CS 725/825 & IT 725
Lecture 14
Transport Layer

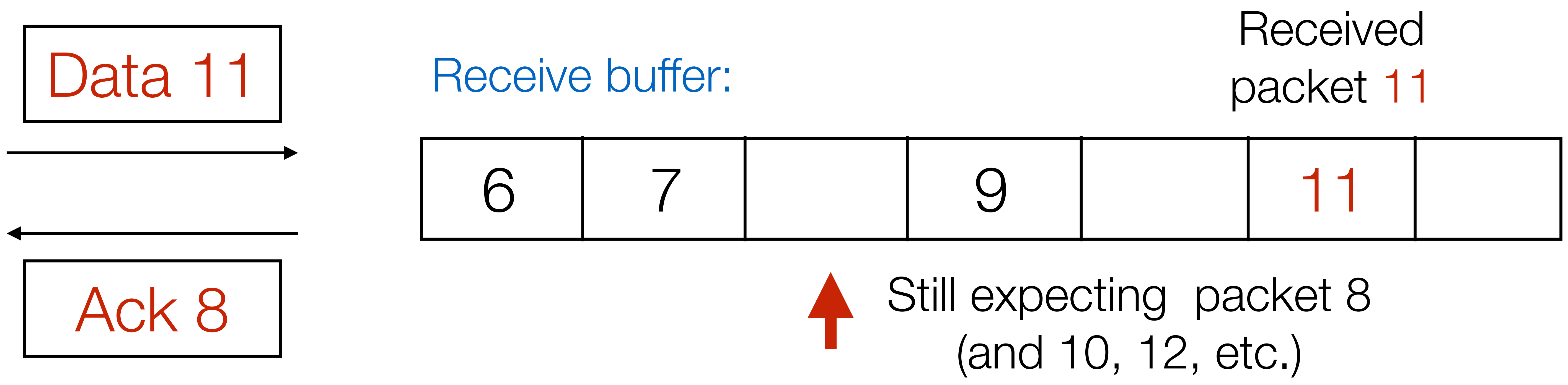
October 16, 2024

Principles of Reliable Transport

- **Goal**: deliver despite unreliability of the network layer or detect that delivery is not possible
- Automatic Repeat reQuest (ARQ):
  - acknowledgment
  - timeout
  - retransmission
  - give up after *k* retransmissions
  - sequence numbers on data packets
  - cumulative acknowledgment numbers

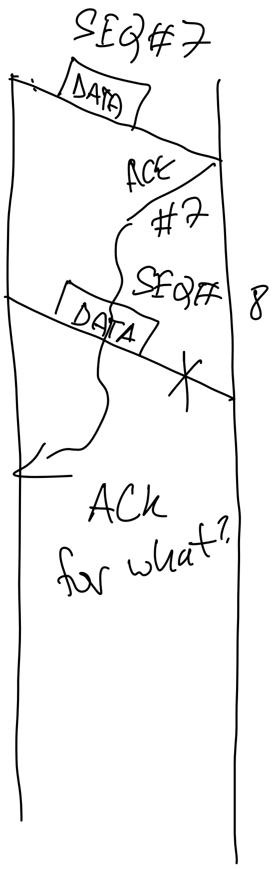
Cumulative ACK numbers

- Method 1 (obvious but not used):
  - ACK carries the sequence number of the packet it acknowledges
- Method 2 (Cumulative ACK, used by TCP)
  - ACK carries the lowest sequence number of the packets that were not yet received (sequence number of the next expected packet)

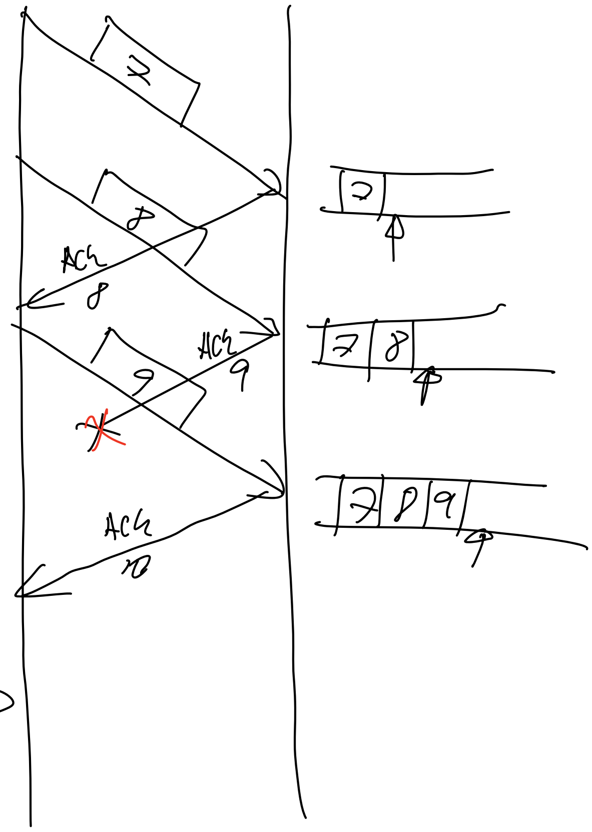


CUMULATIVE ACKS

MOTIVATION



BENEFITS (EX.)



CONFIRMS

7, 8, 9 rec'd

no need

to retr.

Filling the pipe…

- Stop and Wait protocol
  - wait for acknowledgment before sending next packet
- Sliding Window protocols
  - send up to W (window size) packets/bytes before waiting for acknowledgment
  - when a packet is lost:
    - retransmit the packet (*Selective-Reject ARQ*)
    - retransmit all un-acknowledged packets (*Go-Back-N ARQ*)
- Measure: utilization (a.k.a. normalized throughput)
  - the ratio between *goodput* and *maximum theoretical capacity*

