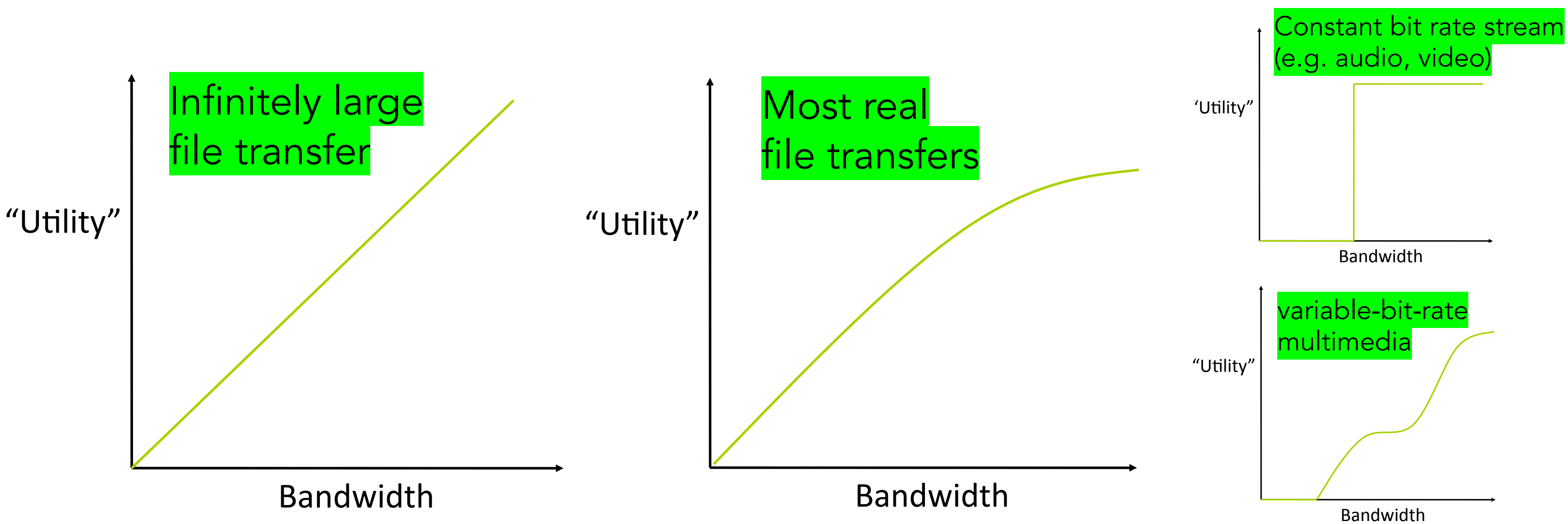
Protocol Design impacts Performance

- Protocols which require many RTTs don't work well in the wide area.
- Example: Opening a network folder in Windows 2000
  - About 80 request/response pairs on average
  - 200ms RTT (e.g. London-Redmond)
    ⇒ more than 16 seconds delay
- Upgrading your network will not help!
  - (and didn't…)

Application Requirements

- Some applications can't use all the network
- Example: constant-bit-rate 320kbs MP3 streaming

- **Utility function**: measure of how useful each network resource (bandwidth, latency, etc.) is to an application

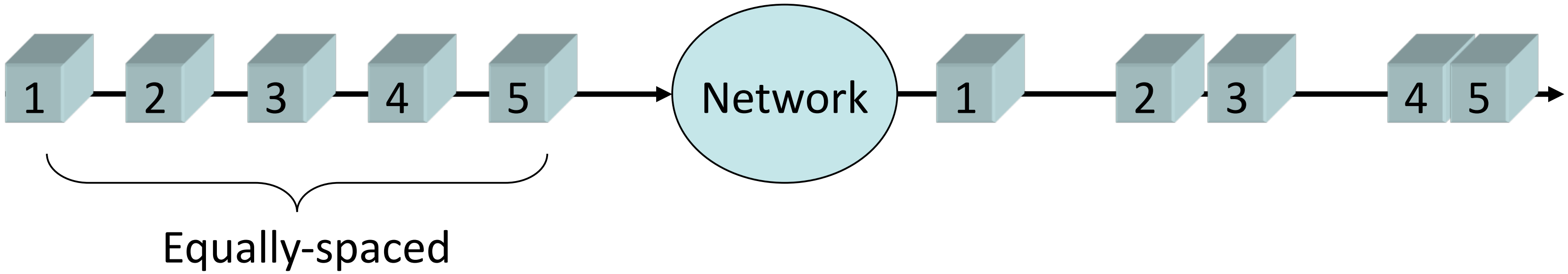


Application Requirements: Burstiness

- Many applications are "bursty"
  - Required network bandwidth varies over time
- Challenge: describe and analyze bursty sources
  - Token buckets, leaky buckets, queuing theory…

