Last time: packet switching Now: congestion control

Congestion control is resource management: assigning limited resources of link rate to flows

- TCP: <u>flow-controlled</u> bidirectional byte stream
 - The speed is regulated by link_rate at the beginning. The steady state is limited by min(link_rate, how fast the reader is draining the byte stream (window_size)).
- Single-flow, single-hop model

- The sender sends a datagram, still waiting for the corresponding ackno, where could the datagram be (in the sender's mind):
 - Propagating on the link
 - Waiting at the router queue (bottleneck queue)
 - Could have been received by the receiver, but ackno still on the way back
 - Or the datagram or the ackno is lost/dropped
- Outstanding segments from the sender's perspective: [ackno, ackno + window_size)
- Window size: cap on the number of "outstanding" bytes
 - "Outstanding" means sent, not acked or judged lost
- Q: what if the window size is really small?

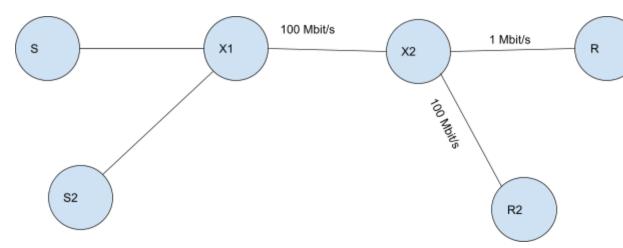
A: throughput $\frac{1 \, byte}{2 \, * \, propogation \, delay} = 0.5 \, byte/s \, (if \, propogation \, delay = 1 \, s)$ (RTT = 2*propogation delay)

Q: what if the window size is gigantic?

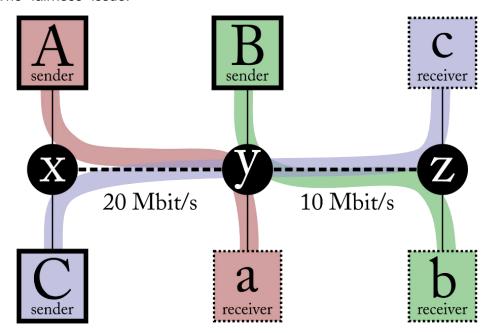
A: maximum throughput: 1 Gbit/s and the router may run out of memory or huge queueing delay.

We call this **congestion**.

- Congestion is bad
 - Congestion collapse (in the 1980s): receivers were advertising large window_size and forced the router to drop lots of packet
 - Or some flows send too much, others are starved. There is an issue for fairness.
- Useful work should increase as demand increases. It's okay if the derivative is less than 1, but it should not be the case that the derivative is negative.
 - Single-flow, single-hop model would not have a collapse, since the throughput stays at 1 Gbit/s even if many packets are dropped
 - The "collapse" issue:



- If there is only 1 flow: S2 → R2, throughput would be 100 Mbit/s
- If there is two flows: S → R and S2 → R2, throughput would be ~51 Mbit/s if S sends at 100 Mbit/s (COLLAPSE)
- If there is two flows: $S \to R$ and $S2 \to R2$, S sends at 1 Mbit/s and S2 sends at 99 Mbit/s. Throughput would be 100 Mbit/s
- The "fairness" issue:



- A -> a, B -> b, C -> c, total
 - 15, 5, 5, 25 20, 10, 0, 30 16, 6, 4, 26 0, 0, 2010, 10

- max-min fair max utilization/efficiency proportionately fair congestion collapse
- The objective: maximizing a utility function: $\max_{\{x_r\} \in S} \sum_r U_r(x_r)$ and $U(x) = \frac{x^{1-\alpha}}{1-\alpha}$
 - $\alpha = 0$, max utilization
 - $\alpha \rightarrow 1$, proportional fairness

- $\alpha = 2$, min-potential-delay fairness
- $\alpha \rightarrow inf$, max-min fairness
- Other objectivesL
 - Minimize flow completion time (of average download)
 - Minimize page load time (with many download flows)
 - Maximize "power" (= throughput / delay)
- Algorithms to prevent collapse are called "congestion control"
 - What is the right window size?
 - How should flows learn the window size?