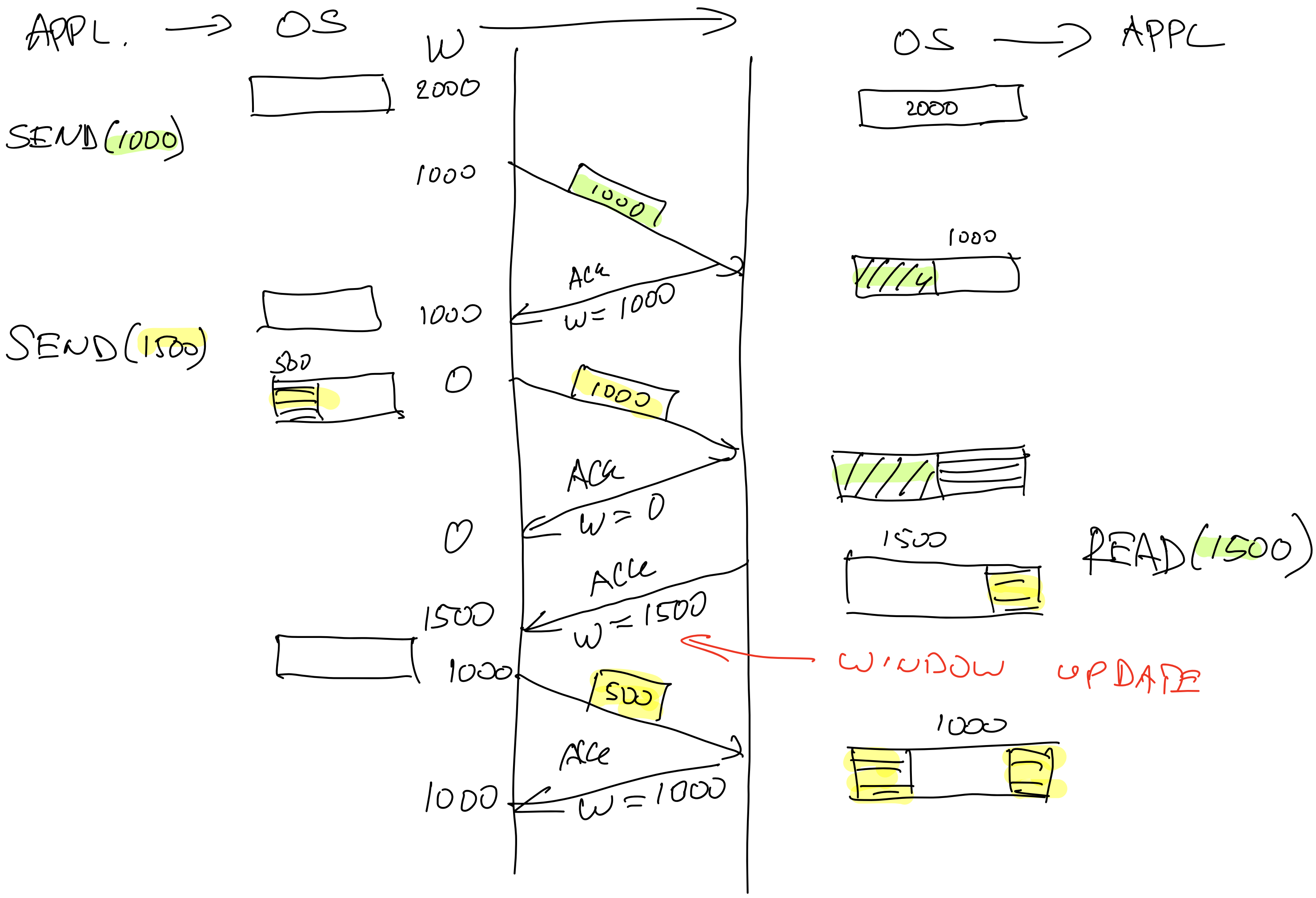
Flow & Congestion Control

- Receiver Congestion
  - receiver is unable to keep up with incoming data
  - solved by explicit feedback from receiver to sender
- Network Congestion
  - nodes or links of the network are overloaded
  - explicit congestion notification (few technologies)
  - implicit congestion notification (Internet)

Congestion control

- Goal: Make the most effective use of the network capacity
  - avoid congestion
  - maximize utilization
  - maintain fairness (or deliver promised service level)
- Method: Controlling the rate with which traffic is injected into the network by the transmitter