Application Requirements: Jitter

- *Variance* in end-to-end latency (or RTT)
- Example: voice (telephony)
- How long to wait to play a received packet?
  - Too long: hard to have a conversation (150ms limit)
  - Too short: some packets arrive too late (lose audio)



Application Requirements: Loss

- Networks do, sometimes, lose packets.
- Loss is complex:
  - Packet loss rate, Bit-error rate
- Losses and errors must be detected:
  - Codes, sequence numbers, checksums
- Handled by:
  - Error-correction
  - Retransmission

Application Requirements: Perspectives

Distance from server to user	Network latency	Typical packet loss	Throughput (quality)	4GB DVD download time
Local: <160 km	1.6ms	0.6%	44Mb/s (HDTV)	12 min.
Regional: 800-1600km	16ms	0.7%	4Mbs (not quite DVD)	2.2 hrs.
Cross-continent ~4800km	48ms	1.0%	1Mbs (not quite TV)	8.2 hrs.
Multi-continent ~10000km	96ms	1.4%	0.4Mbs (poor)	20 hrs.

Source: Tom Leighton, "Improving Performance on the Internet," Communications of the ACM, February 2009.

How to Read a Paper

S. Keshav

You May Think You Already Know How To Read, But…

How to Read a Paper

S. Keshav
David R. Cheriton School of Computer Science, University of Waterloo
Waterloo, ON, Canada
keshav@uwaterloo.ca

ABSTRACT

Researchers spend a great deal of time reading research papers. However, this skill is rarely taught, leading to much wasted effort. This article outlines a practical and efficient *three-pass method* for reading research papers. I also describe how to use this method to do a literature survey.

Categories and Subject Descriptors: A.1 [Introductory and Survey]

General Terms: Documentation.

Keywords: Paper, Reading, Hints.

1. INTRODUCTION

Researchers must read papers for several reasons: to review them for a conference or a class, to keep current in their field, or for a literature survey of a new field. A typical researcher will likely spend hundreds of hours every year reading papers.

4. Glance over the references, mentally ticking off the ones you've already read

At the end of the first pass, you should be able to answer the *five Cs*:

1. *Category*: What type of paper is this? A measurement paper? An analysis of an existing system? A description of a research prototype?
2. *Context*: Which other papers is it related to? Which theoretical bases were used to analyze the problem?
3. *Correctness*: Do the assumptions appear to be valid?
4. *Contributions*: What are the paper's main contributions?
5. *Clarity*: Is the paper well written?

http://www.sigcomm.org/sites/default/files/ccr/papers/2007/July/1273445-1273458.pdf

You Spend a Lot of Time Reading

- Reading for grad classes
- Reviewing conference submissions
- Giving colleagues feedback
- Keeping up with your field
- Staying broadly educated
- Transitioning into a new areas
- Learning how to write better papers ☺

It is worthwhile to learn to read *effectively.*

Keshav's Three-Pass Approach: Pass 1

- A ten-minute scan to get the general idea
  - **Title, abstract, and introduction**
  - Section and subsection titles
  - Conclusion
  - Bibliography
- What to learn: the five C's
  - Category: What type of paper is it?
  - Context: What body of work does it relate to?
  - Correctness: Do the assumptions seem valid?
  - Contributions: What are the main research contributions?
  - Clarity: Is the paper well-written?
- Decide whether to read further…

Keshav's Three-Pass Approach: Pass 2

- A more careful, one-hour reading
  - Read with greater care, but ignore details like proofs
  - Figures, diagrams, and illustrations
  - Mark relevant references for later reading
- Grasp the content of the paper
  - Be able to summarize the main idea
  - Identify whether you can (or should) fully understand
- Decide whether to
  - Abandon reading in greater depth
  - Read background material before proceeding further
  - Persevere and continue for a third pass

