Read 7.E & 7.F

Flow & Congestion Control

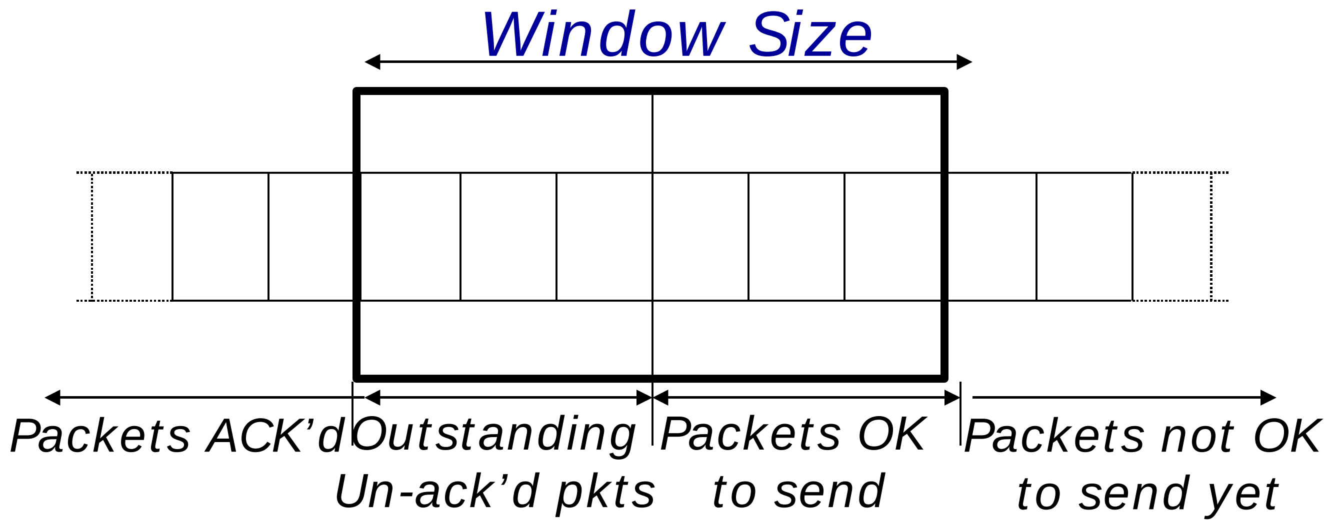
Prof. Dina Katabi

Some slides are from lectures by Nick Mckeown, Ion Stoica, Frans Kaashoek, Hari Balakrishnan, and Sam Madden

