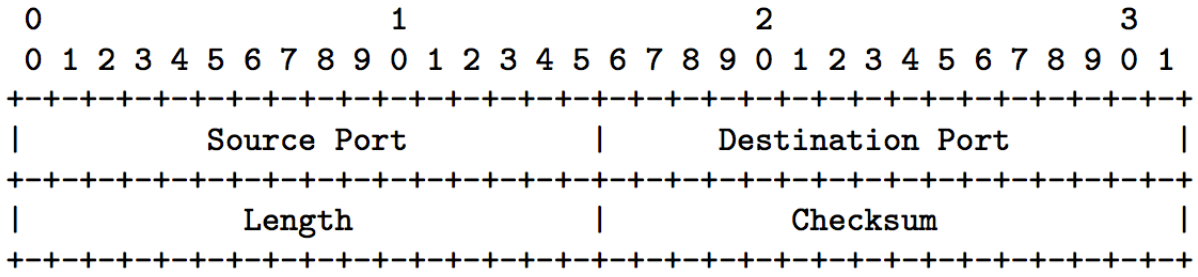
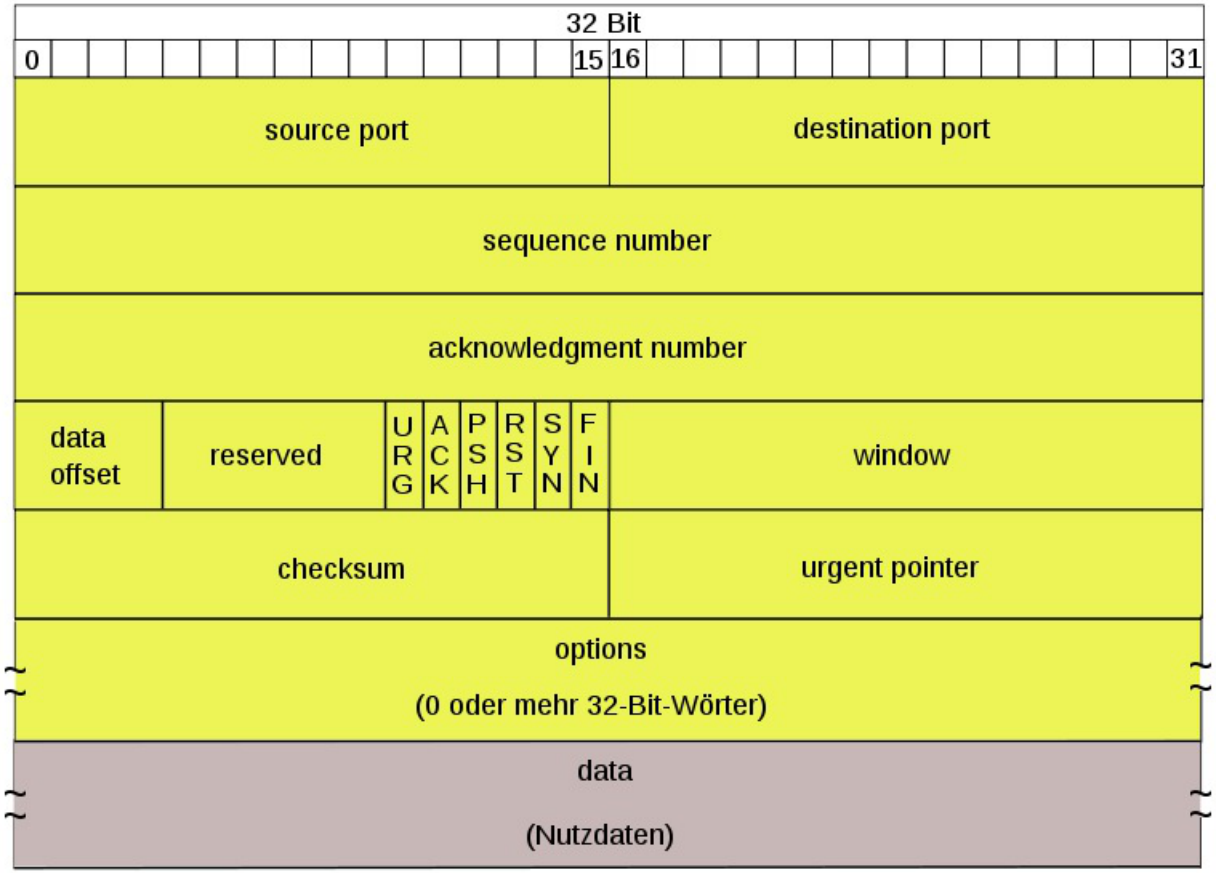


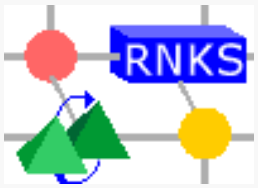


Layer 4 PDU-Structure: UDP Header (top), TCP (bottom)

Synchronisation

In bestimmten Situationen der Protokollausführung ist es erforderlich, dass sich die kommunizierenden Instanzen in definierten Zuständen befinden, um die Konsistenz bestimmter Protokollabläufe sicherzustellen.

