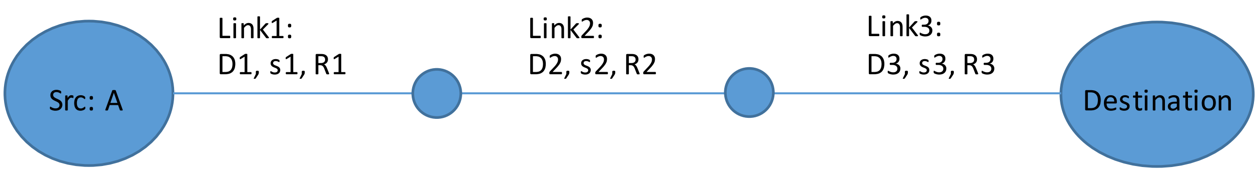
Computer Networks Homework 1 9/23/2016

Username: Ronair

Ch1 P10. Find end-to-end delay:



End to end delay = processing + queueing + transmission + propagation

- Processing delay = 2 * d_{proc} (one at each packet switch)
- Queueing delay = 0
- Transmission delay: time to transmit packet onto link
  $= \sum_{i=1}^{3} L/Ri = L\,(\frac{1}{R1} + \frac{1}{R2} + \frac{1}{R3})$
- Propagation delay: time needed to send the packet through the link
  $= \sum_{i=1}^{3} di/si = \frac{d1}{s1} + \frac{d3}{s2} + \frac{d3}{s3}$

∴ End to end delay = 2 * d_{ouc} $+ L\left(\frac{1}{R1} + \frac{1}{R2} + \frac{1}{R3}\right) + \frac{d1}{s1} + \frac{d3}{s2} + \frac{d3}{s3}$

Substituting values for Part 2:
Converting time to milliseconds, bytes to bits, distances to km

End to end delay = $[2 * 3] + \left[1500 * 8 \left(\frac{3}{2*10^3}\right)\right] + \left[\frac{1}{2.5*10^2} * (5000 + 4000 + 1000)\right]$
$= 64\ msec$

Ch1 P13. Queueing delay:

a. For the first packet, the queueing delay is 0. The 2nd packet has to wait for L/R time units while the 1st packet is being served, the 3rd packet has to wait for 2*L/R time units for packets 1 and 2 to be served. In general, a packet waits for (N-1)*L/R time units. Adding them up and taking the average over N, we get:

Average Queueing delay

$$= \frac{L}{R} * \frac{0 + 1 + 2 + 3 \ldots (N-1)}{N}$$

$$= \frac{L*(N-1)}{2*R} \ldots \text{using } 1 + 2 + 3 \ldots N = \frac{N*(N+1)}{2}$$

$$= \frac{N-1}{2} * \frac{L}{R}$$

b. The transmission time is $N * \frac{L}{R}$ for N packets. So when the next batch of N packets arrive, they see an empty buffer. ∴ the average delay is effectively the average delay within the first set of N packets, which is calculated to be $= \frac{N-1}{2} * \frac{L}{R}$ above

Ch1 P21. Throughput:

If the server can use only one path to send data to the client:

- Path1 throughput = $\min(R_1^1, R_2^1, R_3^1 \ldots R_N^1)$
- Path2 throughput = $\min(R_1^2, R_2^2, R_3^2 \ldots R_N^2)$

…

∴ Max throughput = $\max(\min(R_1^1, R_2^1, R_3^1 \ldots R_N^1), \ldots\ldots \min(R_1^M, R_2^M, R_3^M \ldots R_N^M))$

But the performance would be better if the server can use all M paths:

Throughput = $\sum_{k=1}^{M} \min(R_1^k, R_2^k, R_3^k \ldots R_N^k)$

Ch1 P23. Bottlenecks:

a. If the 1st link is the bottleneck:

The second packet waits for the transmission of the first packet at the first link for $\frac{L}{Rs}$ time units, which is the inter-arrival time at the destination

b. If the 2nd link is the bottleneck:
We need to consider 2 time delays
1. T2 = 2nd packet arriving at the 2nd link

$$= \frac{L}{Rs}(queueing\ delay) + \frac{L}{Rs}(transmission\ delay) + dprop\ (propagation\ delay)$$

2. T1 = 1st packet completely placed on the 2nd link

$$= \frac{L}{Rs}(transmission\ delay) + dprop\ (propagation\ delay) + \frac{L}{Rc}(transmission\ delay)$$

Since Rc < Rs, T1 > T2 i.e., the second packet gets queued at link 2.

