CMSC417 Spring 2016 Lecture #13 3/23/2016

Agenda

⇒ project 3 assigned, due 4/6 @ 11:59pm

⇒ review sliding window

⇒ reciever-based flow control

- □ when to ack
- □ how to ack

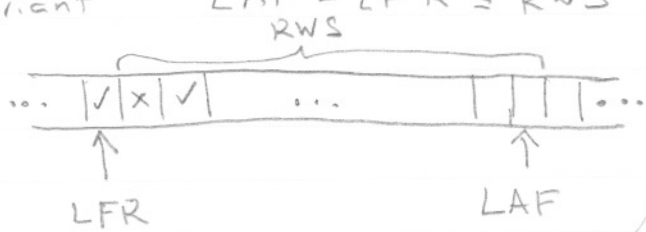
⇒ how many seq #s do you need?

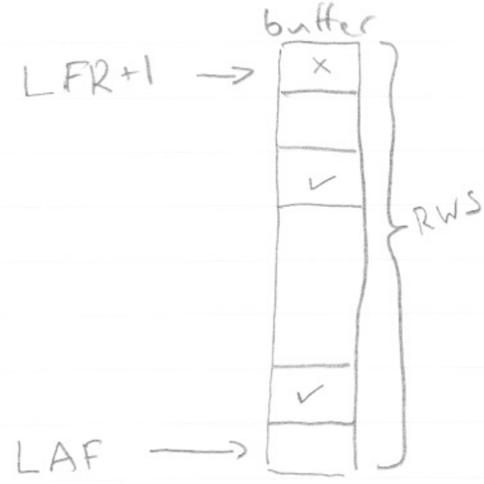
⇒ TCP

Sliding Window Receiver

⇒ Keeps 3 variables
- ⇒ RWS = receiver window size
- ⇒ LAF = last acceptable frame
- ⇒ LFR = last frame recieved

⇒ invariant $LAF - LFR \leq RWS$





SWS ~= buffer size
even if frames arrive out-of-order,
I have this many slots to store frames
and still deliver them to the application
in order

on recving frame w/seq # s

```
if (s ≤ LFR)
    discard (may ack)   // we already got it
else if (LFR < s ≤ LAF)
    accept
    must ack
else
    discard        // no space
```

What to ack?

- ⇒ the frame you just received?
- ⇒ all recvd frames?
- ⇒ the last frame you've recvd everthing up to?
- ⇒ the "holes" you have? (nack == negative ack)

This is called a cummulative ack

- ⇒ what TCP does
- ⇒ in acks, set seq # to LFR
- ⇒ locally keep LAF = LFR + RWS

Example

LFR = 5 RWS = 4 ⇒ LAF = 9

we get 7, 8 → buffer
ack 5

we get 6 → relase 6, 7, 8 to app
ack 8

LFR = 8 RWS = 4 ⇒ LAF = 12

⇒ Do you need to tell the sender the RWS?
could it help?

⇒ How big should RWS be? How big does it need to be?

- ☐ RWS = 1 is called go-back-N
- ☐ why is it called that? when is it efficient?

Finite seq #s

How many seq #s do we need?

clearly |seq #s| ≥ SWS
and
|seq #s| ≥ RWS

assume SWS = RWS = 7
seq #s in 0..7, so 8 total

reciever is now expecting 0 - 6

* oops!
receiver thinks it's the new 0, but it's actually the old 0

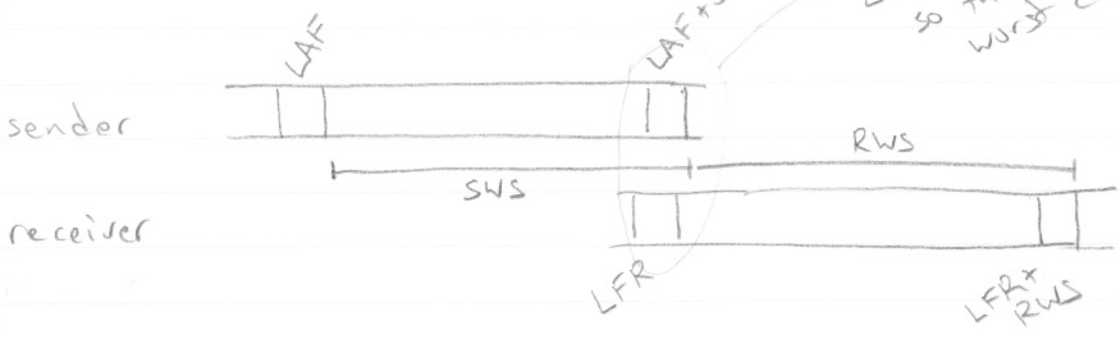
⇒ sender will only send frames from LAR+1 to LAR+SWS
⇒ receiver is expecting frames from LFR+1 to LFR+RWS

⇒ how far apart can these two windows get?
- sender sends everything
- receiver gets everything, but all acks lost
  - ⇒ LFR = LAR + SWS
  - ⇒ receiver expecting through LFR + RWS = LAR + SWS + RWS

|seq #s| ≥ SWS + RWS

seq # space graphically

note that LFR ≤ LAF + SWS, so this is the worst case



don't care about frames befor LAF b/c sender won't send them

don't care about frames after LFR+RWS b/c receiver will ignore them

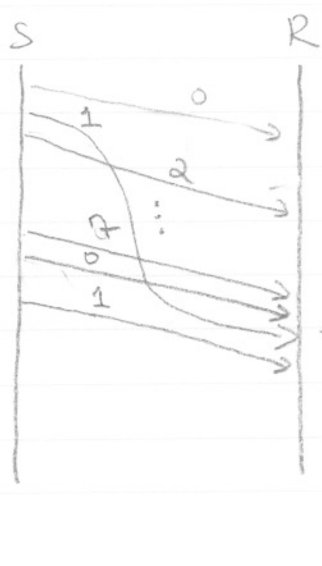
thus we need SWS+RWS unique sequence #s to be safe

if SWS = RWS, then $2\,SWS \le |seq\ \#s|$

$SWS \le |seq\ \#s| / 2$

in the book → $SWS < (|seq\ \#s| + 1)/2$

out of order frames



SWS = RWS = 4

seq #s = 0..7

* oops, receiver takes delayed 1 as the real one

when is out-of-order safe?