Keshav's Three-Pass Approach: Pass 3

- Several-hour virtual re-implementation of the work
  - Making the same assumptions, recreate the work
  - Identify the paper's innovations and its failings
  - Identify and challenge every assumption
  - Think how you would present the ideas yourself
  - Write down ideas for future work

- When should you read this carefully?
  - Reviewing for a conference or journal
  - Giving colleagues feedback on a paper
  - Understanding a paper closely related to your research
  - Deeply understanding a classic paper in the field

Other Tips for Reading Papers

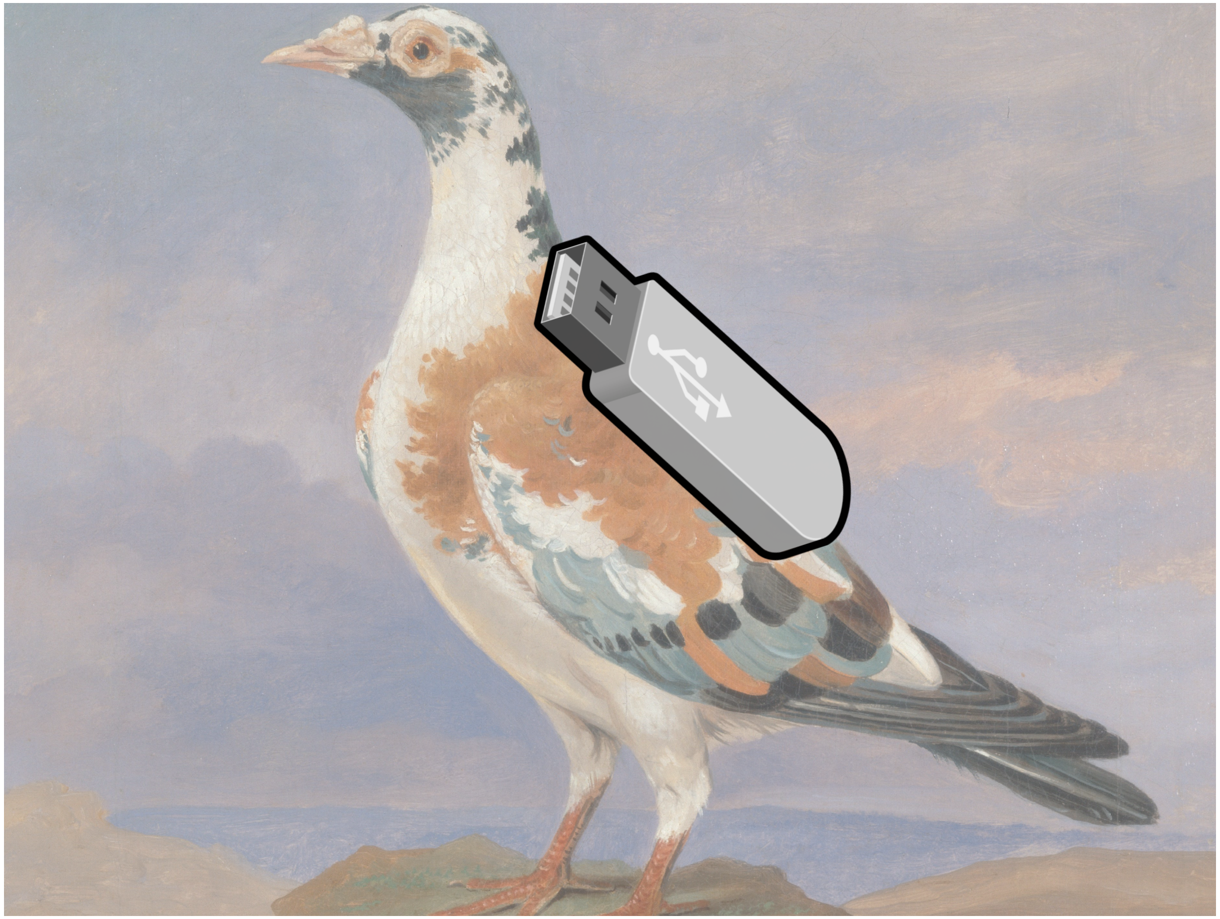
- Read at the right level for what you need
  - "Work smarter, not harder"
- Read at the right time of day and in the right place
  - When you are fresh, not sleepy
  - Where you are not distracted, and have enough time
- Read actively
  - With a purpose (what is your goal?)
  - With a pen or computer to take notes
- Read critically
  - Think, question, challenge, critique, …