This Lecture

- *More about Sliding Window*
- *Flow Control*
- *Congestion Control*

Sliding Window

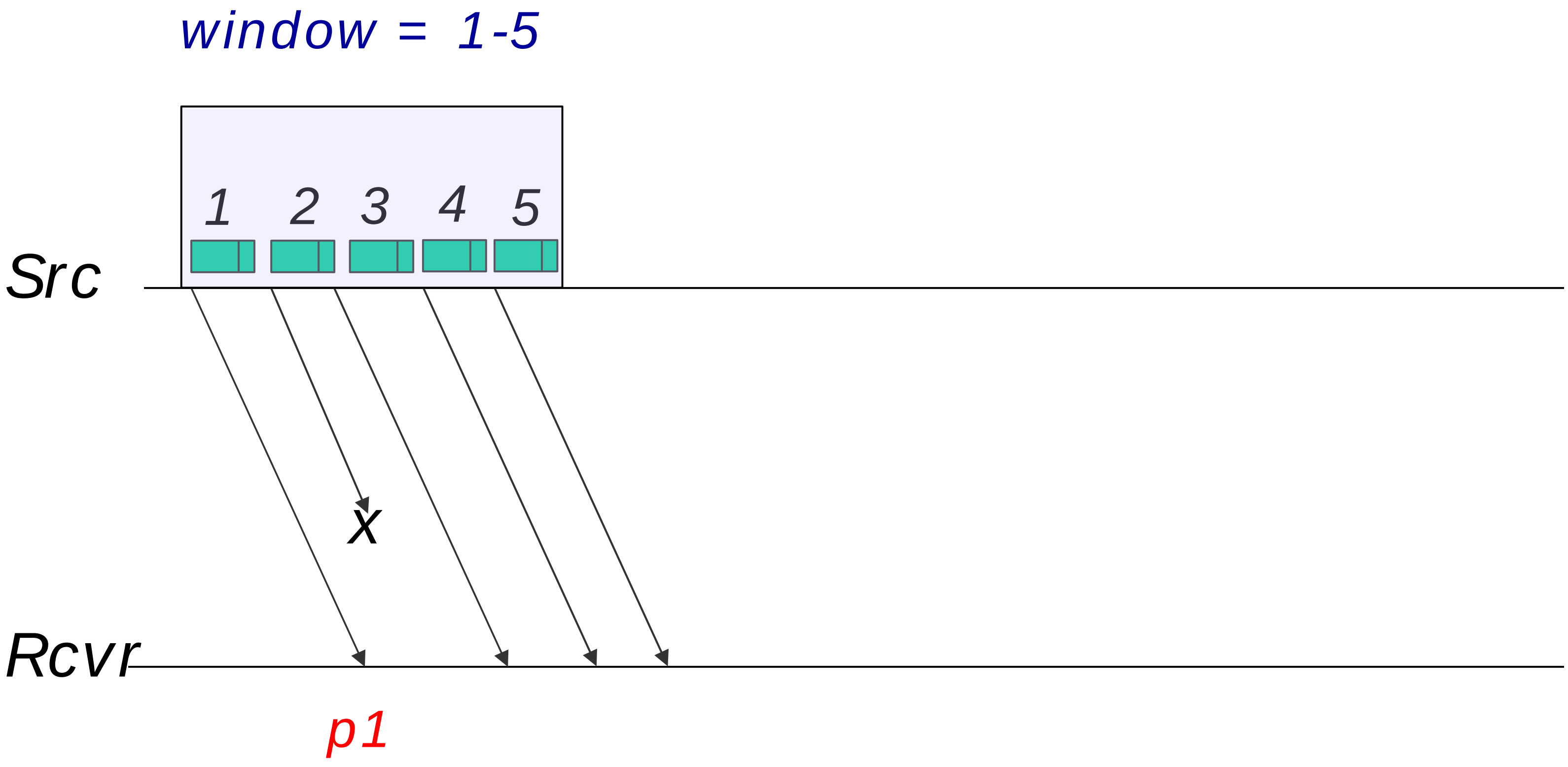


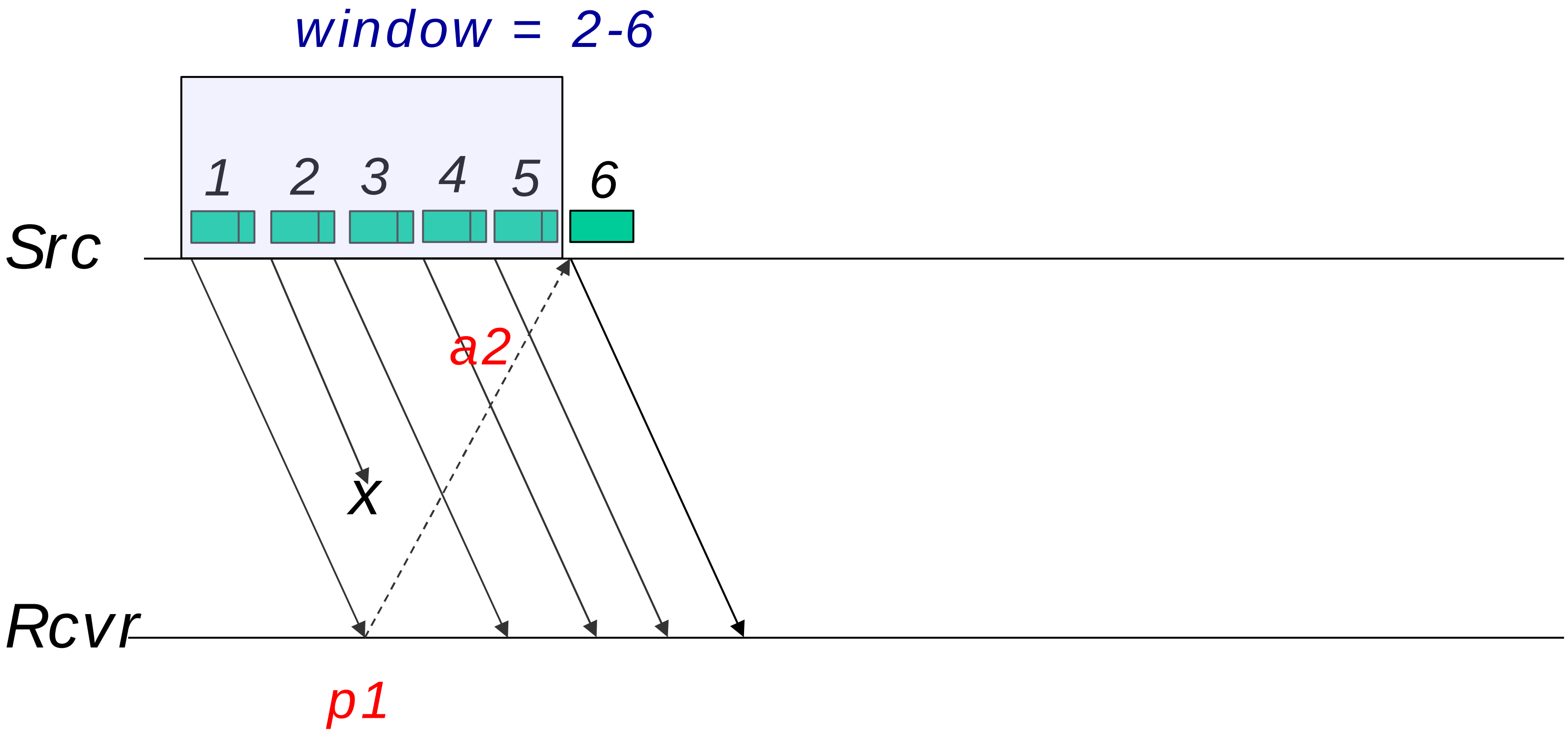
The window advances/slides upon the arrival of an ack

