Advanced Network Technologies

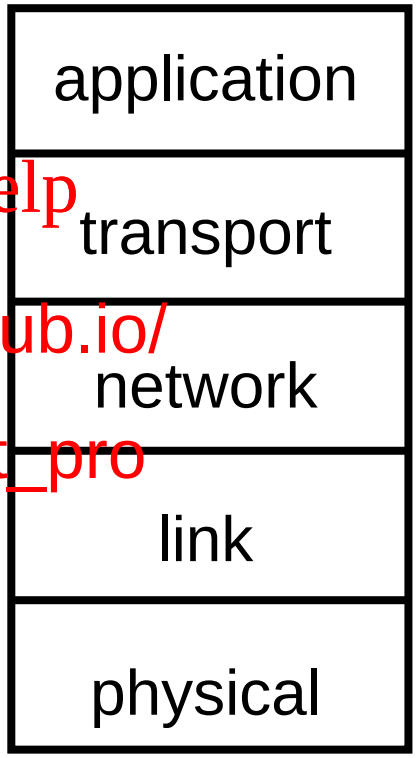
Review

Dr. Wei Bao | Lecturer
School of Computer Science

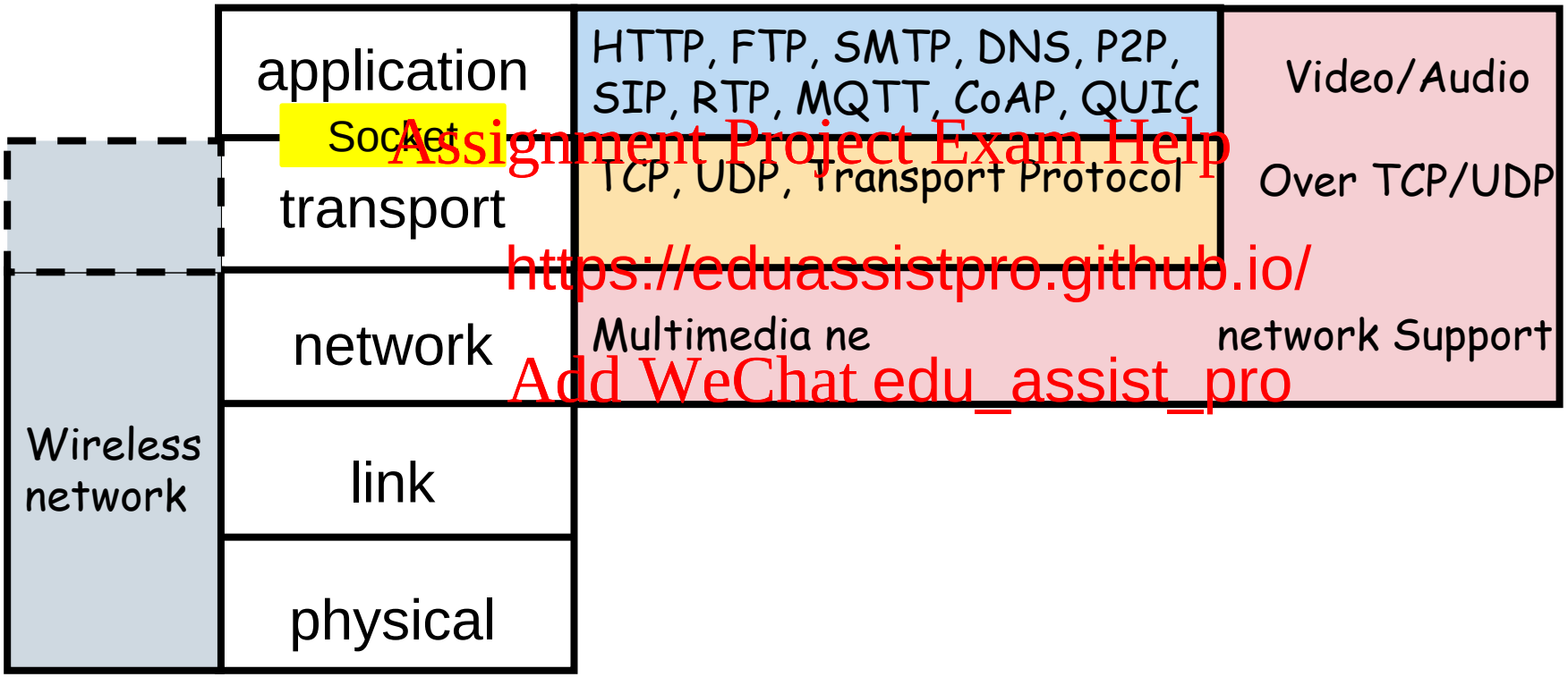
THE UNIVERSITY OF SYDNEY

Protocol Stack

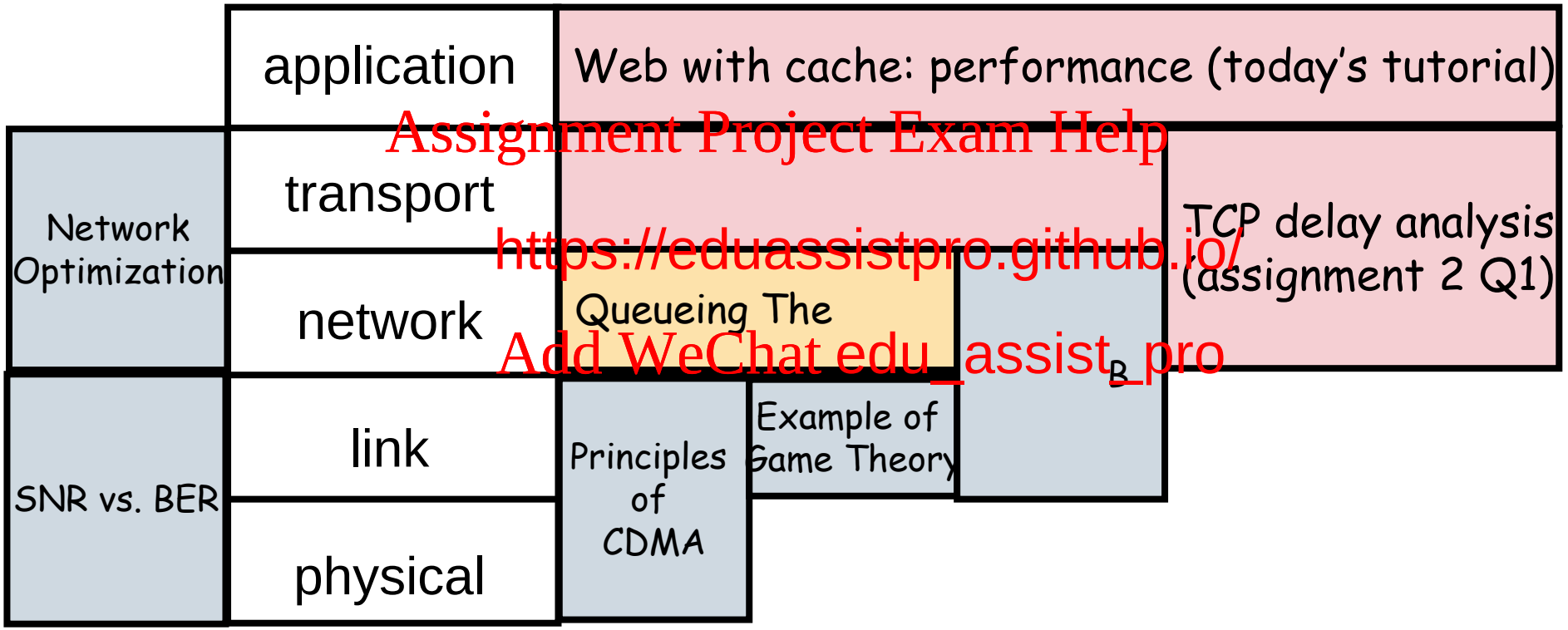
- *application:* supporting network applications
  - FTP, SMTP, HTTP
- *transport:* process-process data transfer
  - TCP, UDP
- *network:* routing of data destination
  - IP, routing protocols
- *link:* data transfer between neighboring network elements
  - Ethernet, 802.11 (WiFi)
- *physical:* bits "on the wire"