Now if the 2nd packet is sent after T secs, to ensure no queueing, we need:
T2 + T >= T1

i.e., $\frac{L}{Rs} + \frac{L}{Rs} + dprop + T \geq \frac{L}{Rs} + dprop + \frac{L}{Rc}$

$\therefore$ min $T = L(\frac{1}{Rc} - \frac{1}{Rs})$

Ch1 P29. Delays:

$R = 10 * 10^6 bits/sec$

$a = \frac{1}{60}\ photos/sec$

$s = 2.4 * 10^8 \frac{meters}{sec}$

$d = 3.6 * 10^7 meters$

a. Propagation delay $= \frac{d}{s} = 150\ msec$

b. BW-delay product = R * propagation delay $= 1.50 * 10^6$ bits

c. For continuous transmission, we must have the traffic intensity = 1 i.e., $a * \frac{x}{R} = 1$

$\therefore x = 60 * 10 * 10^6 = 6 * 10^8 bits$

On verifying, we see that x/R (=60 secs) > d/s (0.150 secs), so the deciding factor is x/R.

Ch2 P1. True or False:

a. False
b. True
c. False
d. False
e. False

Ch2 P6. (d) HTTP/1.1 RFC 2616

Yes, it is possible that one side starts closing a connection while the other side is transmitting data via this connection.

Below are the lines from the RFC (section 8.1.4) that justify this claim

- "Servers will usually have some time-out value beyond which they will no longer maintain an inactive connection"
- "The use of persistent connections places no requirements on the length (or existence) of this time-out for either the client or the server"
- **"A client, server, or proxy MAY close the transport connection at any time. For example, a client might have started to send a new request at the same time that the server has decided to close the "idle" connection. From the server's point of view, the connection is being closed while it was idle, but from the client's point of view, a request is in progress."**
- "This means that clients, servers, and proxies MUST be able to recover from asynchronous close events. Client software SHOULD reopen the transport connection and retransmit the aborted sequence of requests without user interaction so long as the request sequence is idempotent."

Ch2 P10. Non-persistent vs persistent HTTP

Note:

- Each downloaded object fits into one packet.
- Let t be one-way propagation delay from client to server or the other way round

<u>Non-persistent:</u>
10 parallel connections imply each gets 150/10 = 15bits/sec bandwidth.
Total time taken

$= \frac{200}{150} + t$ … connection request

$+ \frac{200}{150} + t$ … connection response

$+ \frac{200}{150} + t$ … object request

$\frac{100000}{150} + t$ … object response

(now for the 10 referenced objects)

$+ \frac{200}{15} + t$ … parallel connection requests

$+ \frac{200}{15} + t$ … parallel connection responses

$+ \frac{200}{15} + t$ … parallel object request

$+ \frac{100000}{15} + t$ … parallel object response

$= \mathbf{7377 + 8 * t}$ **secs**

Persistent:

Total time taken

$= \frac{200}{150} + t$ … connection request

$+ \frac{200}{150} + t$ … connection response

$+ \frac{200}{150} + t$ … object request

$\frac{100000}{150} + t$ … object response

(now for the 10 referenced objects)

$+10 * \left(\frac{200}{150} + t + \frac{100000}{150} + t\right)$ … object request and response done 10 times

$= \mathbf{7351 + 24 * \boldsymbol{t}}$ **secs**

Now, if we consider transmission speed to be $3 * 10^8\ m/sec$, then $t = \frac{10m}{(3*10^8\,m/sec} =$ $30\ nano\ secs$ .. which is negligible.

So for non-persistent we have 7377 secs and for persistent we have 7351 secs, which is not a very big gain.