Performance Measures and Application Performance

Kyunghan Lee
Networked Computing Lab (NXC Lab)
Department of Electrical and Computer Engineering
Seoul National University
https://nxc.snu.ac.kr
kyunghanlee@snu.ac.kr

Adapted from Dr. Ankit Singla's slides

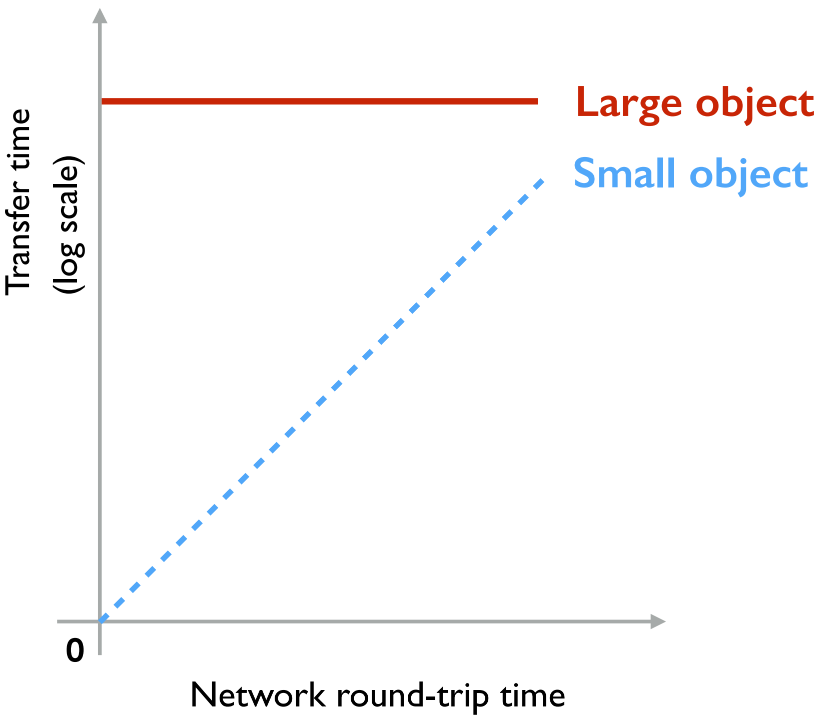
Performance: Bandwidth? Latency?



4400 km to Fly ➔ 80 km/h with 1 TB USB stick = 40 Mbps

... but 55 hours

Bandwidth and Latency

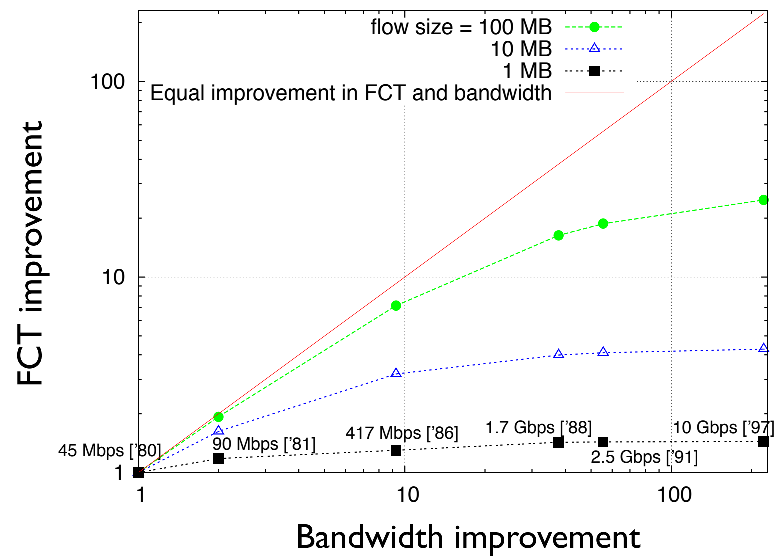


ACM CCR 2006

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



Beyond Flow Completion Time?

- How long does https://www.google.com take to load?
- 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!

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."*

ACM CCR 2007


Flow Rate Fairness: Dismantling a Religion

Bob Briscoe
BT Research & UCL
bob.briscoe@bt.com

Let’s Talk about Reliability

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

- What if no reliable transport is provided?
  - Programmer burden
    - Every application that needs reliability has to engineer it from scratch
  - 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

- Solution 1:
  Check reliability at every step (involving network layer)
- Solution 2:
  Allow unreliable steps (network layer is best-effort) B checks and tells A to retry on failure (Can still fail, but only if A / B themselves fail.)