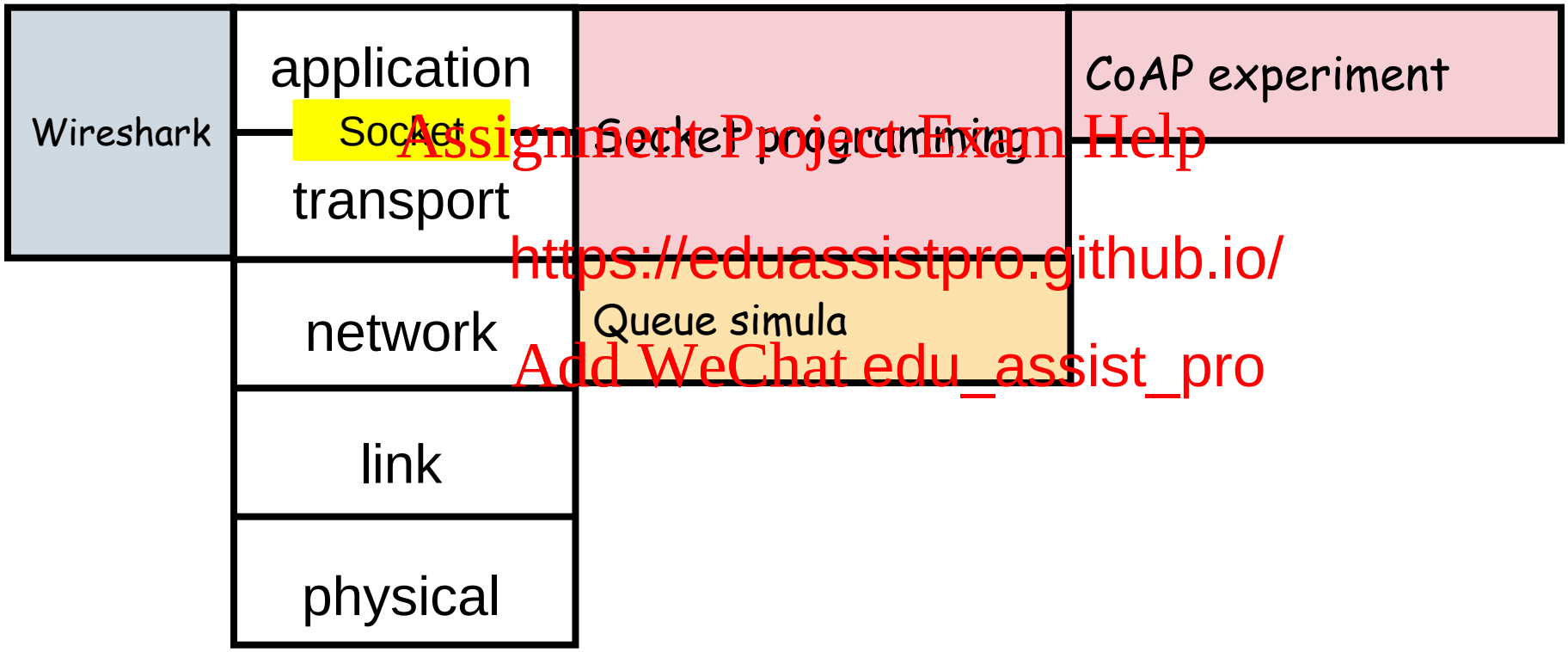
Internet Protocol Stack: Practice



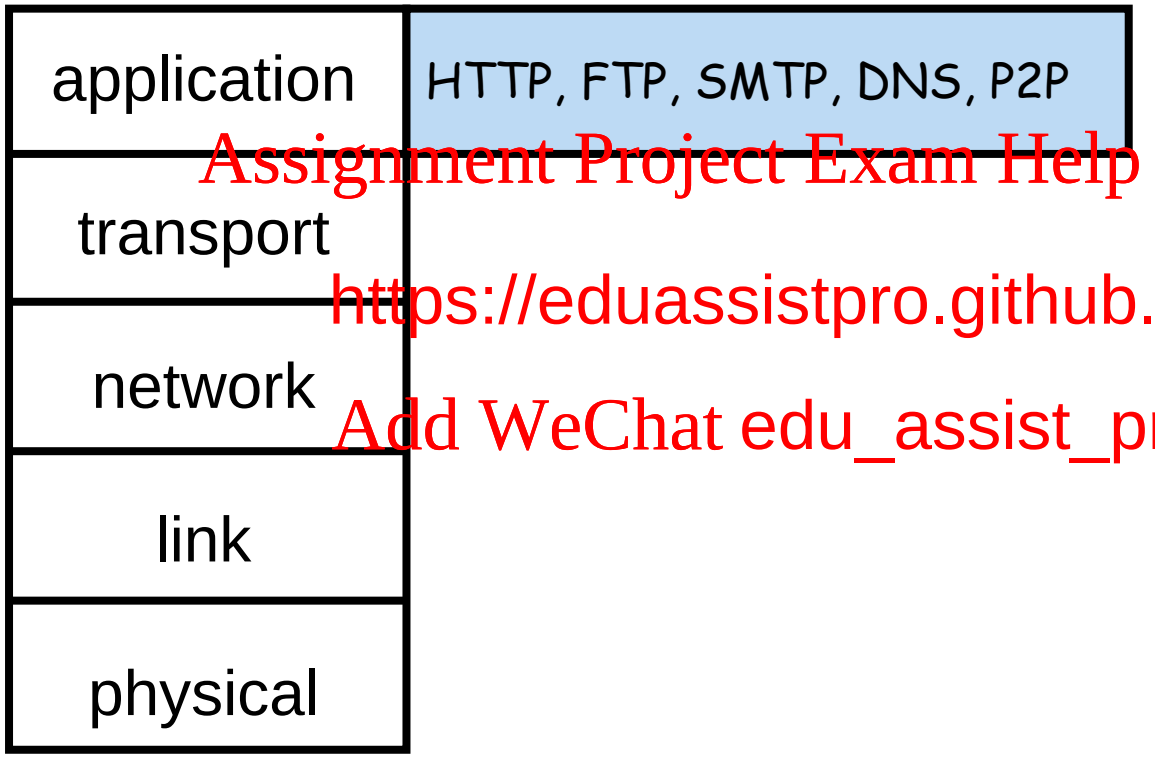
Internet Protocol Stack: Theory



Internet Protocol Stack: Programming/Experiment



Internet Protocol Stack



HTTP: hypertext transfer protocol

- Web's application layer protocol
- client/server mod
  - *client:* browser t requests, receives, (using HTTP protocol) and "displays" Web objects
  - *server:* Web server sends (using HTTP protocol) objects in response to requests

HTTP connections

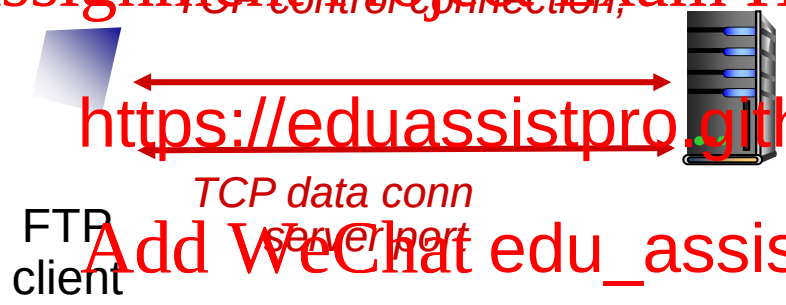
non-persistent HTTP

- at most one object sent over TCP connection
  - connection th
- downloading multiple objects required multiple connections

