Network performance

Lecture 25

CS 638 Web Programming

Overview

- Measures of network performance
- Network congestion
- Caching
- Performance-related features of HTTP 1.1

The performance of one link

- Data rate (a.k.a. bandwidth): the number of bits one can send on the link every second
  - Measured in Kbps, Mbps, Gbps
    - 1 Kbps = 1,000 bits per second
    - 1KB (kilobyte) = 2^{10} bytes (1,024 bytes)
- Propagation delay: time it takes for one bit to travel from one end of the link to the other
- Latency of a message: time from when the first bit of the message to when last bit received at other end
  - Latency = propagation delay + transmit time
  - Transmit time = message size / data rate

What makes a link “fast”?

- It depends on message size whether propagation delay or data rate dominates latency

Link characteristics	Latency (in ms)	
	1 byte message	1 KB message
Data rate: 1Kbps Propag. delay: 1ms	1+8=**9**	1+8192=**8193**
Data rate: 1Mbps Propag. delay: 100ms	100+0.008=**100.008**	100+8.192=**108.192**

Performance of a network path

- Path between sender and receiver has multiple links with various data rates and propagation delays
- The rate at which you can send data cannot exceed the smallest of the data rates of the links
  - If your web page is too large it will take long to download
- Path latency is sum of link latencies
  - Routers on the path send message to next link only after they receive entire message from previous link
- Round-trip-time: time it takes for a small packet to go from sender to receiver and back
  - Time between request and reply ≥ round trip time

Typical network performance

- Typical data rates for various types of links
  - Dial up modems 10 – 50 Kbps (still widely used!)
  - DSL around 1 Mbps
  - Cable TV between 1 and 10 Mbps
  - Local area networks between 10 Mbps and 100Mbps
  - High speed network backbones tens of Gbps
- Typical roundtrip times
  - Within local area network under 1 ms
  - Within U.S. between 10 and 50 ms
  - To overseas between 100 and 250 ms

Overview

- Measures of network performance
- Network congestion
- Caching
- Performance-related features of HTTP 1.1

Many users share the network

- What happens when two packets that need to go on the same link arrive to router at same time?
  - Router stores one of them until it sends out the other
  - Queuing delay adds to roundtrip time
- What happens when the rate of traffic for a link is larger than the link's data rate?
  - Router queue fills up and packets are dropped
- Network congestion results in large queuing delays and many dropped packets
- Often the data rate achieved by an individual transfer is below the data rate of the network path

Internet congestion control

- Core idea: when a computer observes a packet loss, it sends future traffic slower
  - If there are no packet losses and sender has data to send, rate is increased slowly
- Implemented as part of the TCP protocol by every computer on the Internet
- Due to this strategy, severe packet losses are rare
- Malicious users can still send large amounts of traffic to congest network (network floods)

Overview

- Measures of network performance
- Network congestion
- Caching
- Performance-related features of HTTP 1.1

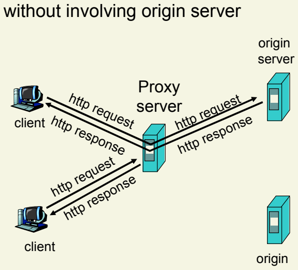
Local caching at client

- Your browser builds a cache of the documents you visited recently (html files, images, style sheets, etc.)
- When you request a new page the browser first checks the cache before contacting the server
  - Serving a request from the local cache is much faster
  - Images, style sheets, and javascript files may be shared by multiple pages, so the cache can help even with pages never visited before
- Server may mark dynamically generated pages as uncacheable
  - Images in such pages can still be cached

In-network caches (a.k.a. proxy servers)

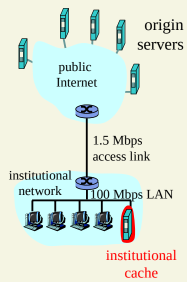
Goal: satisfy client request without involving origin server

- User sets browser: Web accesses via web cache
- Client sends all http requests to web cache
  - If object in web cache, it is returned to client
  - Otherwise web cache requests object from origin server, then returns object to client



Why use network caches?

- Assuming cache close to client
- Advantages
  - Smaller response time
  - Decrease traffic to distant servers (uplink often bottleneck)
- Disadvantages
  - Introduces new point of failure
  - Some overhead on misses
  - Does not work with dynamic personalized content
- Decreasing popularity



Content delivery networks

- Run by companies that own many web caches throughout the Internet (e.g. Akamai)
- Large web sites can buy the services of CDNs
  - Benefit: lower load at servers, lower latency at clients
  - Often CDNs carry only the images, not the actual html files
  - Typically URL of images in html files changed
- Clients need not configure anything
  - By cleverly manipulating DNS, the CDN makes clients retrieve the images from the nearest cache
  - You have used CDNs before

Overview

- Measures of network performance
- Network congestion
- Caching
- Performance-related features of HTTP 1.1

HTTP and performance

- HTTP 1.1 introduced in 1997
- Most new features help improve performace
  - Support for compression
  - Persistent connections
  - Pipelining
  - Better support for caching

Persistent connections

- HTTP 1.0 opened a separate TCP connection for each request
  - When opening a TCP connection, the client has to wait at least one roundtrip before sending the HTTP request (due to TCP handshake)
- HTTP 1.1 uses persistent connections: the same TCP connection can be used for multiple requests to the same server
  - Improves performance when a page contains many objects
- Request pipelining: the client can send next request before receiving the answer to the previous one

Conditional GET: client-side caching

- Goal is not to send object if client has up-to-date cached version
- Client specifies date of cached copy in request
  `If-modified-since: <date>`
- Server response contains no object if cached copy is up-to-date:
  `HTTP/1.0 304 Not Modified`