- **Beispiele:**
  - Verbindungsaufbau
    - gewährleisten, dass beide Seiten über den Aufbau der Verbindung informiert sind und von gleichen Anfangswerten bei den Parameter-einstellungen (z. B. Sequenznummer, Fenstergröße) ausgehen
  - Verbindungsabbau
    - Verbindung muss wirklich auf beiden Seiten abgebaut sein
    - nicht zufällig einseitig !!!
- **Problem der Synchronisation**
  - nur über den Austausch von Nachrichten möglich
    - können verloren gehen !!!



Handshake-Verfahren

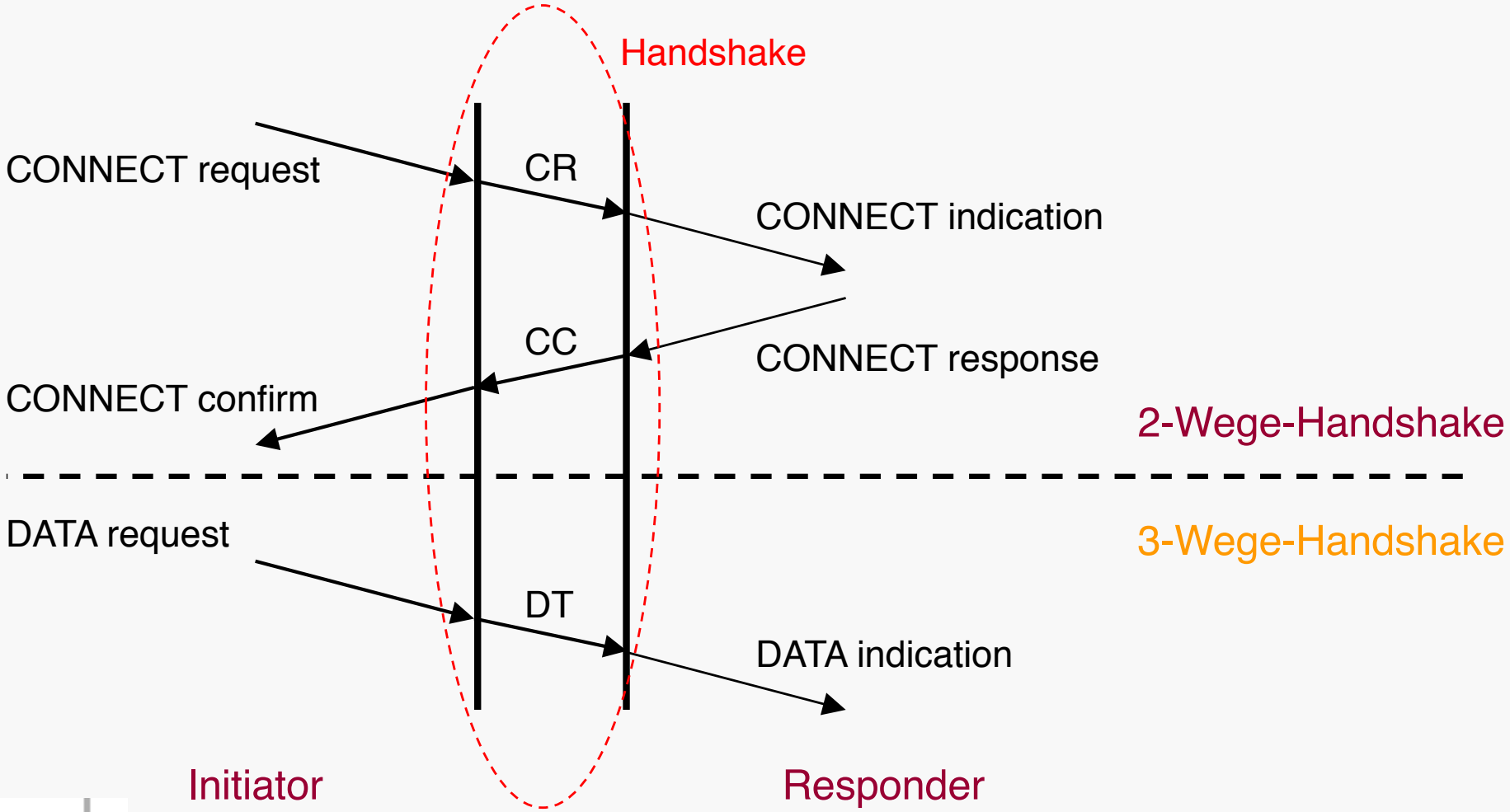
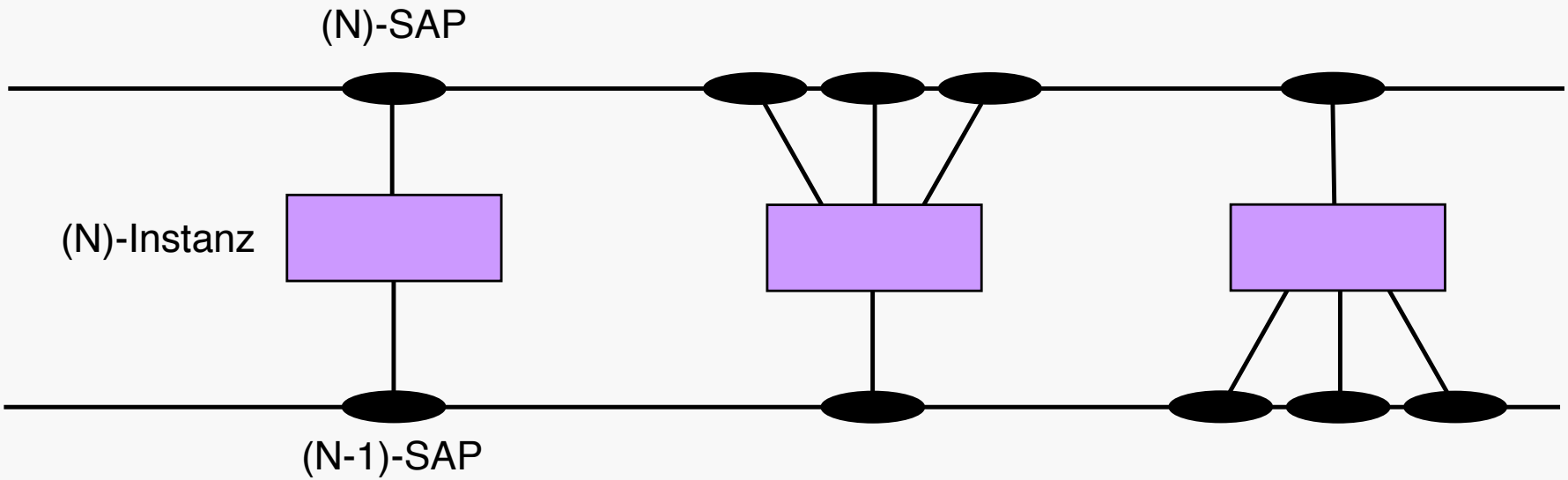