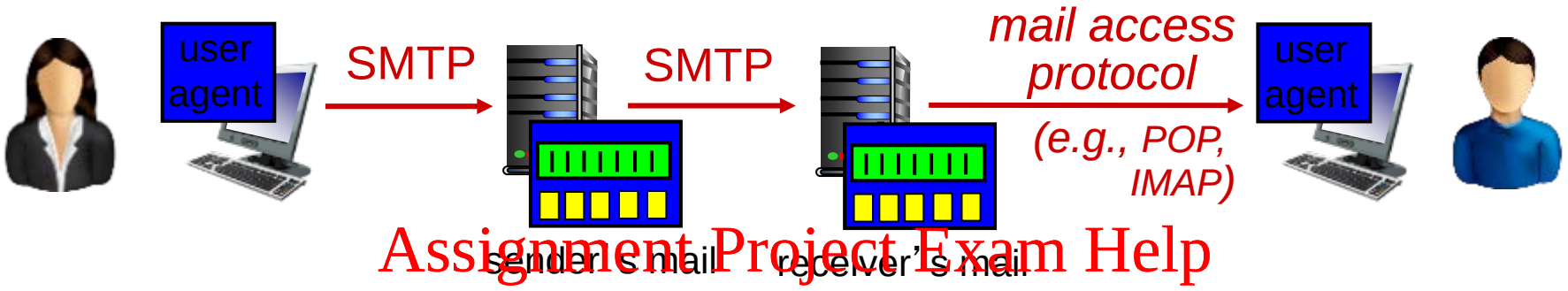
persistent HTTP

- multiple objects can be sent over single TCP
  n between
  c er

FTP: the file transfer protocol



Email

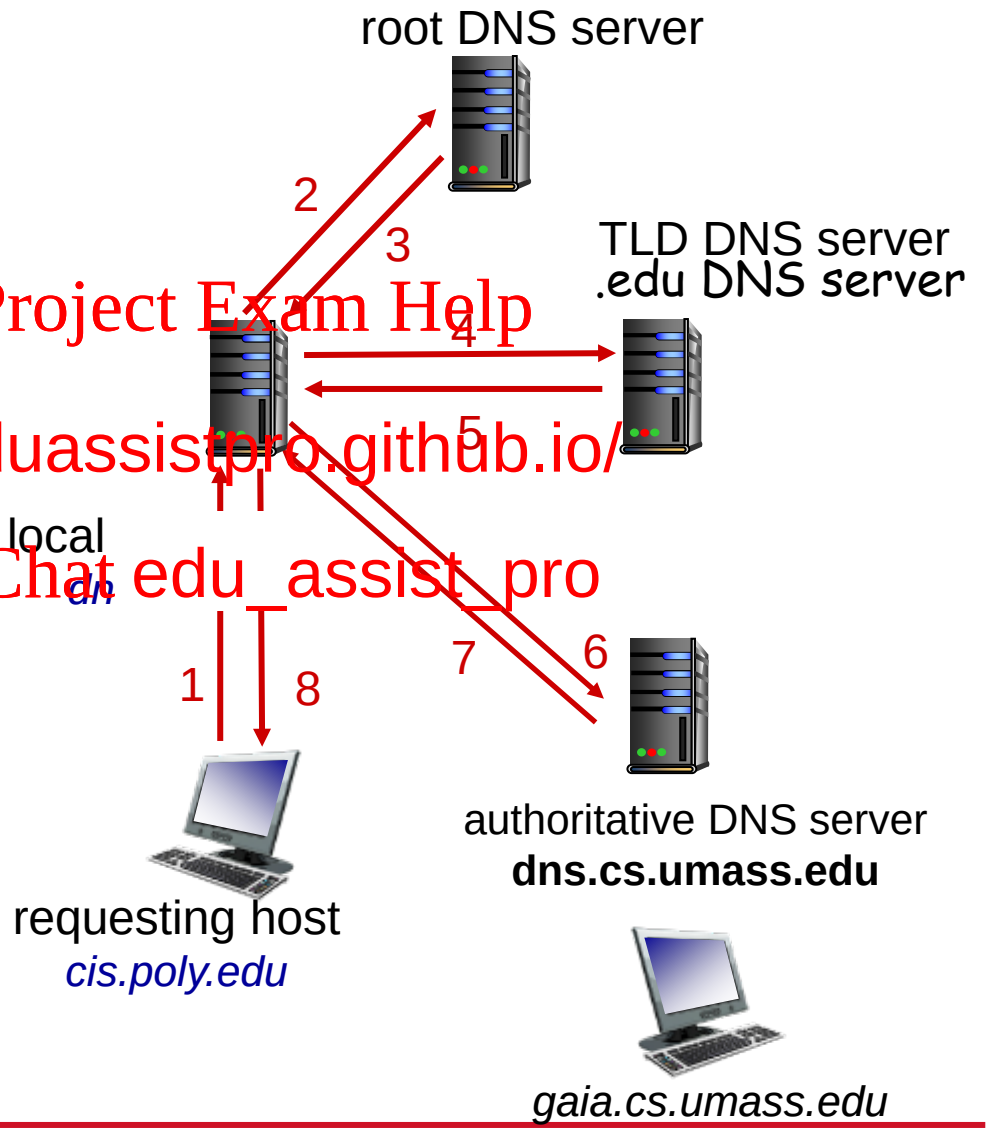


- SMTP: delivery/stora
- mail access protocol: retrieval from ser
  - POP: Post Office Protocol [RFC 1939]: download