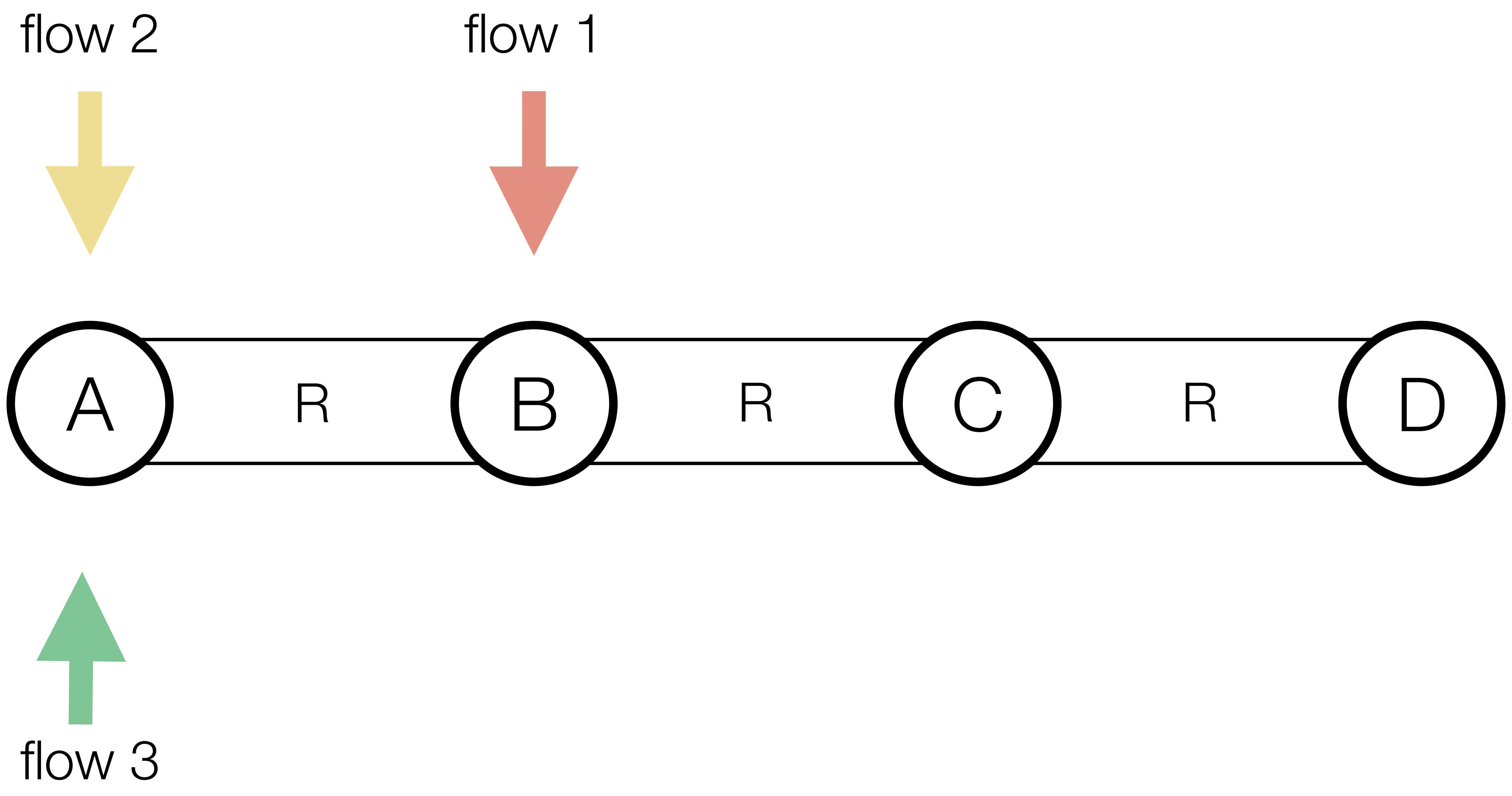
FLOW CONTROL



Congestion control

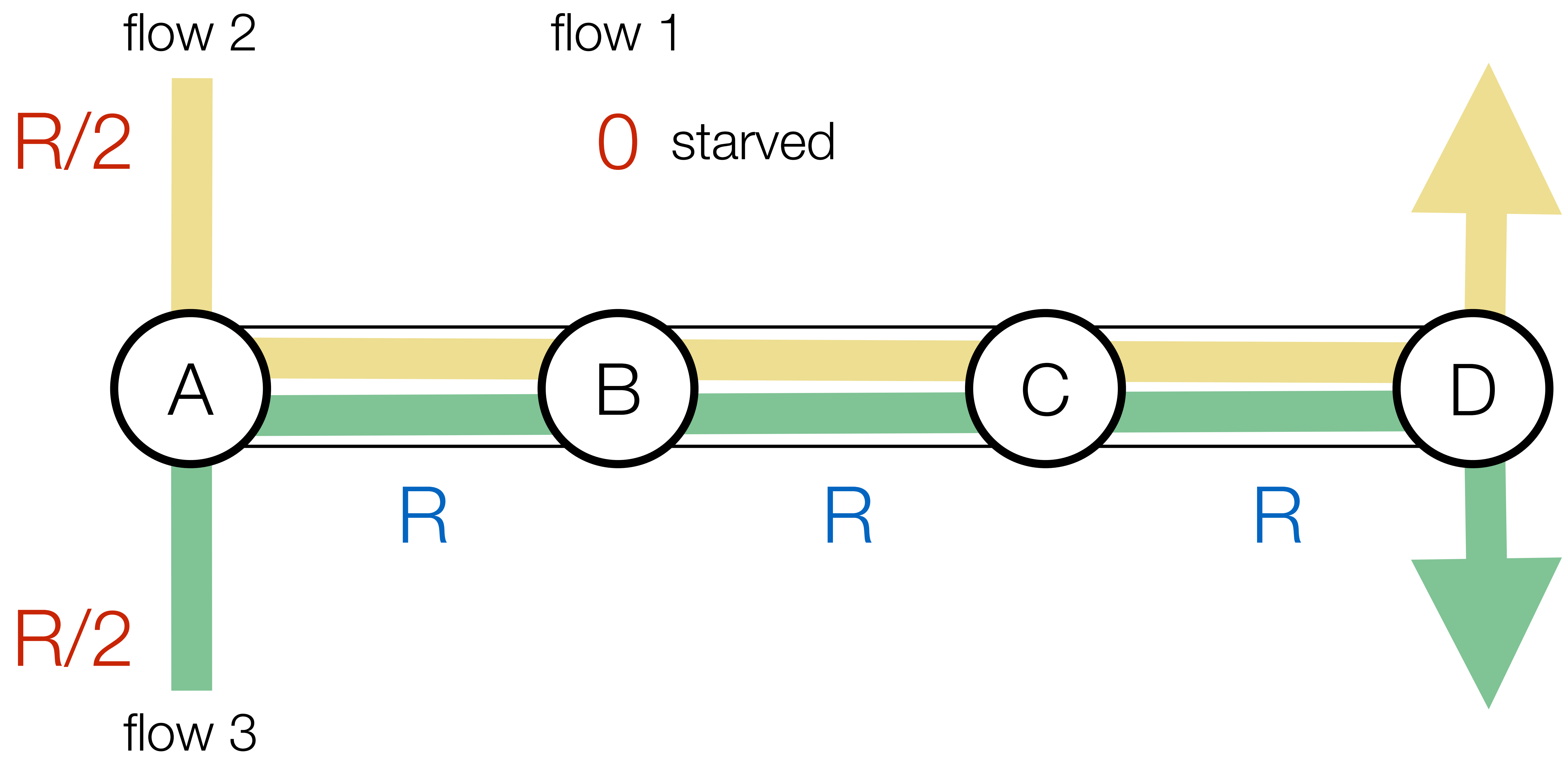
- Reasons why congestion control mechanisms are critical for the stable operation of the Internet [RFC 8085]:
- Prevention of congestion collapse
  - i.e., a state where an increase in network load results in a decrease in useful work done by the network
- Establishment of a degree of fairness
  - i.e., allowing multiple flows to share the capacity of a path reasonably equitably.