Should we ever Implement Reliability in the Network?

$$P\ (retry) = 1 - 0.90^{10} = 0.65$$

$$P\ (retry) = 1 - 0.99^{10} = 0.10$$

...

- Implementing reliability in the network ...
  - ... does not reduce end-host complexity
  - ... does increase network complexity
  - ... often imposes overhead for apps that don't need it
  - ... but can enhance performance in some cases

End-to-End Argument Interpretations

IEEE ICDCS 1981

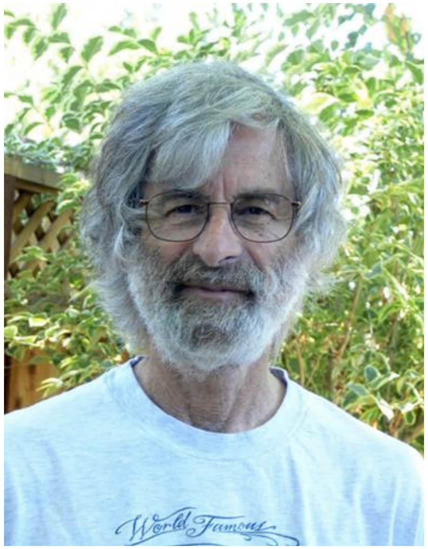
<table><tr><td>
END-to-END Arguments in System Design

J.H. Saltzer, D.P. Reed, and D.D. Clark*

M.I.T. Laboratory for Computer Science
</td></tr></table>

1. Only if sufficient
   - Don't implement a function at a lower layer unless that is complete
2. Only if necessary
   - Don't implement a function at a lower layer unless hosts cannot
3. Only if useful
   - If hosts can, implement in-network only as an optimization BUT only if not burdensome for apps that don't need it

The Fate-Sharing Principle



A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable.

— *Leslie Lamport*, Microsoft Research

How do we prevent this?

The Fate-Sharing Principle

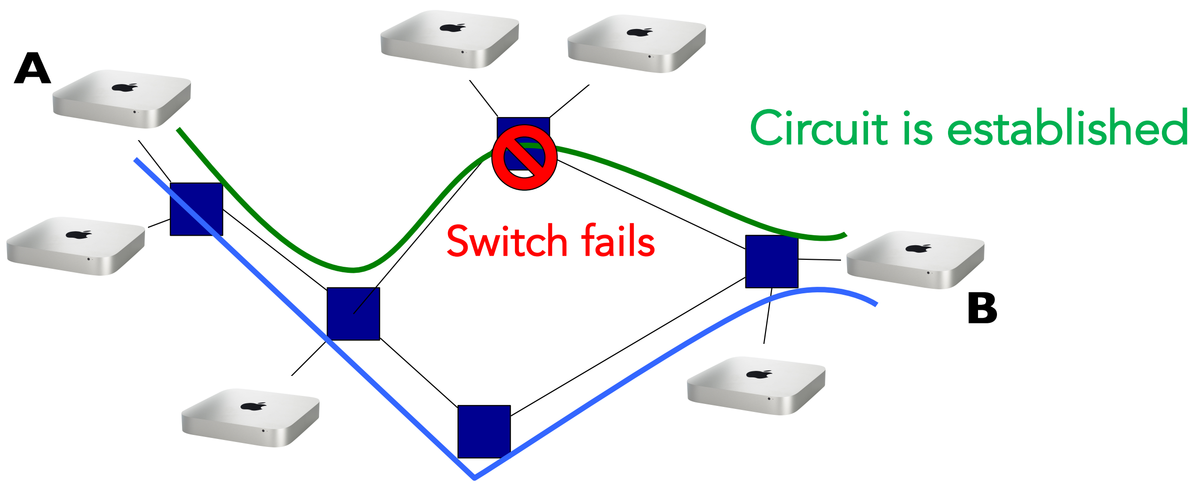
- When storing state in a distributed system, co-locate it with entities that rely on that state!

- State is lost only if those entities fail; then it doesn't matter. e.g., network connection state at end hosts.

- Survivability in Networking
  - End-points should be able to continue communicating without resetting conversation, even under failures.