  - IMAP: Internet Mail Access Protocol [RFC 1730]: more features, including manipulation of stored msgs on server
  - HTTP: Using a browser to access a webmail `https://webmail.sydney.edu.au`

› host at cis.poly.edu wants IP address for gaia.cs.umass.edu

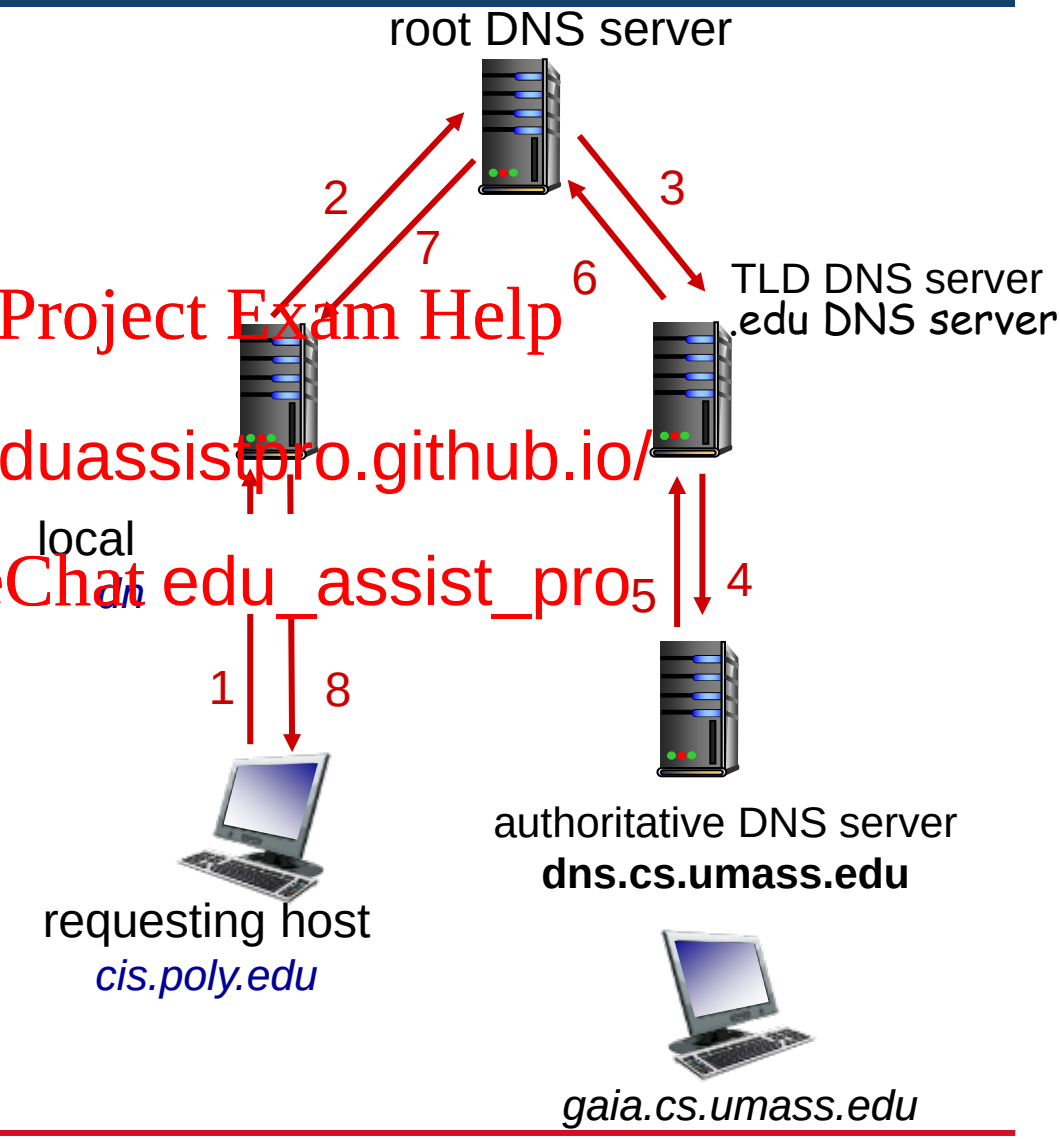
iterated query:

- contacted server replies with name of server to contact
- "I don't know this name, but ask this server"