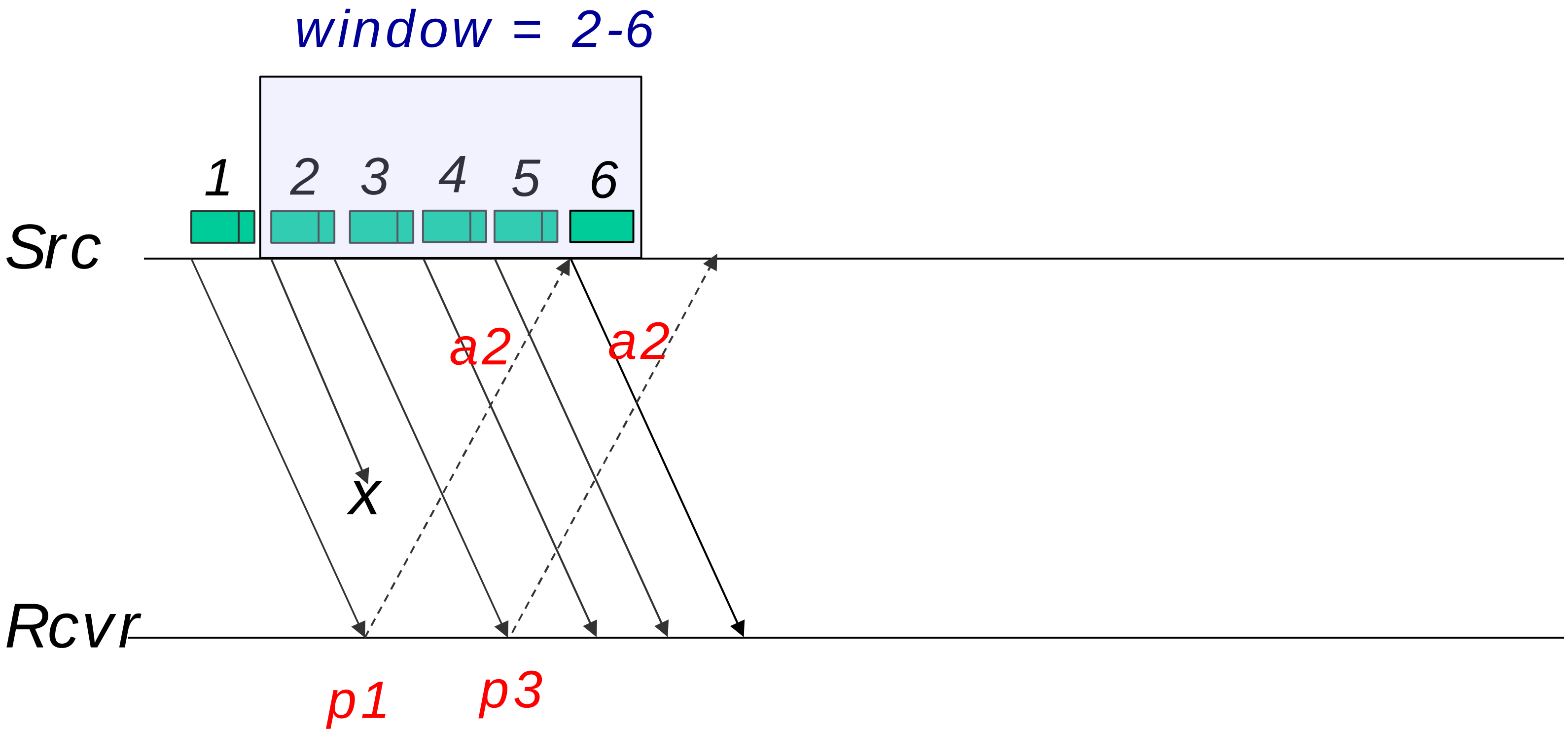
The sender sends only packets in the window