Abbildung von N- auf (N-1)-Verbindungen

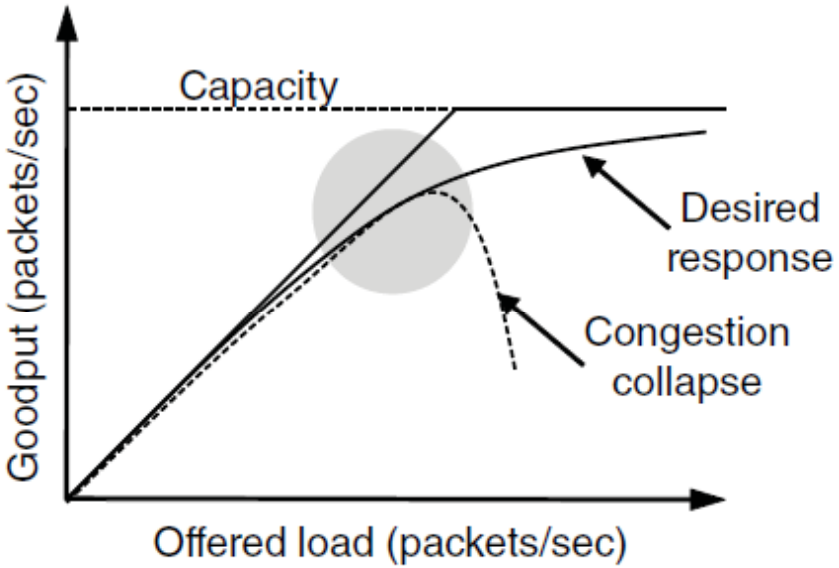
Eine weitere Aktivität des Verbindungsmanagements ist die Abbildung der Verbindung auf die (N-1)-Schicht.



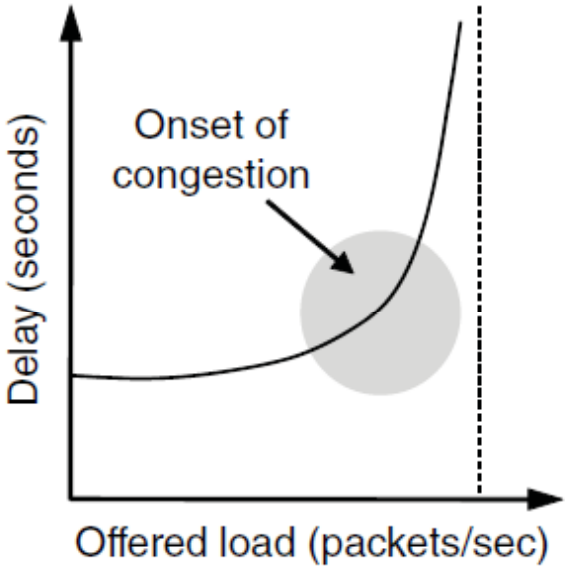
a) Eins-zu-Eins b) Multiplexen c) Splitten

Desirable Bandwidth Allocation (1)