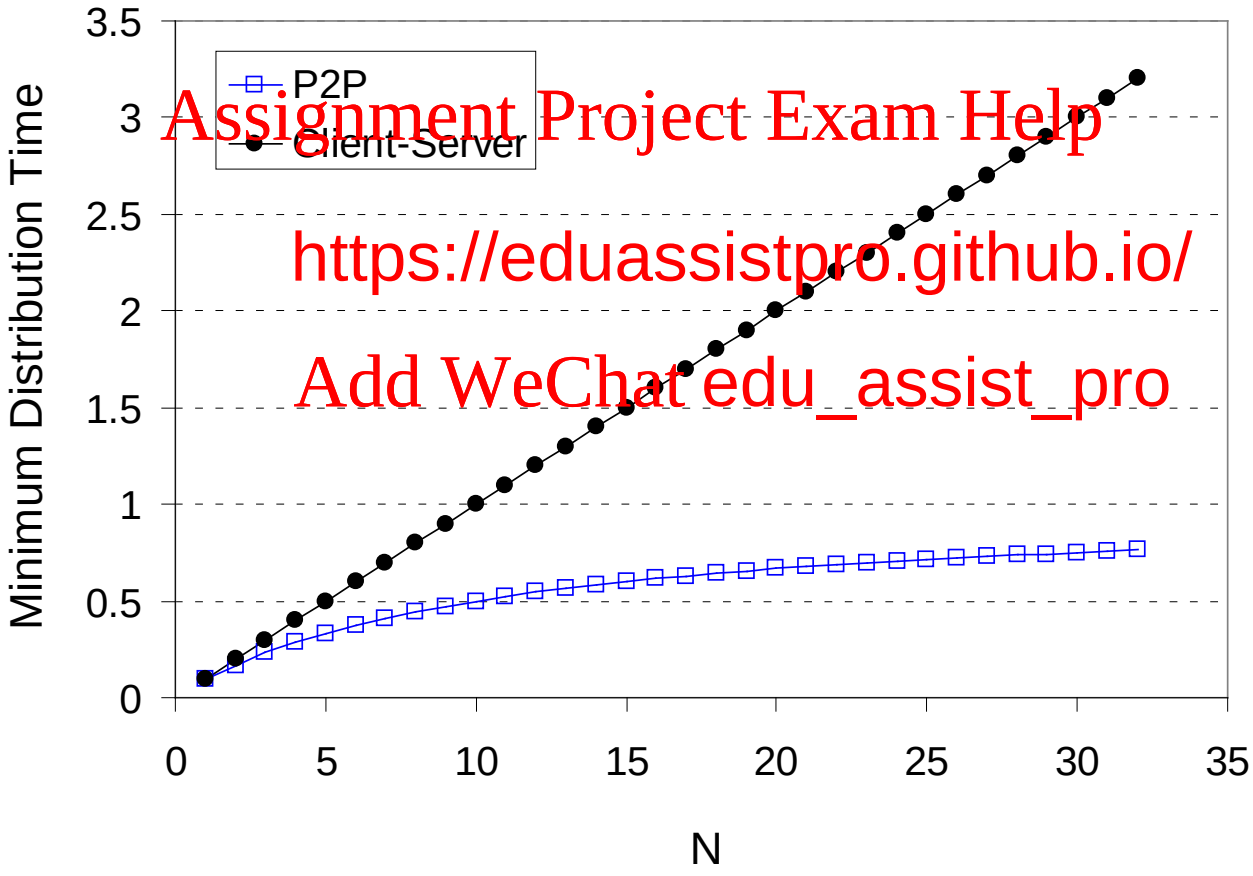


recursive query:

- puts burden of name resolution on contacted name server
- heavy load at upper levels of hierarchy?



P2P Principle



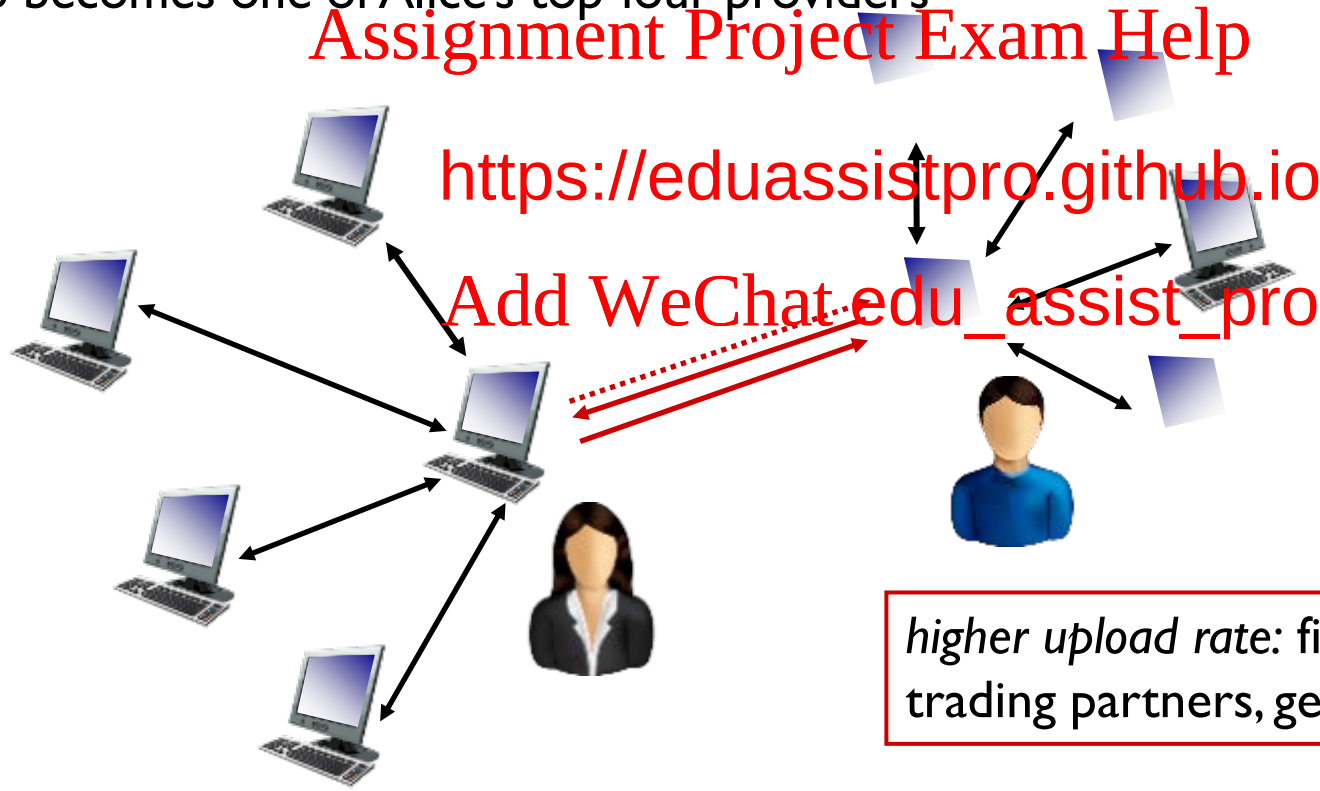
BitTorrent: tit-for-tat

(1) Alice sends chunks to those four peers currently sending her chunks at highest rate