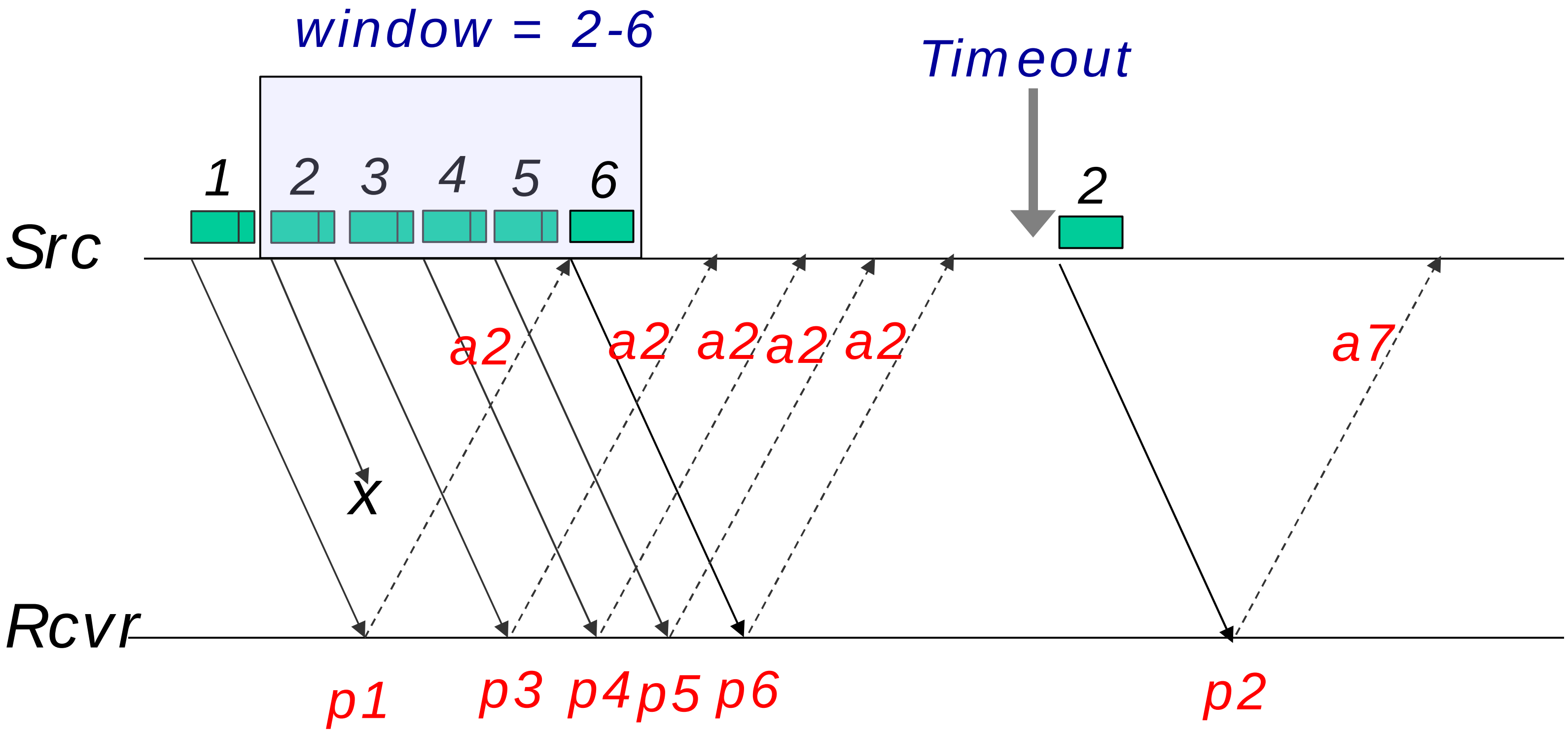
Receiver usually sends cumulative acks

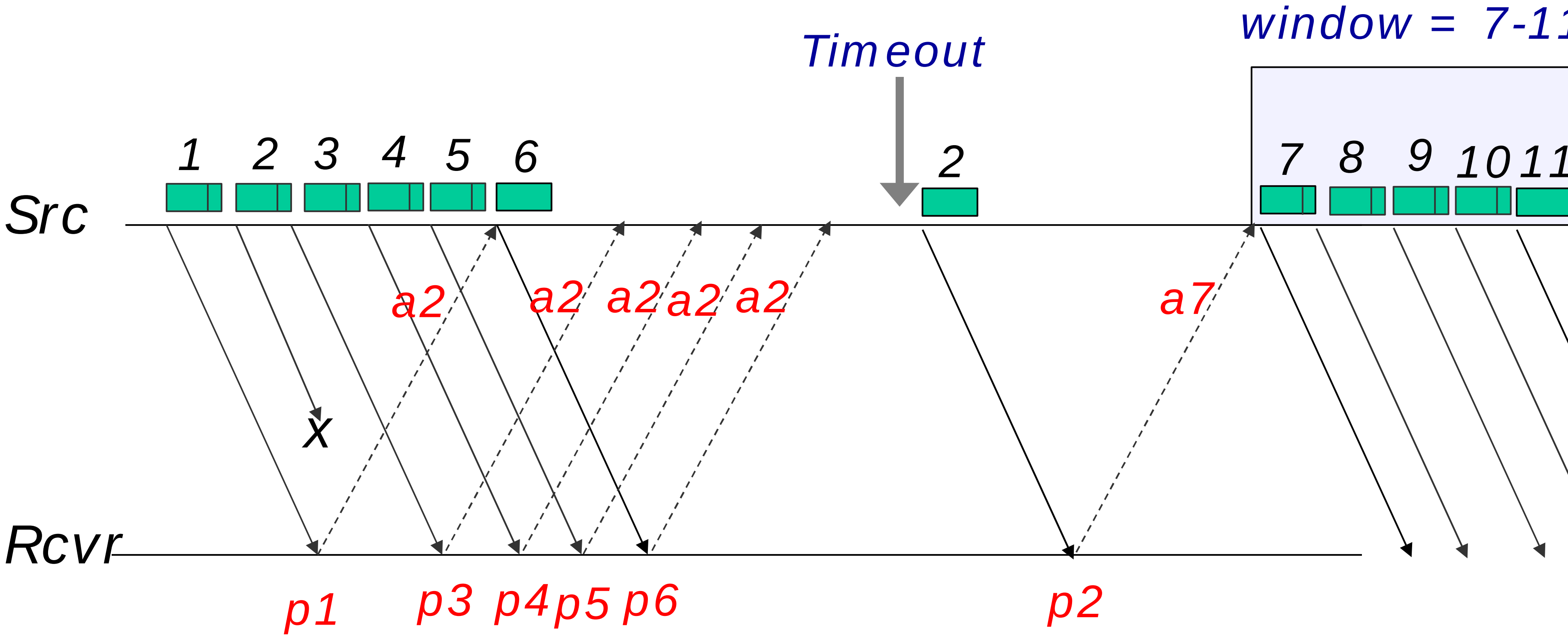
- *i.e., receiver acks the next expected in-order packet*