Efficient use of bandwidth gives high goodput, low delay



Goodput rises more slowly than load when congestion sets in

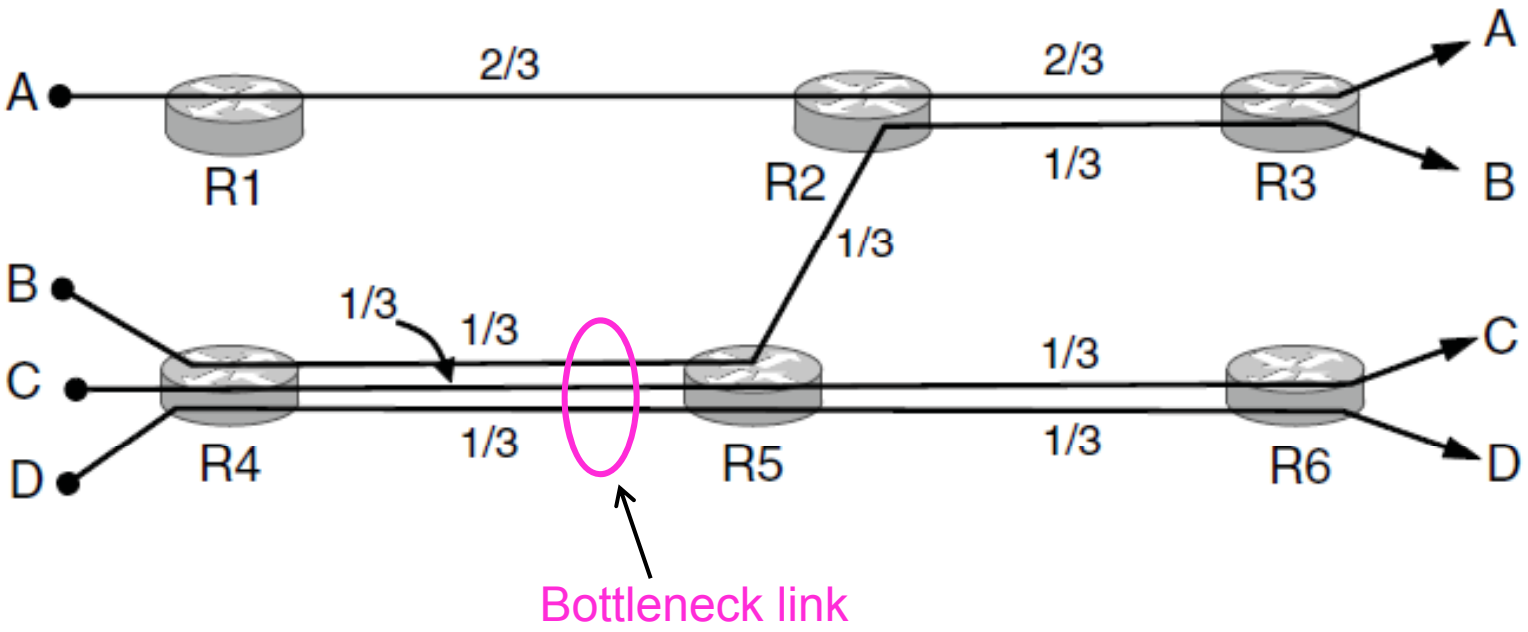


Delay begins to rise sharply when congestion sets in

Desirable Bandwidth Allocation (2)