Utilization vs fairness



R - link rate

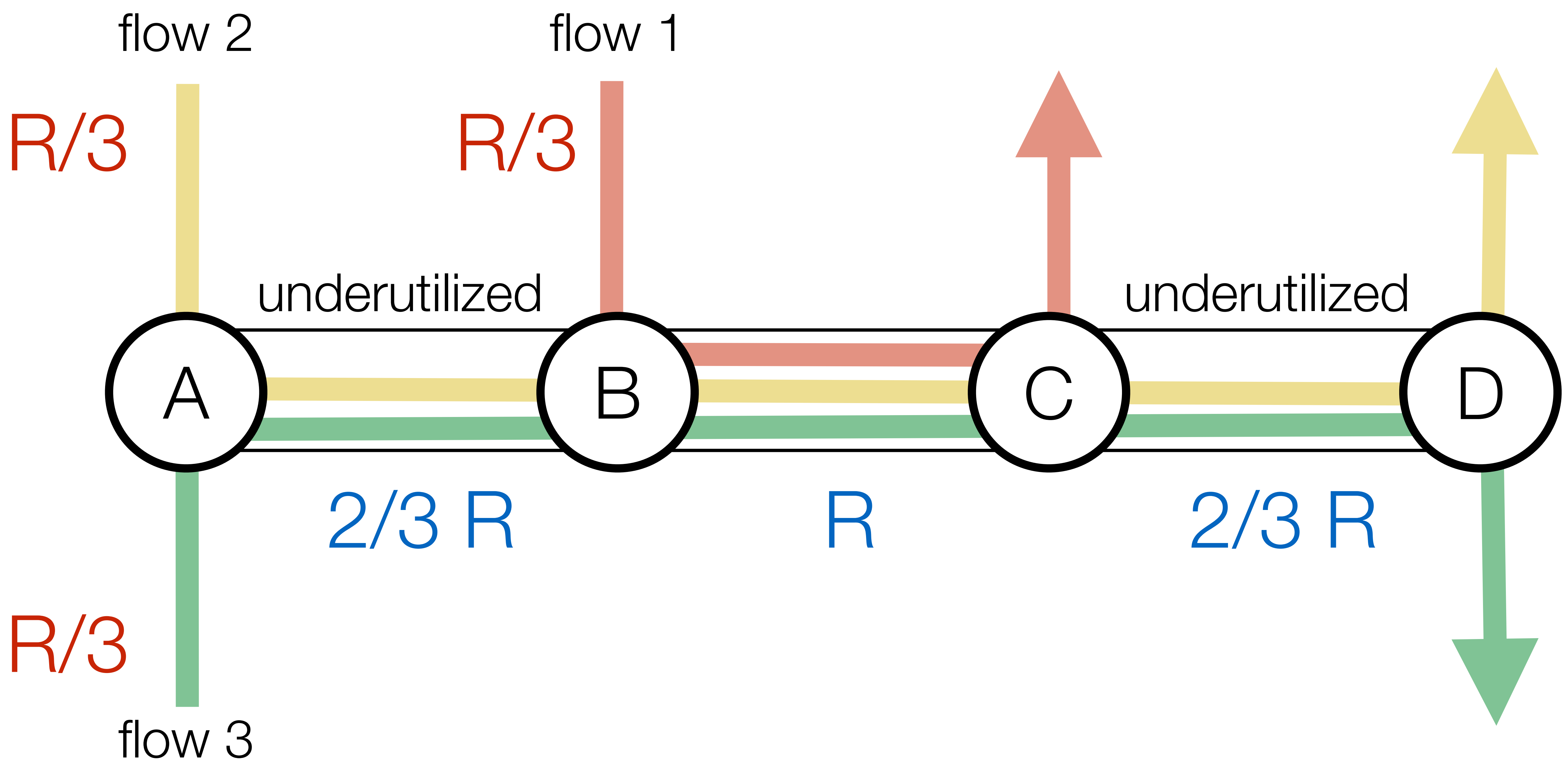
Utilization vs fairness



	Max utilization	Max fairness
flow 1	0	R/3
flow 2	R/2	R/3
flow 3	R/2	R/3

R - link rate

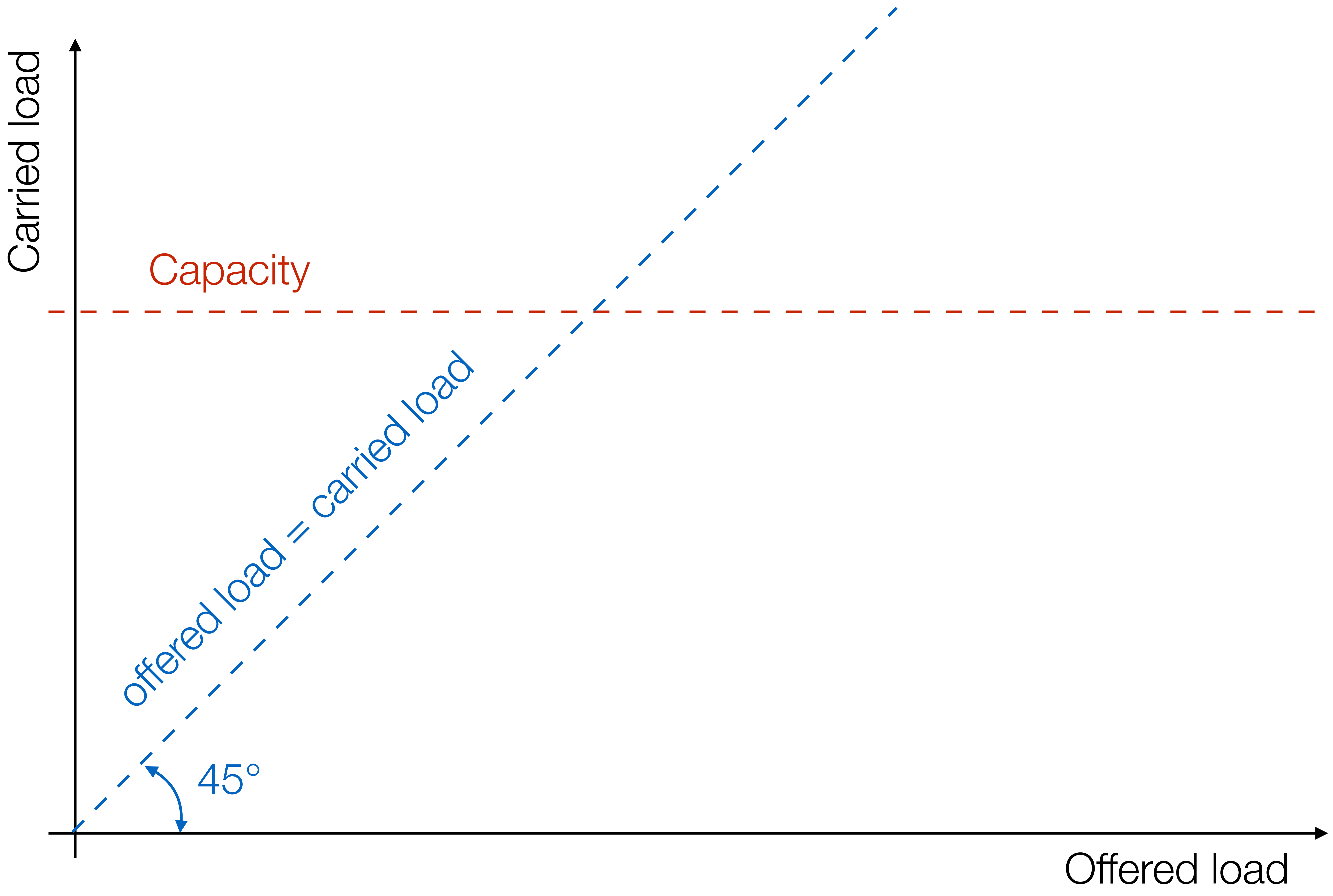
Utilization vs fairness



	Max utilization	Max fairness
flow 1 →	0	R/3