window = 1-5
1
2
3
4
5
Src
x
Rcvr
p1


window = 2-6
1
2
3
4
5
6
Src
a2
X
Rcvr
p1


window = 2-6
1
2
3
4
5
6
Src
a2
a2
x
Rcvr
p1
p3


window = 2-6
Timeout
1
2
3
4
5
6
2
Src
Rcvr
x
a2
a2
a2
a2
a2
a7
p1
p3
p4
p5
p6
p2



In this example, the receiver sent cumulative acks, but the same behavior happens if the receiver acks the received sequence number

This Lecture

- More about Sliding Window
- ➡ Flow Control
- Congestion Control

What is the right window size?

The window limits how fast the sender sends

Two mechanisms control the window:

- *Flow control*
- *Congestion control*

Flow Control

- *The receiver may be slow in processing the packets   receiver is a bottleneck*
- *To prevent the sender form overwhelming the receiver, the receiver tells the sender the maximum number of packets it can buffer fwnd*
- *Sender sets $W \leq fwnd$*

But, what if the bottleneck is a slow link inside the network   Need Congestion Control

This Lecture

- *More about Sliding Window*
- *Flow Control*
- *Congestion Control*

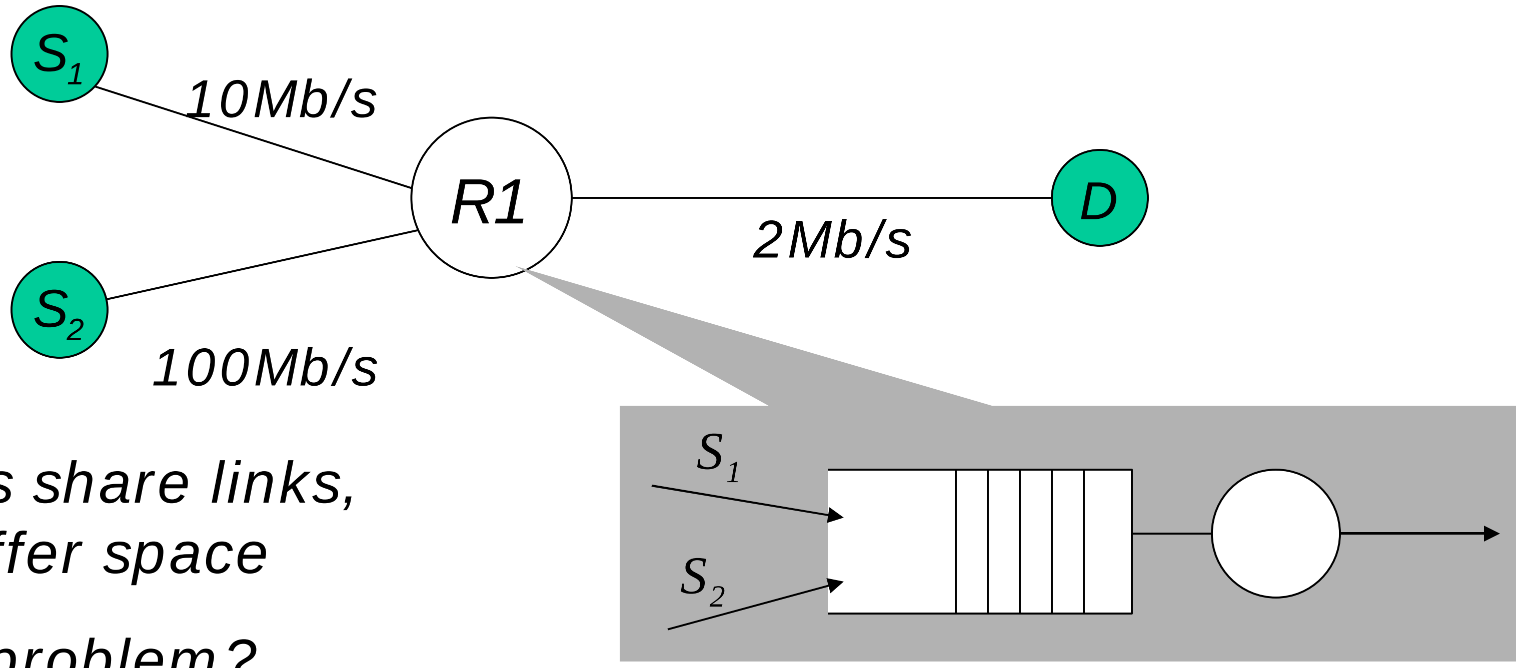
Sharing the Internet

How do you manage the resources in a huge system like the Internet, where users with different interests share the same resources?

Difficult because of:

- Size
  - Millions of users, links, routers
- Heterogeneity
  - bandwidth: 9.6Kb/s (then modem, now cellular), 10 Tb/s
  - latency: 50us (LAN), 133ms (wired), 1s (satellite), 260s (Mars)

Congestion



Sources share links, and buffer space

Why a problem?

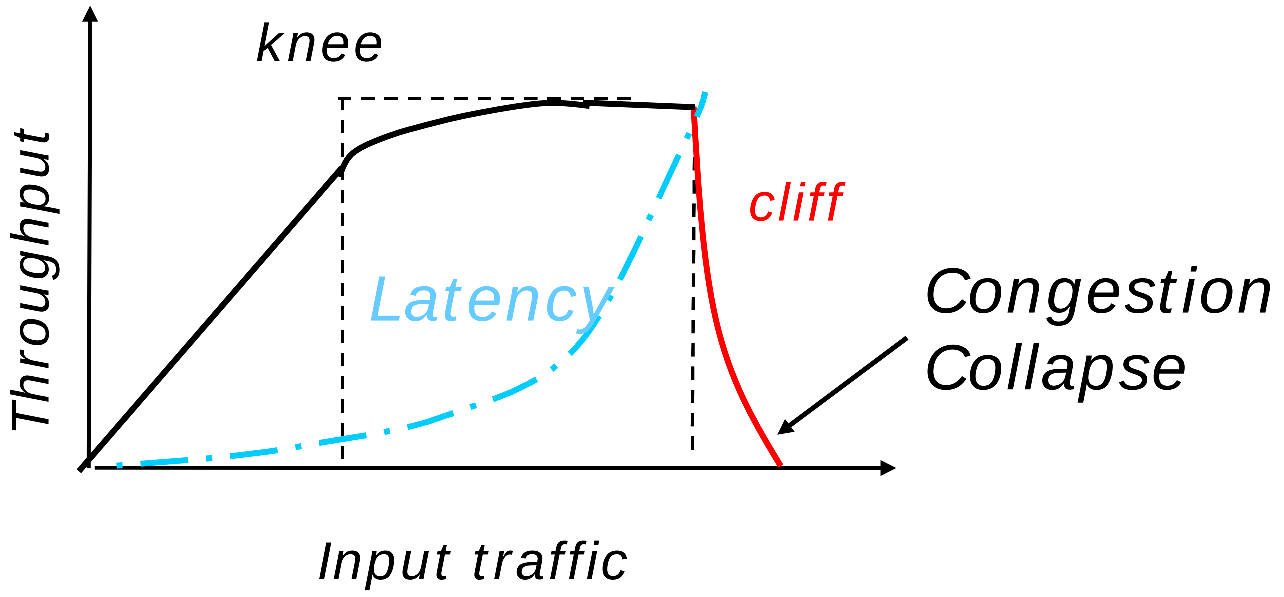
- Sources are unaware of current state of resource
- Sources are unaware of each other

Manifestations:

- Lost packets (buffer overflow at routers)
- Long delays (queuing in router buffers)

Congestion Collapse

Increase in input traffic leads to decrease in useful work



Causes of Congestion Collapse

- *Spurious retransmissions of packets that are still in flight*
- *Packet consume resources and then they are dropped downstream*

What can be done?

Increase network resources

- *But demands will increase too!*

Admission Control & Scheduling

- *Used in telephone networks*
- *Hard in the Internet because can't model traffic well*

Pricing

- *senders pay more in times of congestion*

Congestion control: ask the sources to slow down

- *But how?*
  - *How do the sources learn of congestion?*
  - *What is the correct window?*
  - *How to adapt the window as the level of congestion changes?*

How do senders learn of congestion?