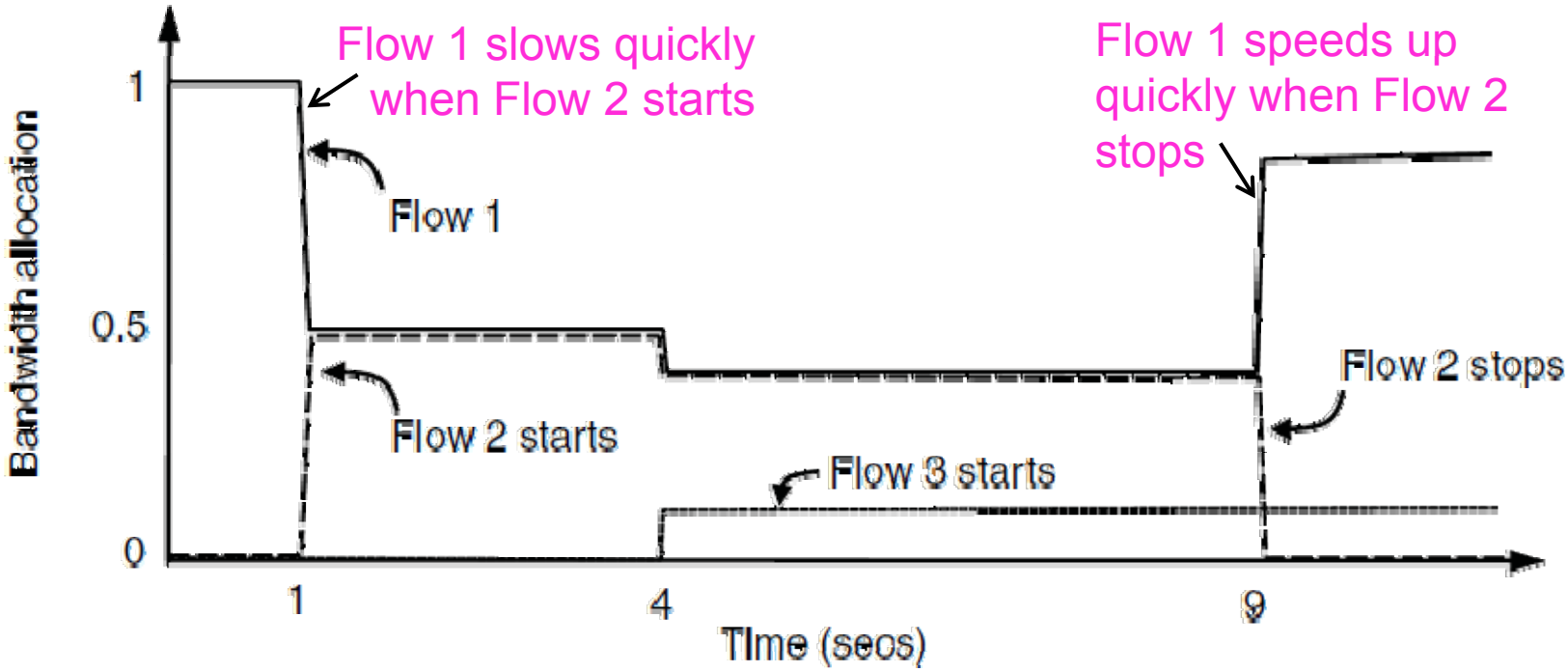
Fair use gives bandwidth to all flows (no starvation)

- Max-min fairness gives equal shares of bottleneck



Desirable Bandwidth Allocation (3)

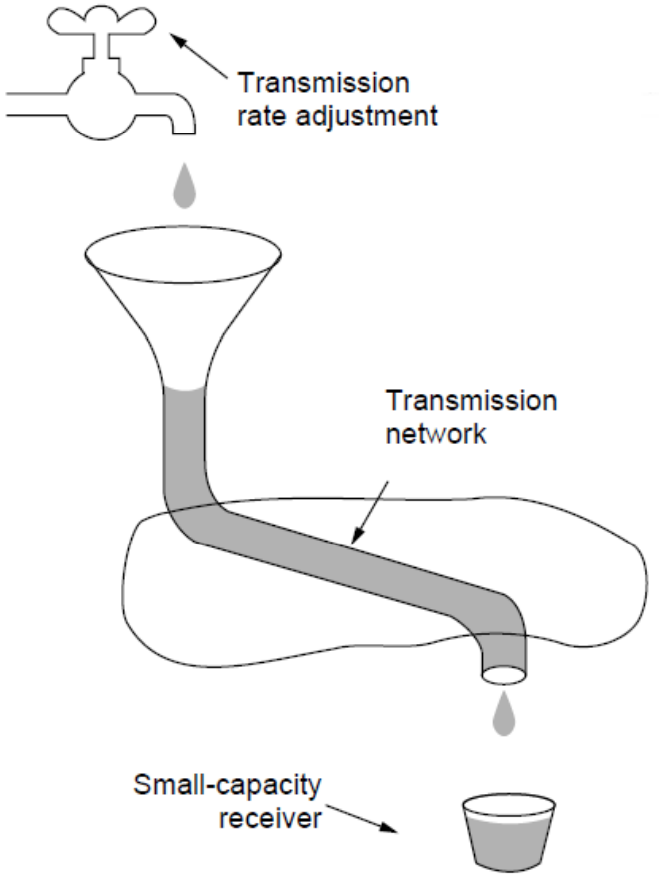
We want bandwidth levels to converge quickly when traffic patterns change



Regulating the Sending Rate (1)

Sender may need to slow down for different reasons:

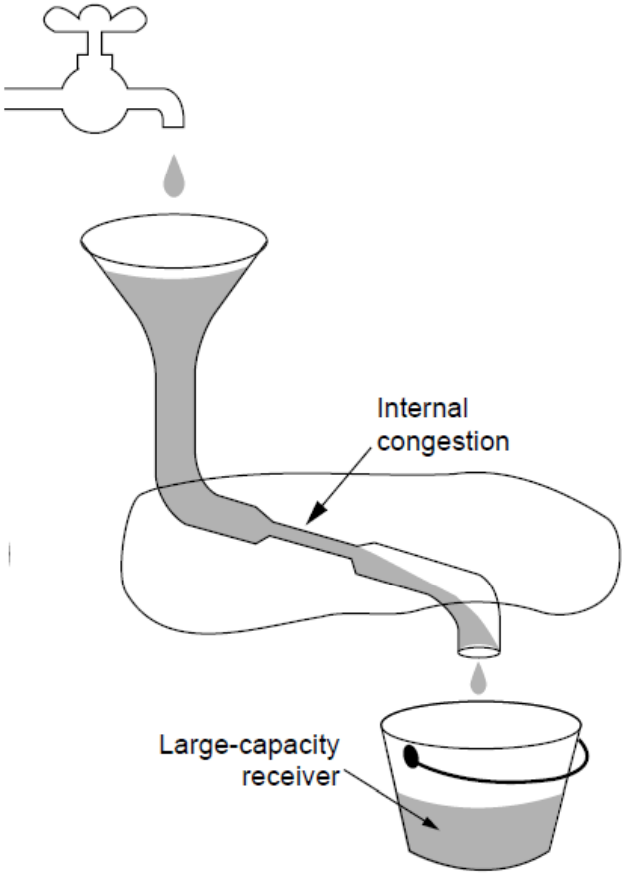
- Flow control, when the receiver is not fast enough [right]
- Congestion, when the network is not fast enough [over]