(2) Alice randomly unchokes Bob

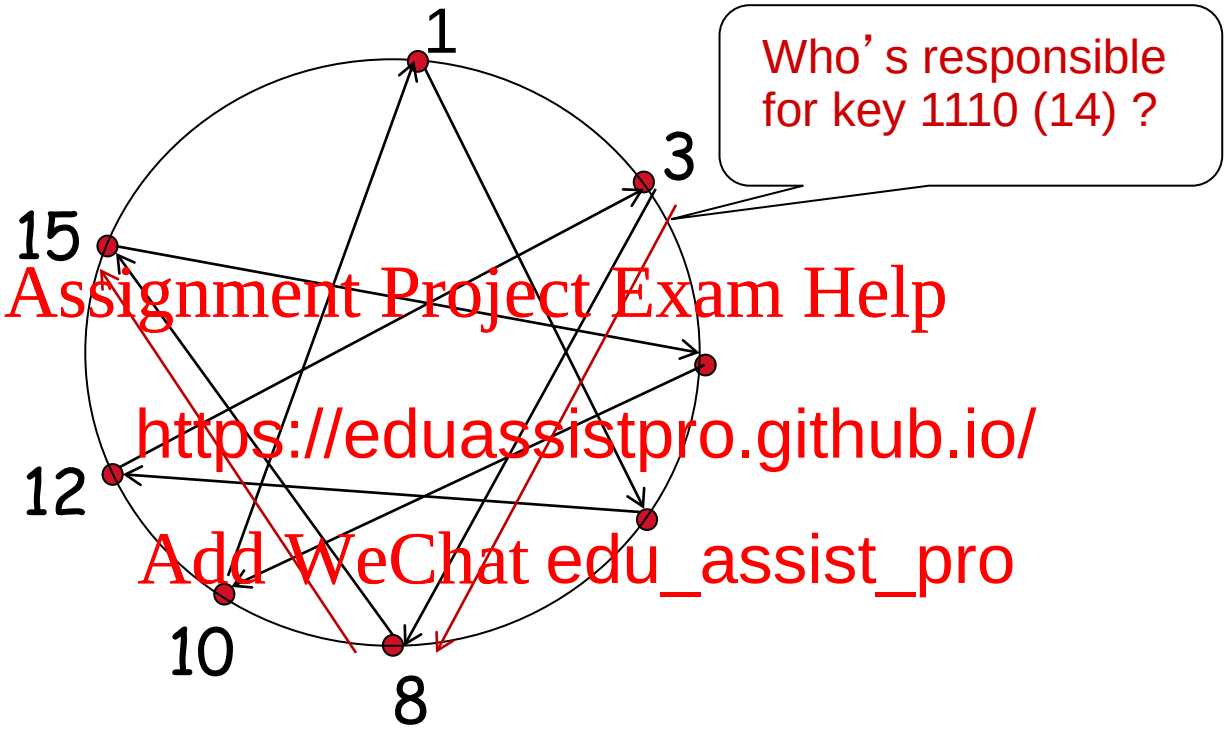
(3) Alice becomes one of Bob's top-four providers;

(4) Bob becomes one of Alice's top-four providers



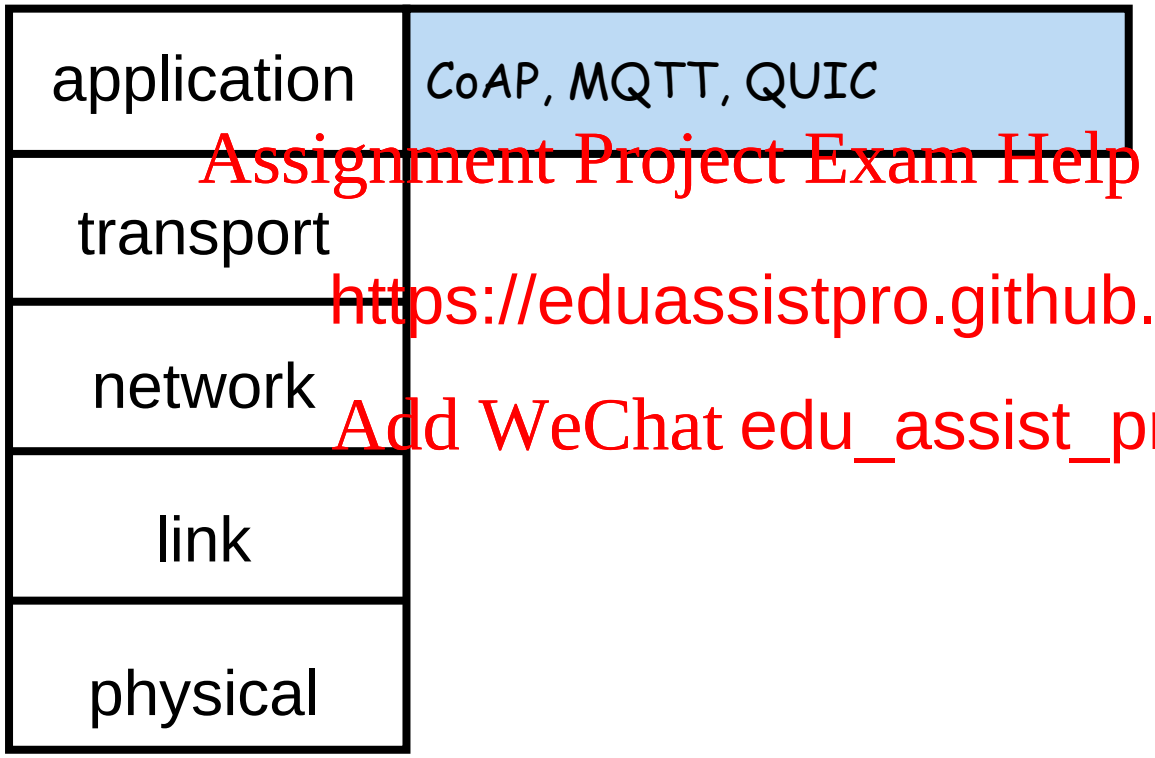
higher upload rate: find better trading partners, get file faster !