Packet vs. Circuit Switching

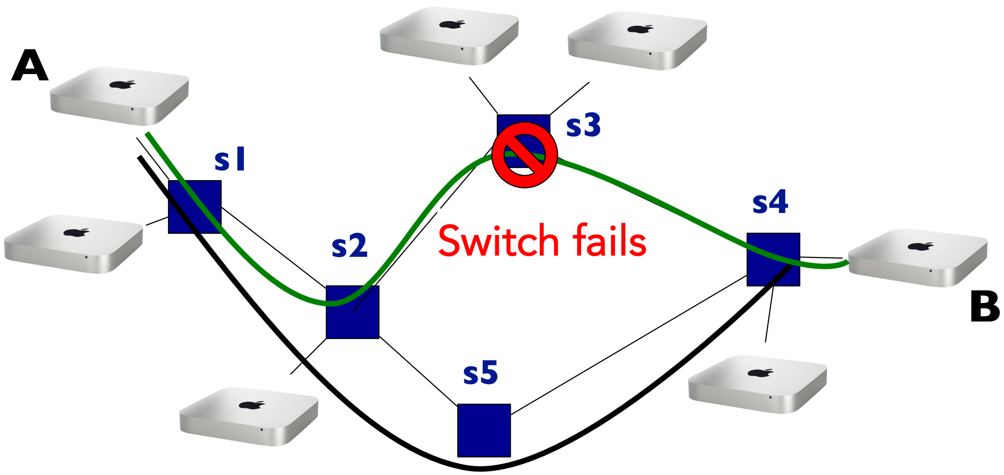
- One problem with circuit switching is that it doesn't route around trouble



A is forced to signal a new circuit to restore communication.

Packet vs. Circuit Switching

- One problem with circuit switching is that it doesn't route around trouble



Route recomputed on the fly by s2

Packet switching beats Circuit switching with respect to *resilience* and *efficiency.*

Packet vs. Circuit Switching

Advantages

- efficient use of resources simpler to implement than circuit switching
- route around trouble

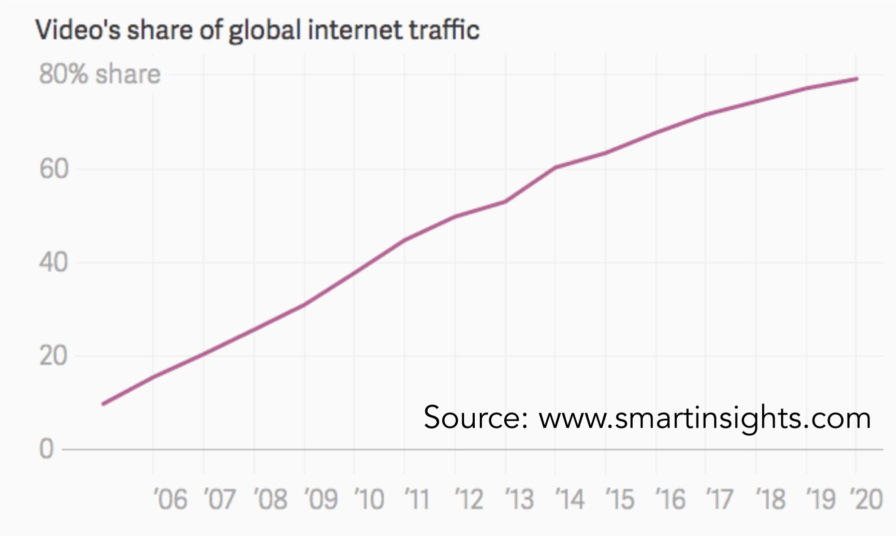
Disadvantages

- unpredictable performance
- requires buffer management and congestion control

Reservation makes sense when
Peak / Average ratio is small
Voice traffic has a ratio of 3 or so

Reservation wastes capacity when
Peak / Average ratio is big
Data applications are bursty, ratios >100 are common

Is this true for Internet video?



Quiz 0 (Submit it within 20 minutes and Leave)

- Write down the followings in the header of your A4 paper.
  - [Quiz 0] "Your student ID" – "Your name"
  - Submit it with the email title: [2020S WN] Quiz # Student ID Your Name
    - Toward nxclab@gmail.com

- Explain the following terms one by one and describe further how these terms are correlated to each other as much as detailed. (Define notations if needed.)
  - Doppler shift
  - Doppler spread
  - Delay spread
  - Coherence distance
  - Coherence time
  - Coherence bandwidth