A fast network feeding a low-capacity receiver
→ flow control is needed

Regulating the Sending Rate (2)

Our focus is dealing with this problem – congestion



A slow network feeding a high-capacity receiver
→ congestion control is needed

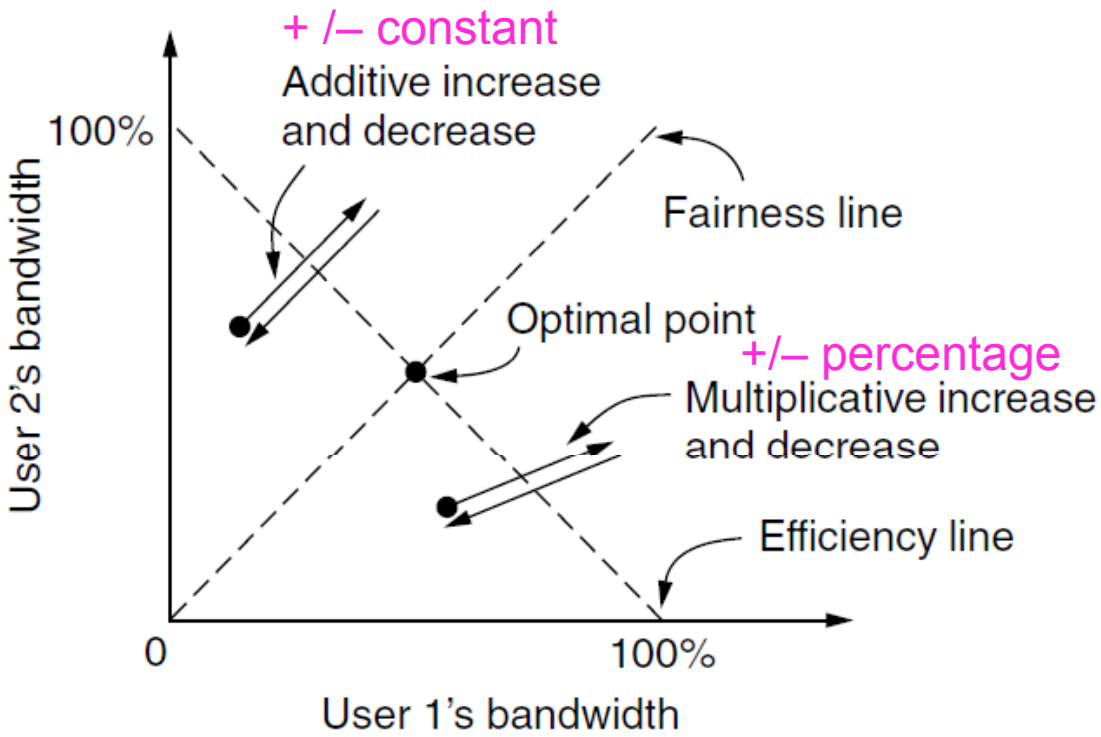
Regulating the Sending Rate (3)

Different congestion signals the network may use to tell the transport endpoint to slow down (or speed up)

Protocol	Signal	Explicit?	Precise?
XCP	Rate to use	Yes	Yes
TCP with ECN	Congestion warning	Yes	No
FAST TCP	End-to-end delay	No	Yes
CUBIC TCP	Packet loss	No	No
TCP	Packet loss	No	No

Regulating the Sending Rate (3)