Potential options:

- Router sends a Source Quench to the sender
- Router flags the packets indicating congestion
- Router drops packets when congestion occurs
  - Sender learns about the drop because it notices the lack of ack
  - Drops are the solution currently used in the Internet

How do senders learn how much to send?

- Define a congestion control window *cwnd*
- Sender's window is set to $W = \min(fwnd, cwnd)$
- Simple heuristic to find cwnd:
  - Sender increases its cwnd slowly until it sees a drop
  - Upon a drop, sender decreases its cwnd quickly to react to congestion
  - Sender increases again slowly

TCP Increase/decrease algorithm

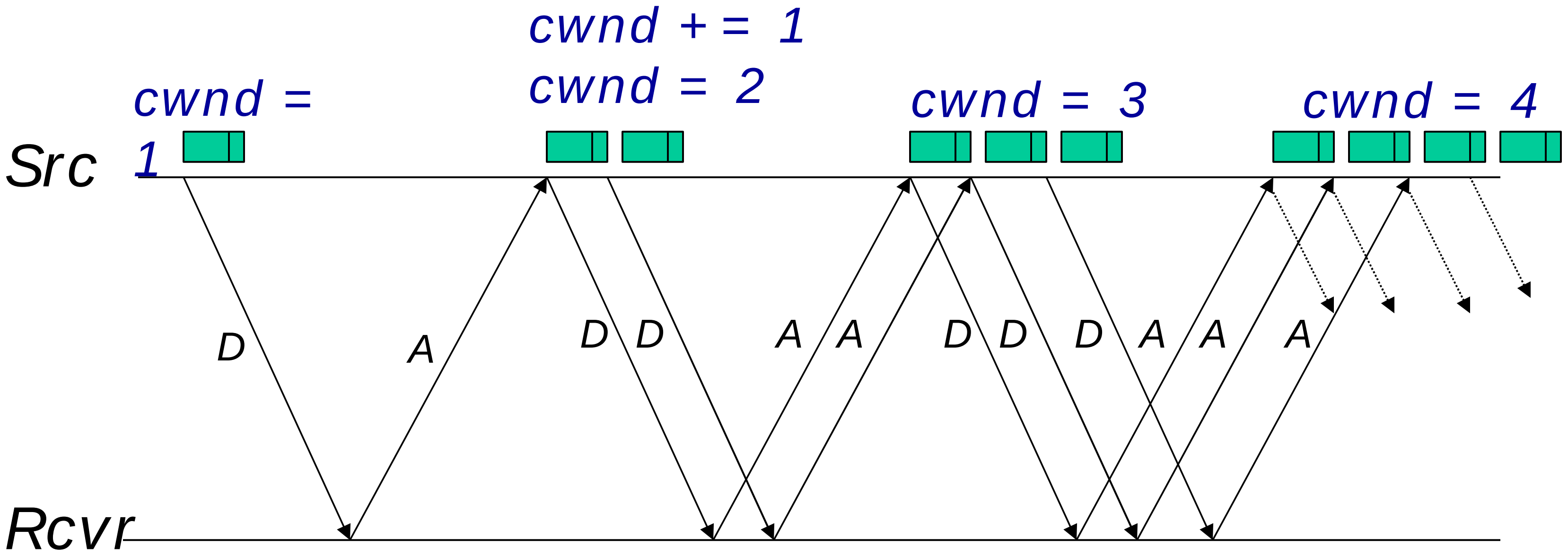
Additive Increase / Multiplicative Decrease (AIMD)

Every RTT:

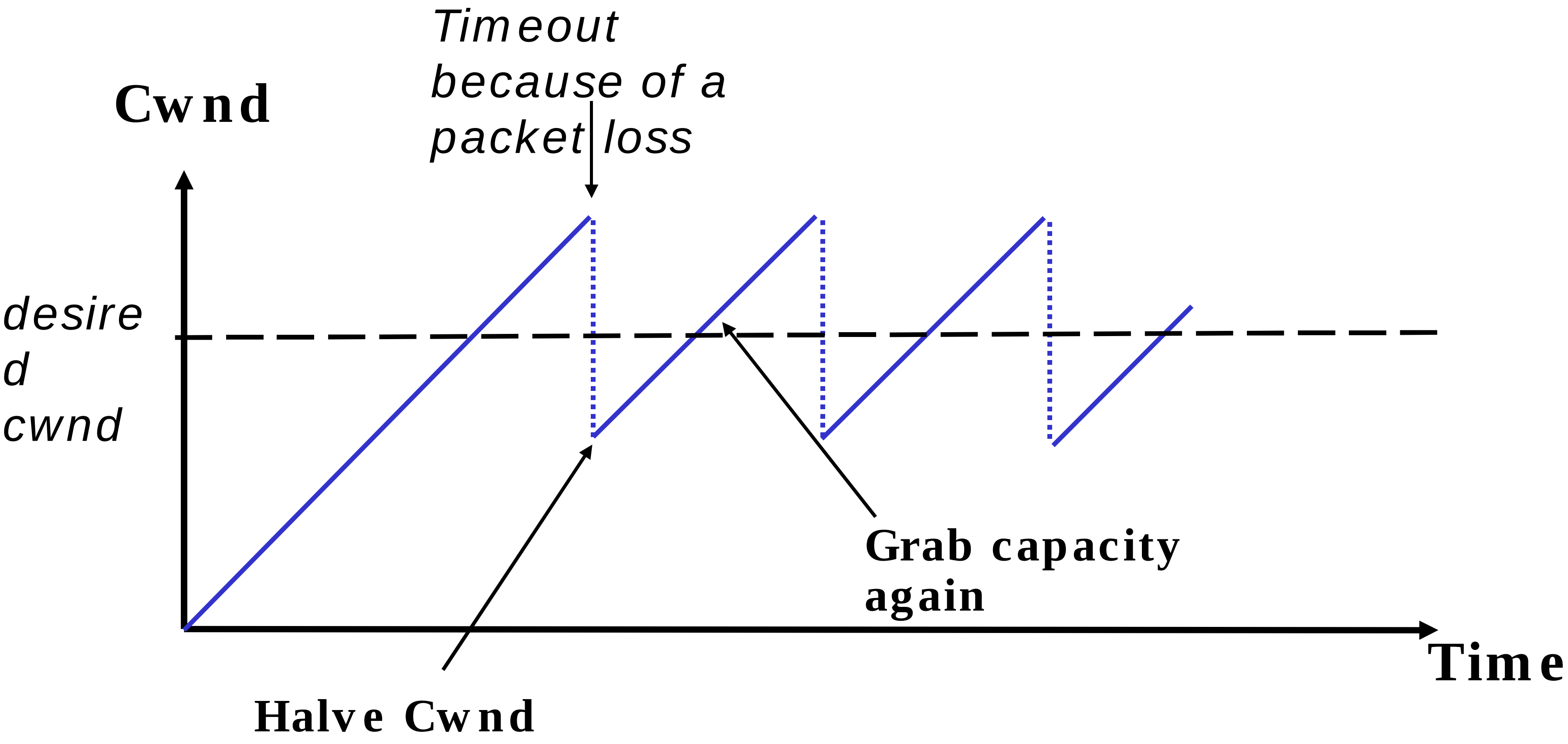
No drop: cwnd = cwnd + 1

drop: cwnd = cwnd /2

Additive Increase



TCP AIMD



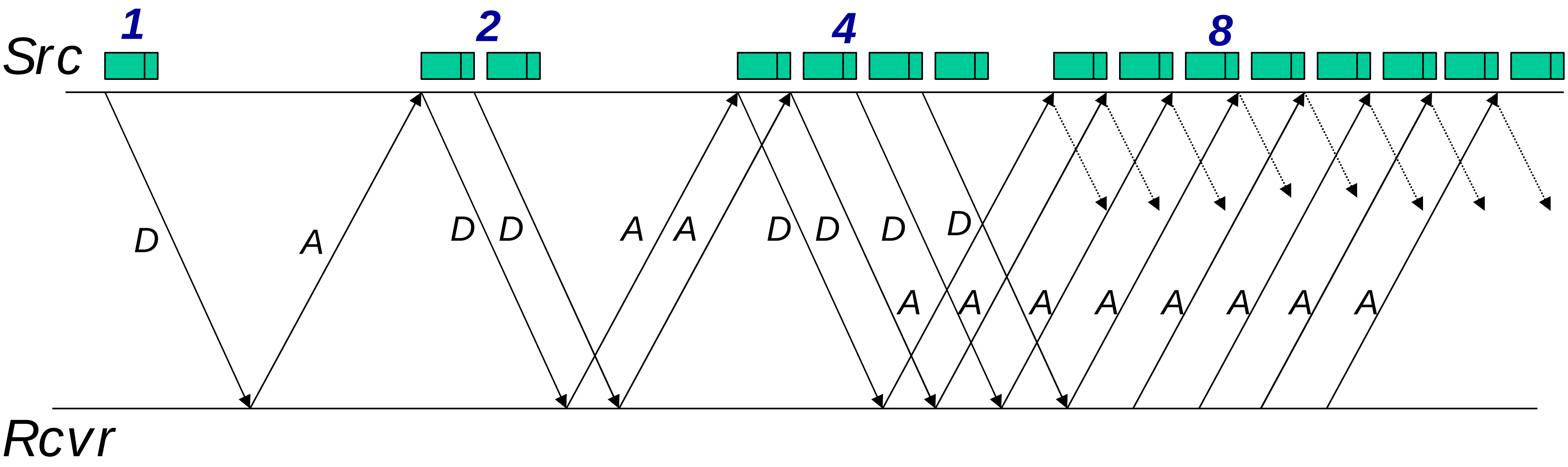
Need the queue to absorb these saw-tooth oscillations

TCP "Slow Start"

How to set the initial cwnd?

At the beginning of a connection, increase exponentially

- *Every RTT, double cwnd*



Slow Start + AIMD