flow 2 →	R/2	R/3
flow 3 →	R/2	R/3

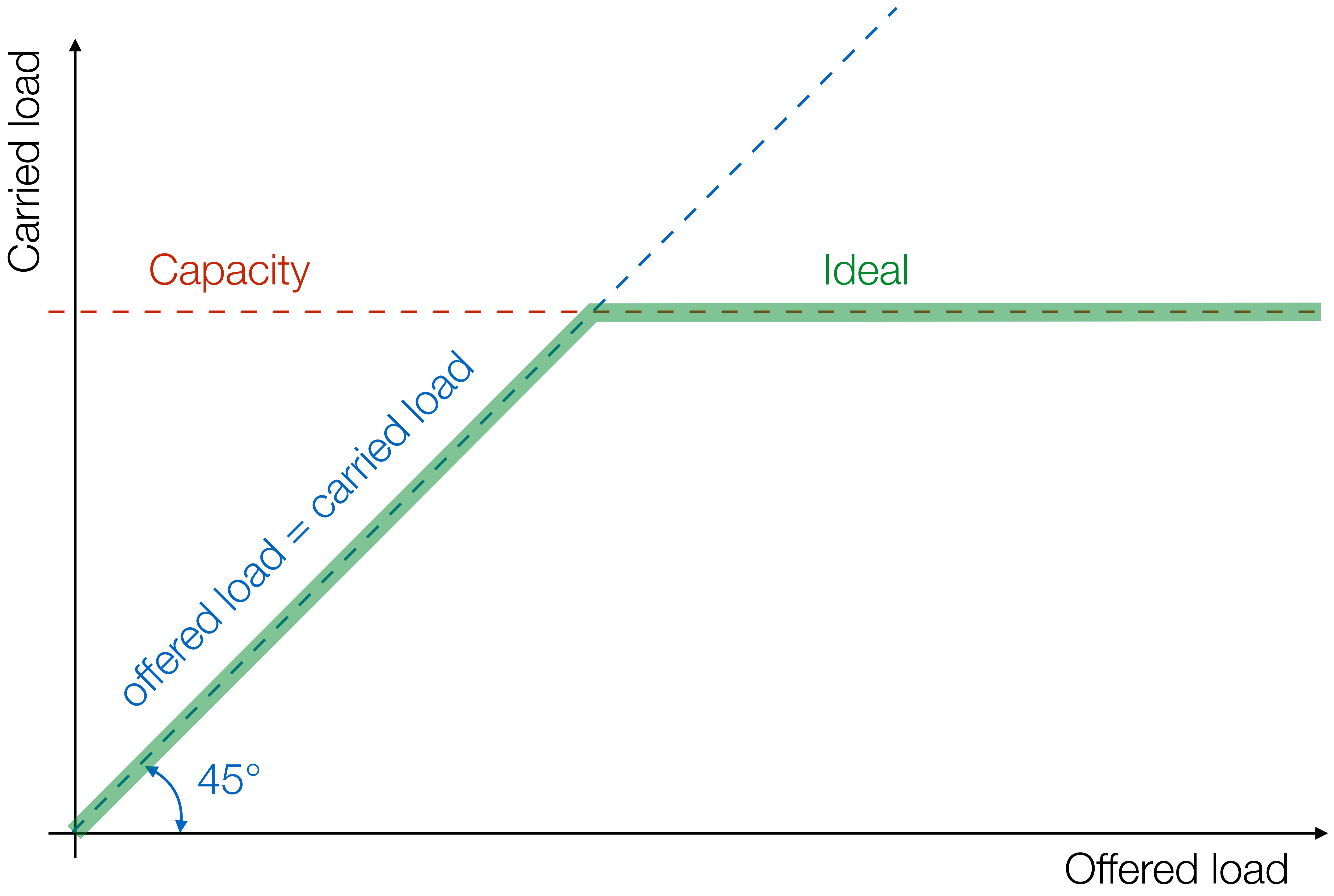
R - link rate

Impact of Congestion

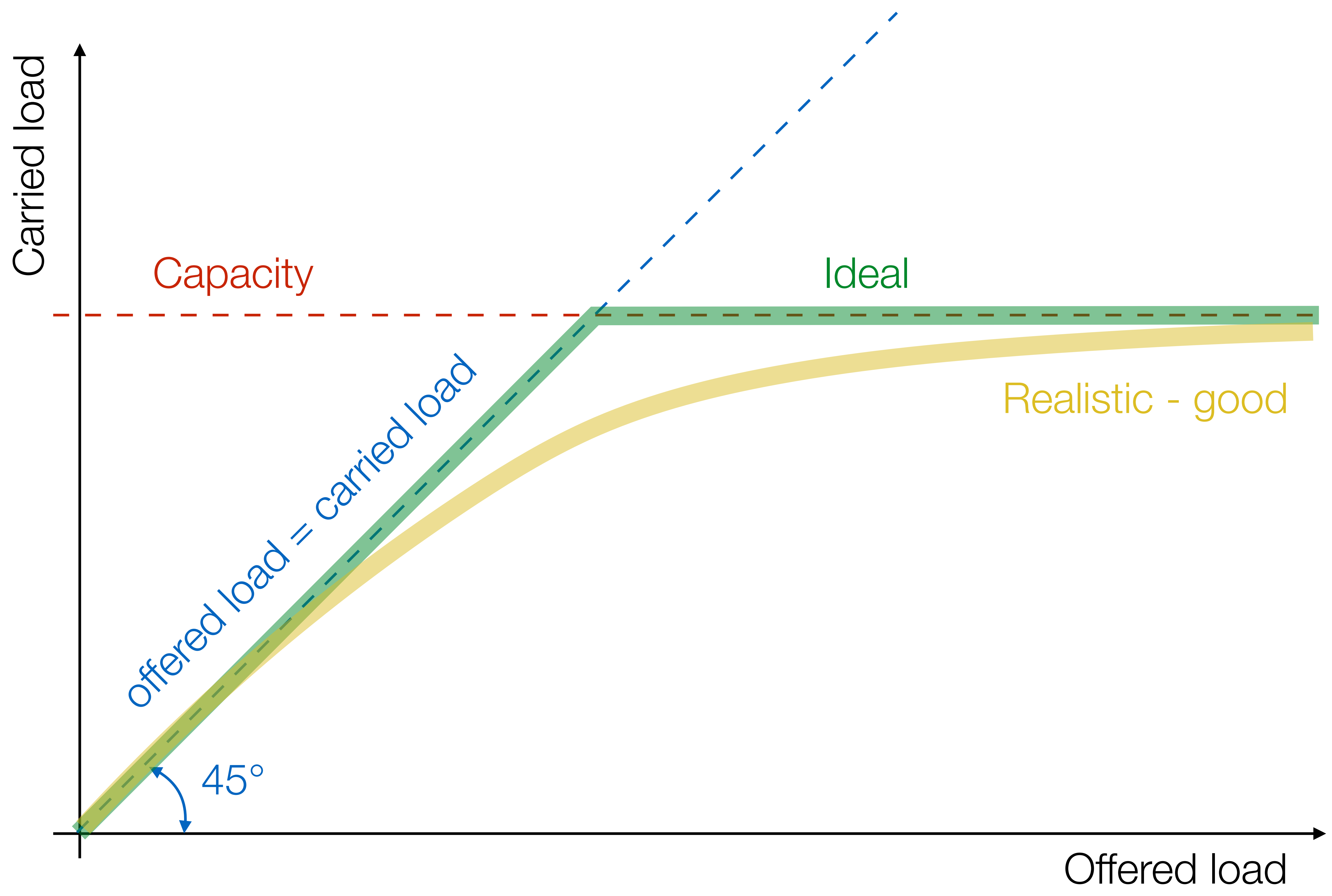


Offered vs carried load graph

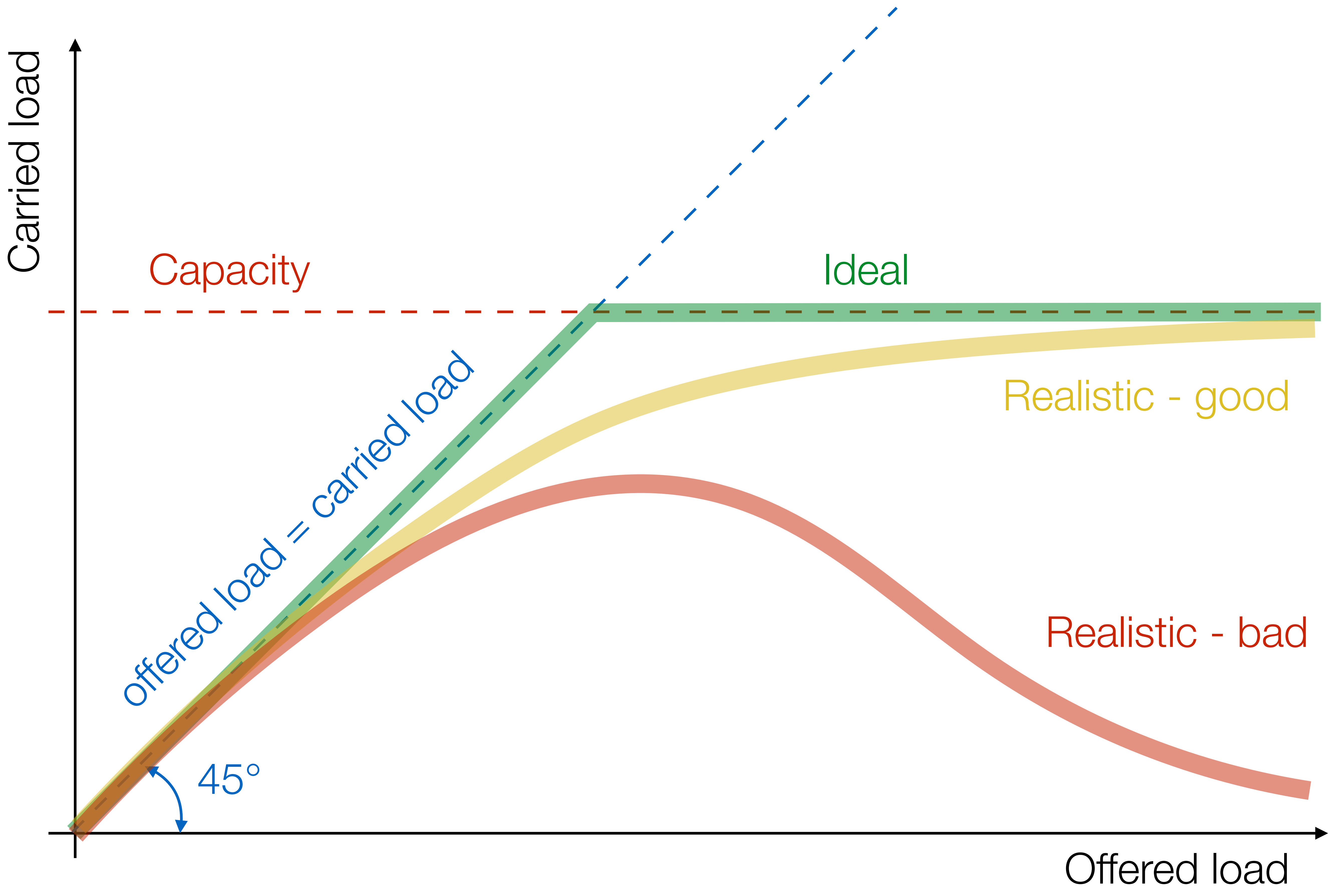
Impact of Congestion



Impact of Congestion



Impact of Congestion



TCP

- Transport Control Protocol
- Design parameters and objectives
  - used by most popular applications, majority of Internet traffic is transported over TCP
  - significant impact on congestion behavior of the Internet
  - must operate over networks with widely-varying characteristics
  - must be robust and (relatively) simple to implement