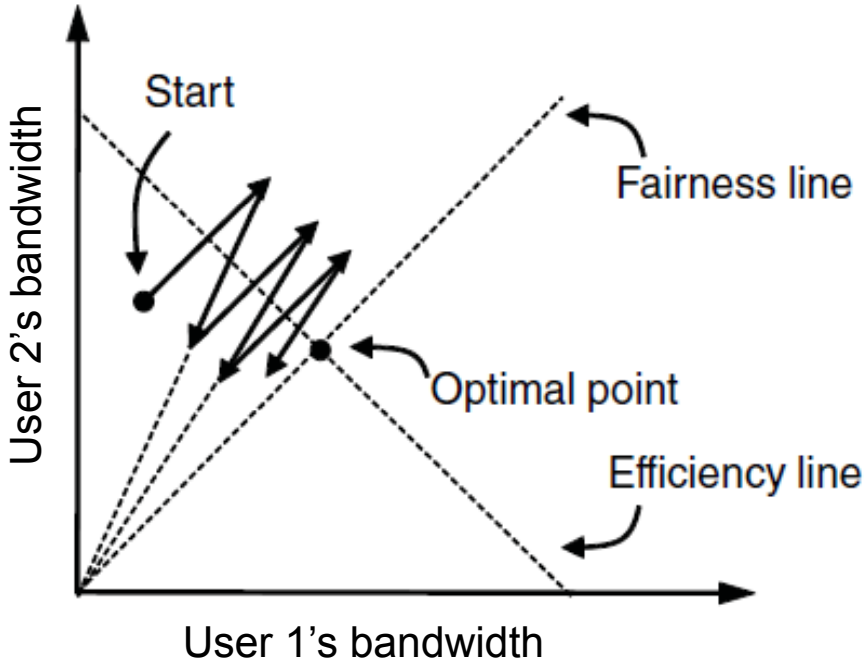
If two flows increase/decrease their bandwidth in the same way when the network signals free/busy they will not converge to a fair allocation



Regulating the Sending Rate (4)

The AIMD (Additive Increase Multiplicative Decrease) control law does converge to a fair and efficient point!

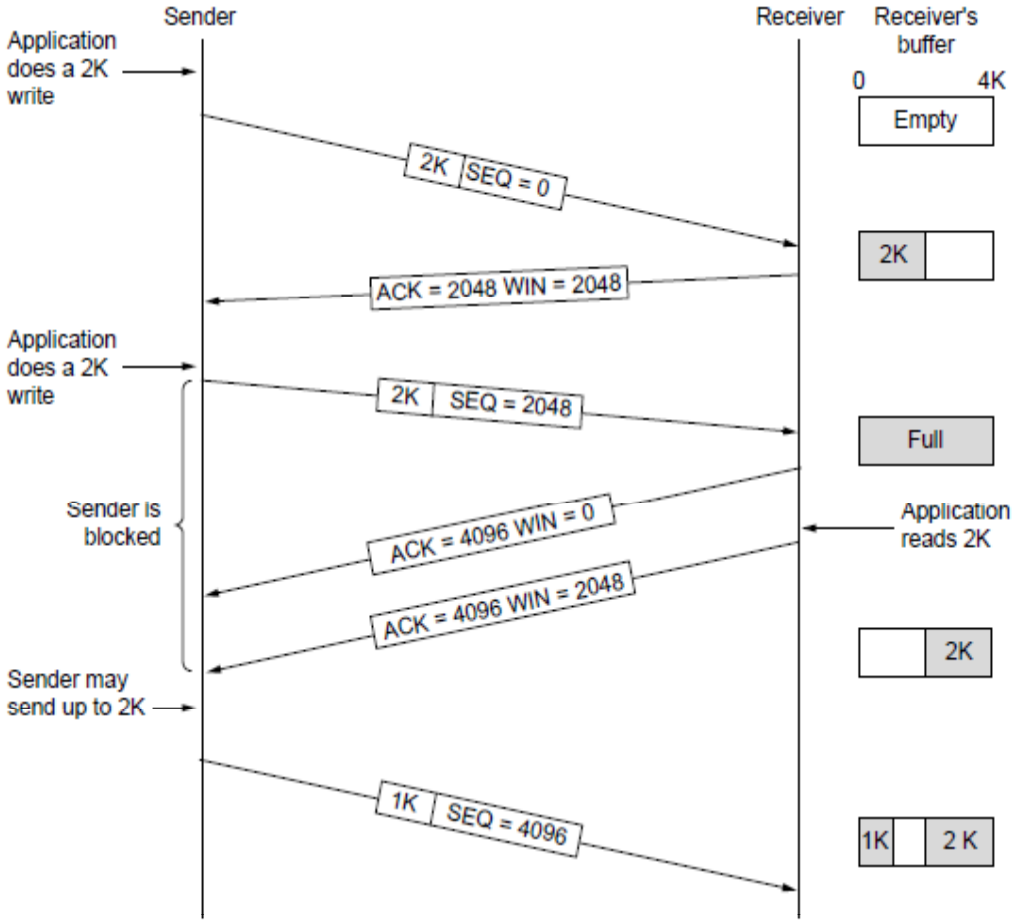
- TCP uses AIMD for this reason



TCP Sliding Window (1)