Circular DHT with shortcuts



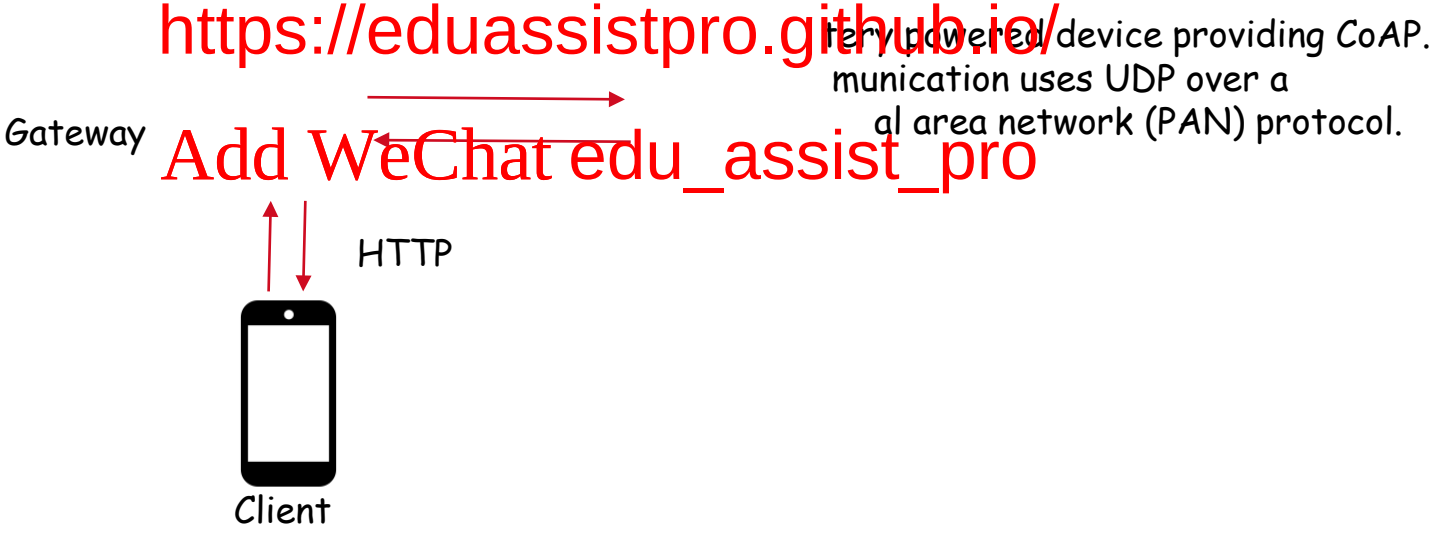
› each peer keeps track of predecessor, successor, short cuts.

Internet Protocol Stack



CoAP

- CoAP provides a request/response interaction like HTTP.
- Over UDP.
- GET, PUT, observe



MQTT

- MQTT: Lightweight, publish-subscribe network protocol that transports messages between devices.
- Runs over TCP
- Two types of entities:
  - Broker: se ards messages.
  - Client: de er

https://mqtt.org/

Over UDP

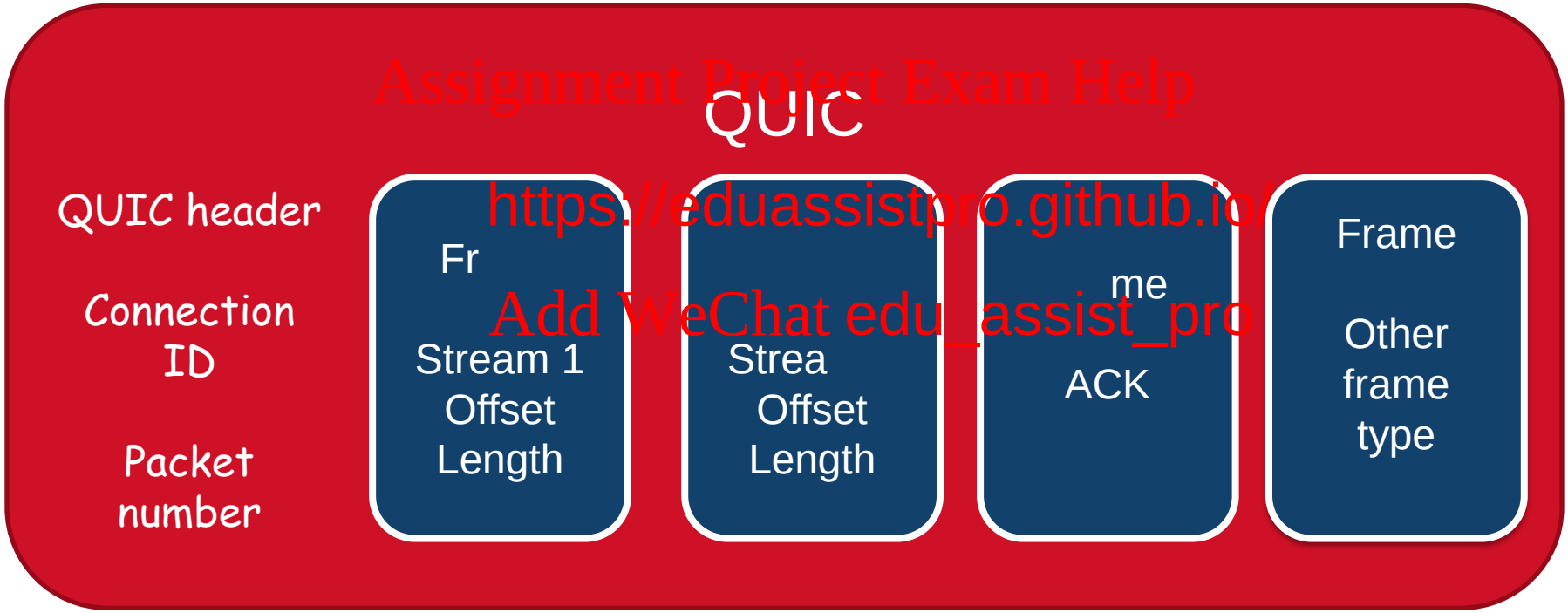
Avoid head-of-line blocking.

DP does not care about
rdering of packet and if
ket get lost.