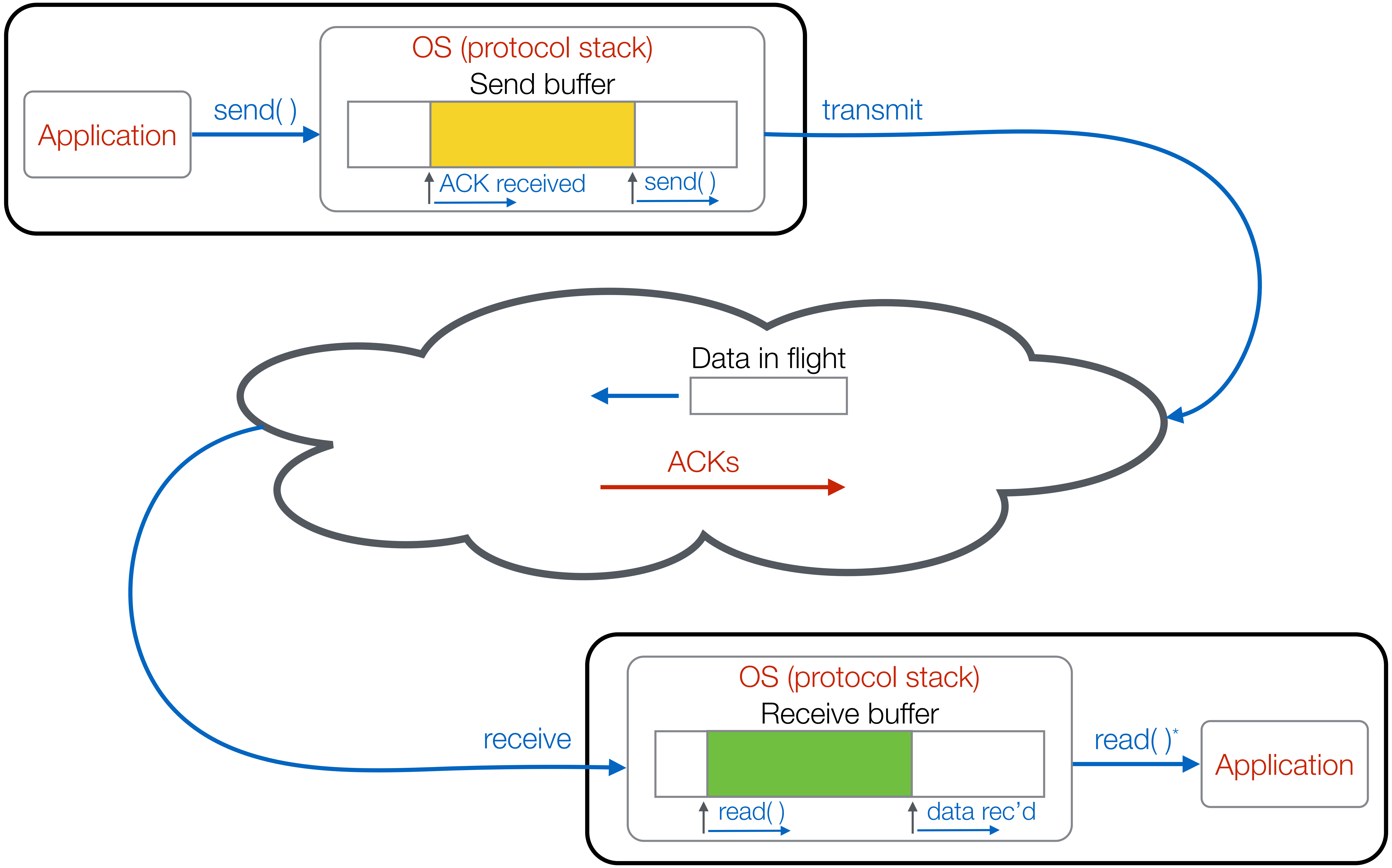
TCP Header

TCP Header

<table>
<tr><th>Offsets</th><th>Octet</th><th colspan="8">0</th><th colspan="8">1</th><th colspan="8">2</th><th colspan="8">3</th></tr>
<tr><th>Octet</th><th>Bit</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><th>0</th><th>0</th><td colspan="16">Source port</td><td colspan="16">Destination port</td></tr>
<tr><th>4</th><th>32</th><td colspan="32">Sequence number</td></tr>
<tr><th>8</th><th>64</th><td colspan="32">Acknowledgment number (if ACK set)</td></tr>
<tr><th>12</th><th>96</th><td colspan="4">Data offset</td><td colspan="3">Reserved
0 0 0</td><td>N
S</td><td>C
W
R</td><td>E
C
E</td><td>U
R
G</td><td>A
C
K</td><td>P
S
H</td><td>R
S
T</td><td>S
Y
N</td><td>F
I
N</td><td colspan="16">Window Size</td></tr>
<tr><th>16</th><th>128</th><td colspan="16">Checksum</td><td colspan="16">Urgent pointer (if URG set)</td></tr>
<tr><th>20
...</th><th>160
...</th><td colspan="32">Options (if Data Offset > 5, padded at the end with "0" bytes if necessary)
...</td></tr>
</table>

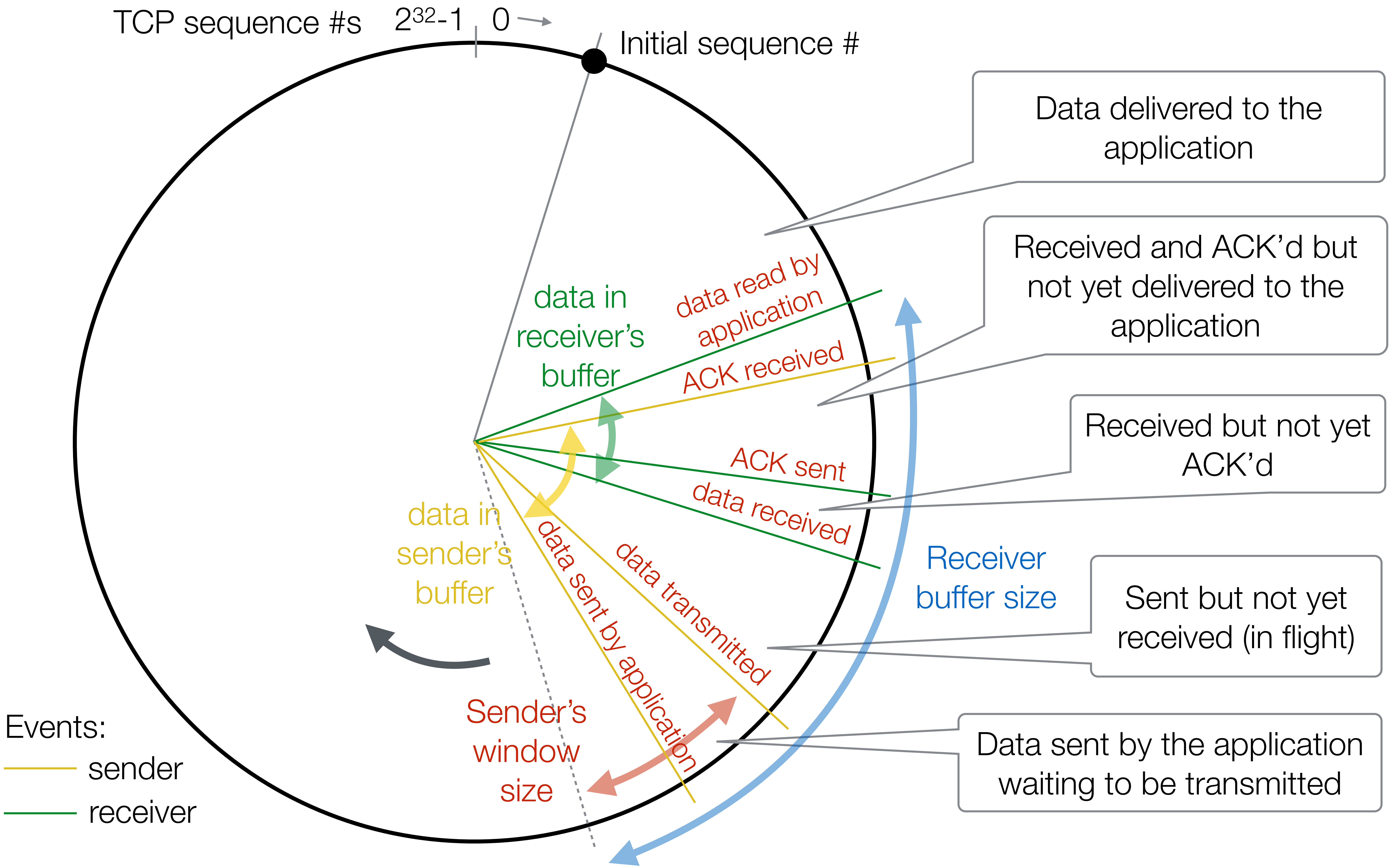
Another image appropriated from Wikipedia...