TCP adds flow control to the sliding window as before

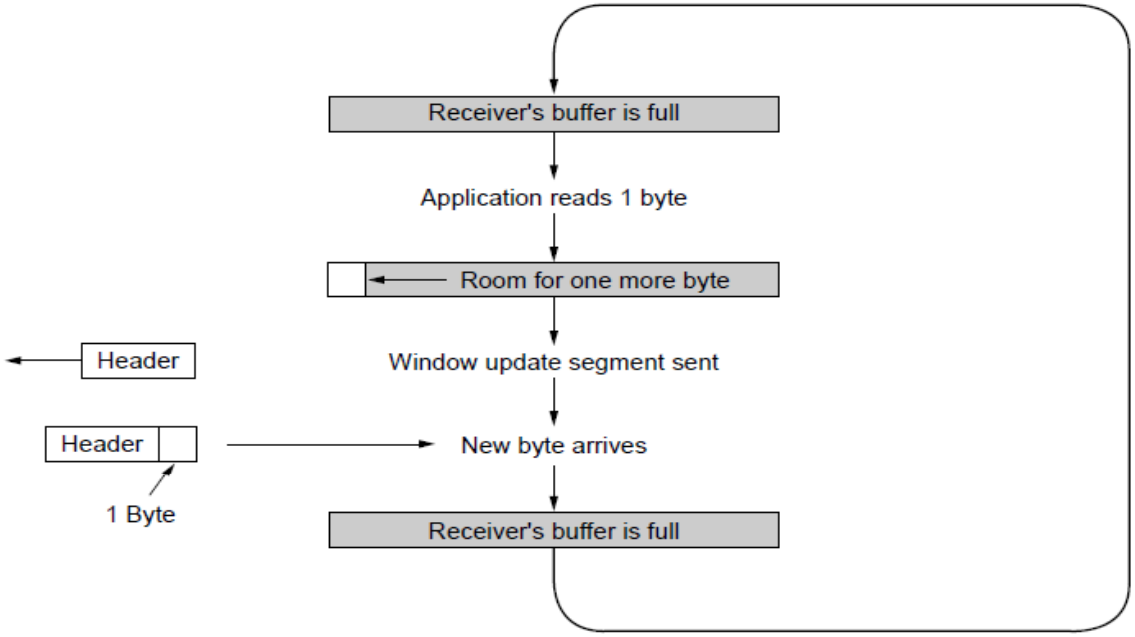
- ACK + WIN is the sender's limit



TCP Sliding Window (2)

Need to add special cases to avoid unwanted behavior

- E.g., silly window syndrome [below]

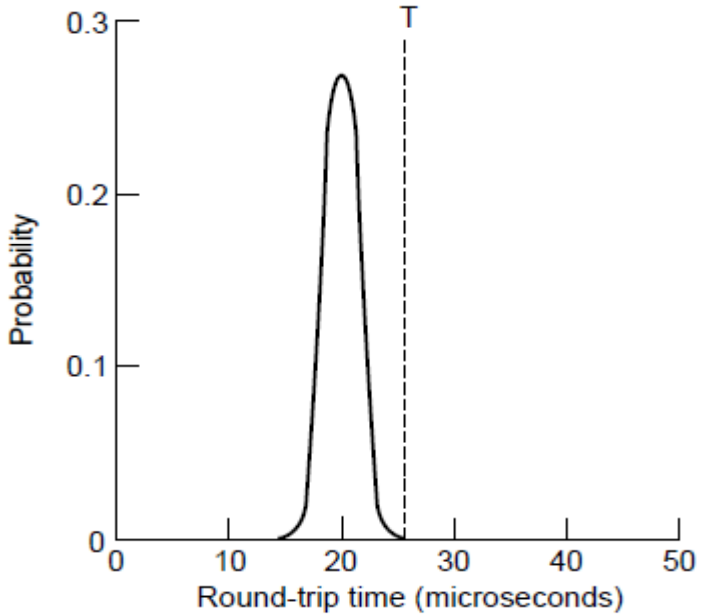


Receiver application reads single bytes, so sender always sends one byte segments

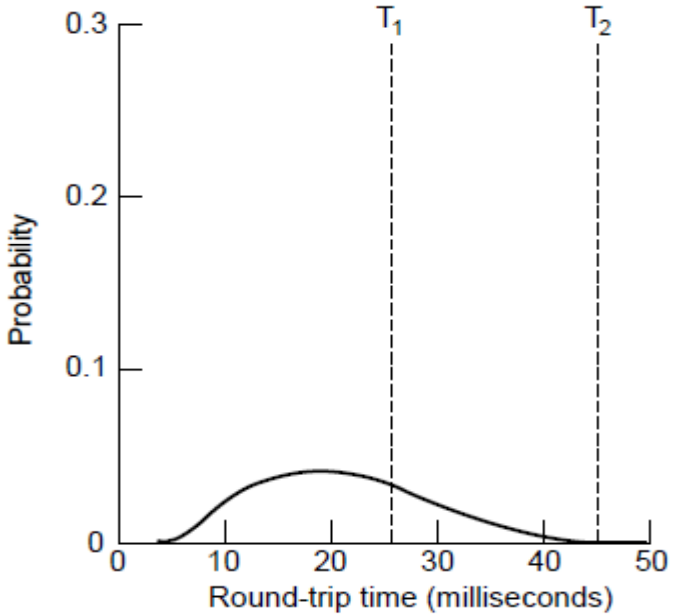
TCP Timer Management

TCP estimates retransmit timer from segment RTTs

- Tracks both average and variance (for Internet case)
- Timeout is set to average plus 4 x variance



LAN case – small, regular RTT



Internet case – large, varied RTT