TCP buffering and data flow

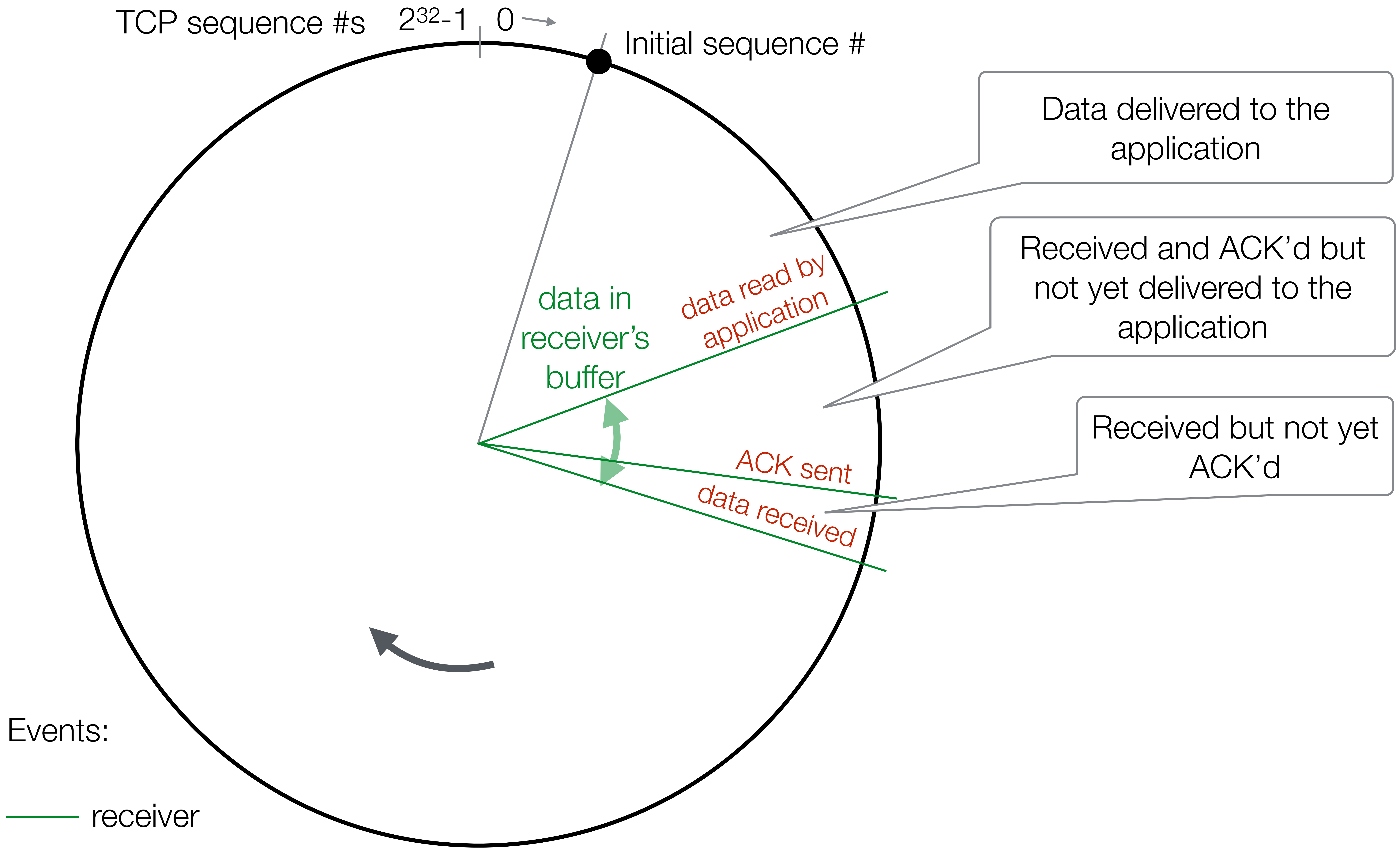


(*) many APIs call the read() operation "receive" (eg: recv()), read is used here to avoid confusion with receiving data on an interface

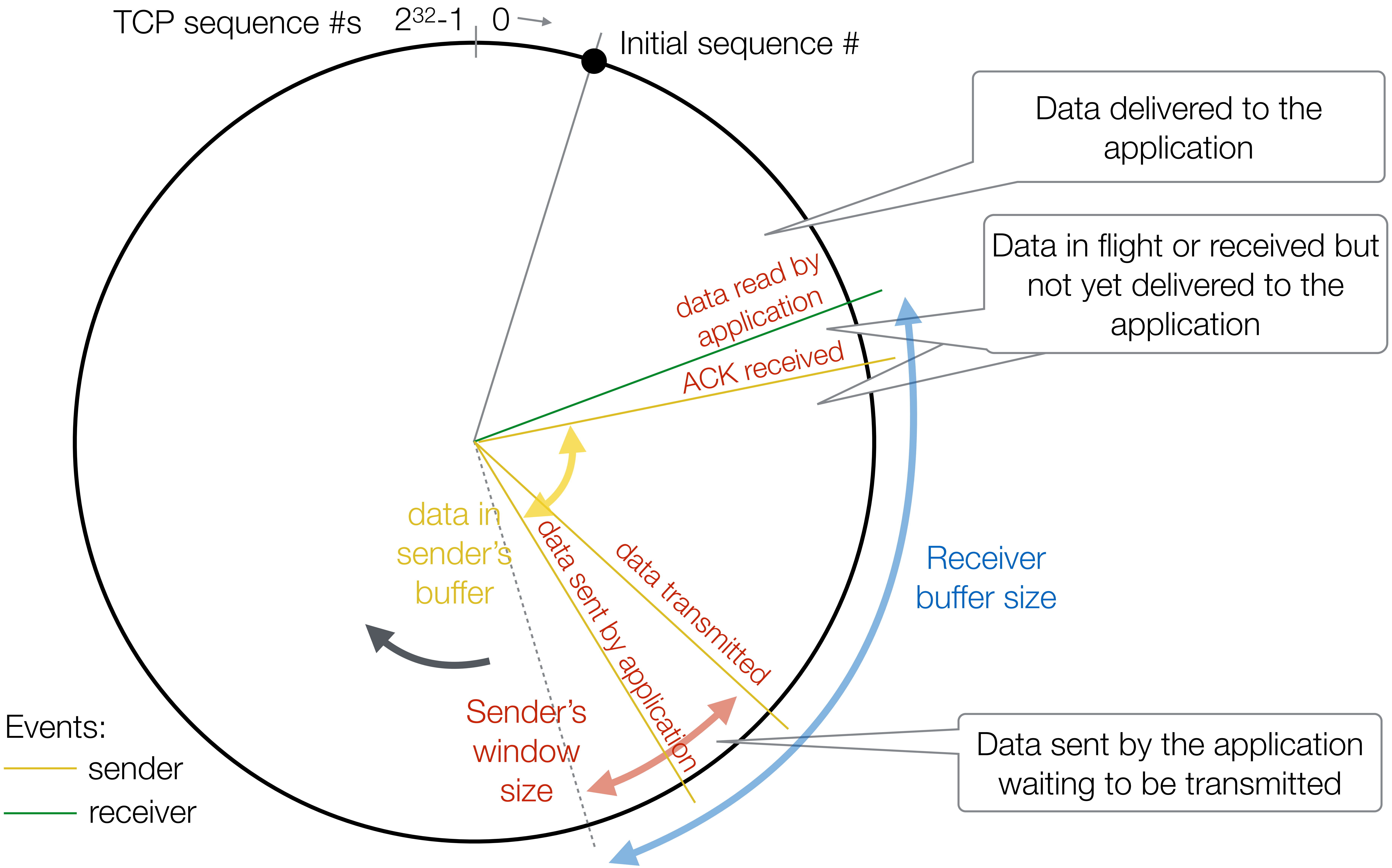
TCP Sliding Window



TCP Sliding Window



TCP Sliding Window



TCP session management

TCP Flags

TCP Header

<table>
<tr><th>Offsets</th><th>Octet</th><th colspan="8">0</th><th colspan="8">1</th><th colspan="8">2</th><th colspan="8">3</th></tr>
<tr><th>Octet</th><th>Bit</th><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><th>0</th><th>0</th><td colspan="16">Source port</td><td colspan="16">Destination port</td></tr>
<tr><th>4</th><th>32</th><td colspan="32">Sequence number</td></tr>
<tr><th>8</th><th>64</th><td colspan="32">Acknowledgment number (if ACK set)</td></tr>
<tr><th>12</th><th>96</th><td colspan="4">Data offset</td><td colspan="3">Reserved
0 0 0</td><td>NS</td><td>CWR</td><td>ECE</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td colspan="16">Window Size</td></tr>
<tr><th>16</th><th>128</th><td colspan="16">Checksum</td><td colspan="16">Urgent pointer (if URG set)</td></tr>
<tr><th>20
...</th><th>160
...</th><td colspan="32">Options (if Data Offset > 5, padded at the end with "0" bytes if necessary)
...</td></tr>
</table>

Another image appropriated from Wikipedia...