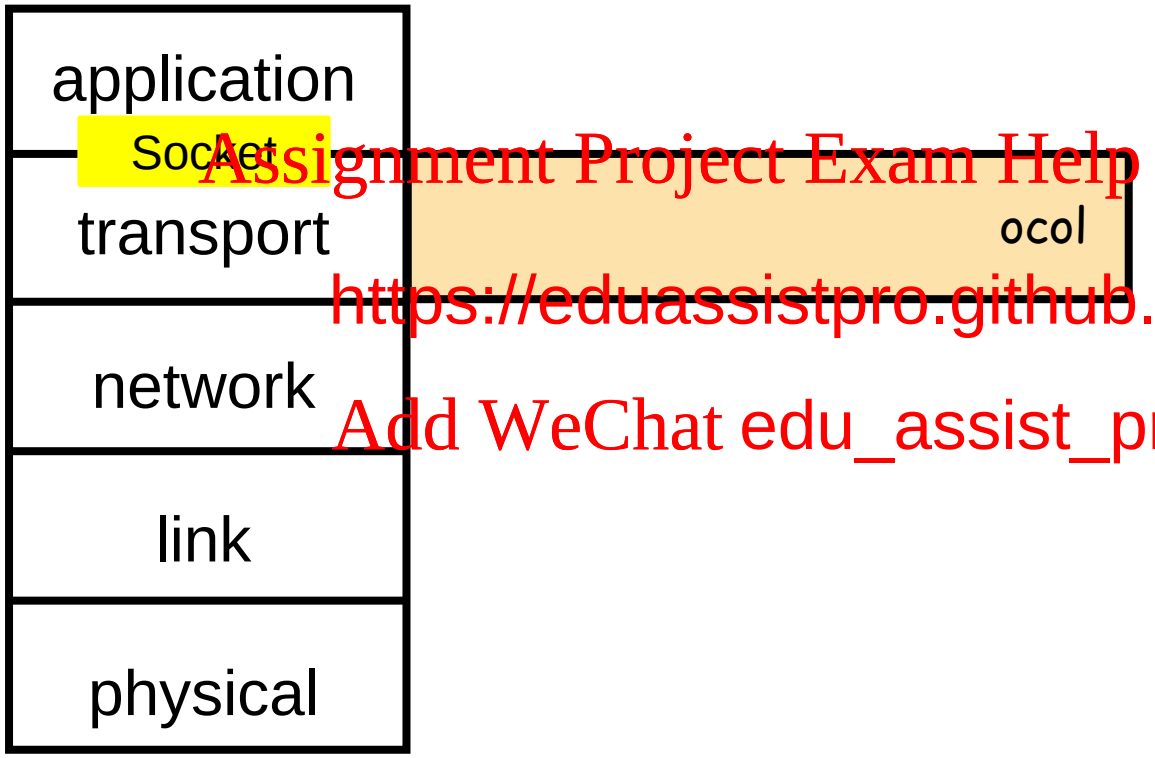
QUIC is solving this issue and it will take care of packet lost in particular stream.

https://medium.com/faun/http-2-spdy-and-http-3-quic-bae7d9a3d484

Connection ID



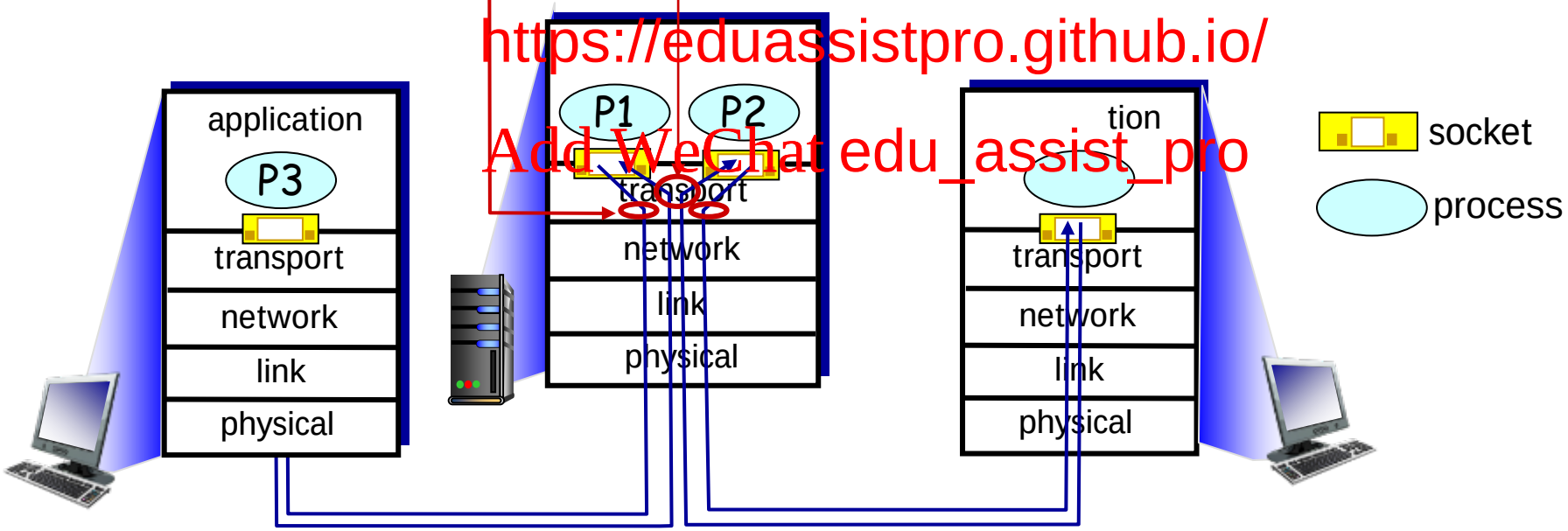
Protocol Stack

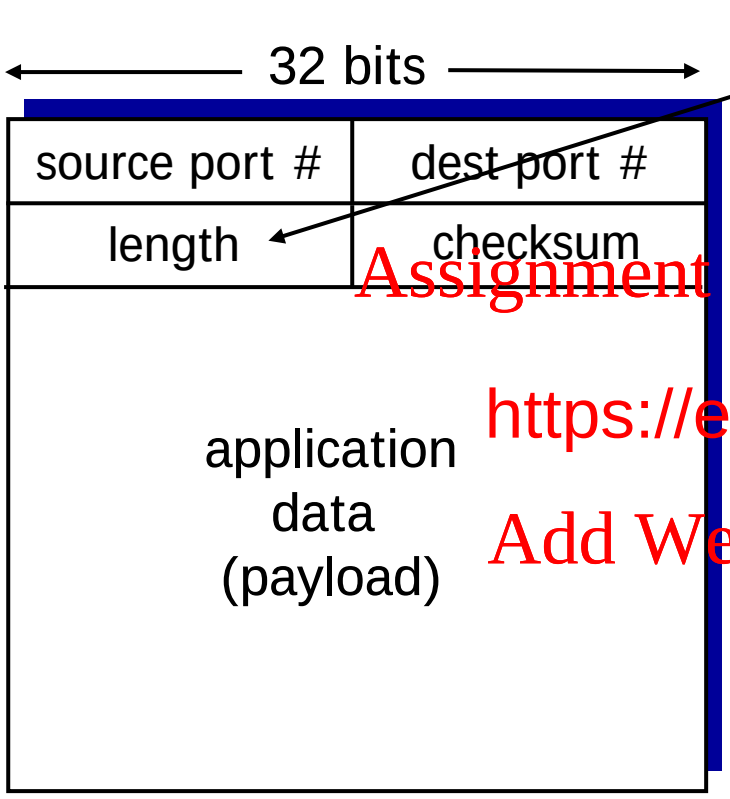


Multiplexing/demultiplexing

multiplexing at sender:
handle data from multiple sockets, add transport header (later used for demultiplexing)

demultiplexing at receiver:
use header info to deliver received segments to correct socket





length, in bytes of UDP segment, including header

UDP segment format

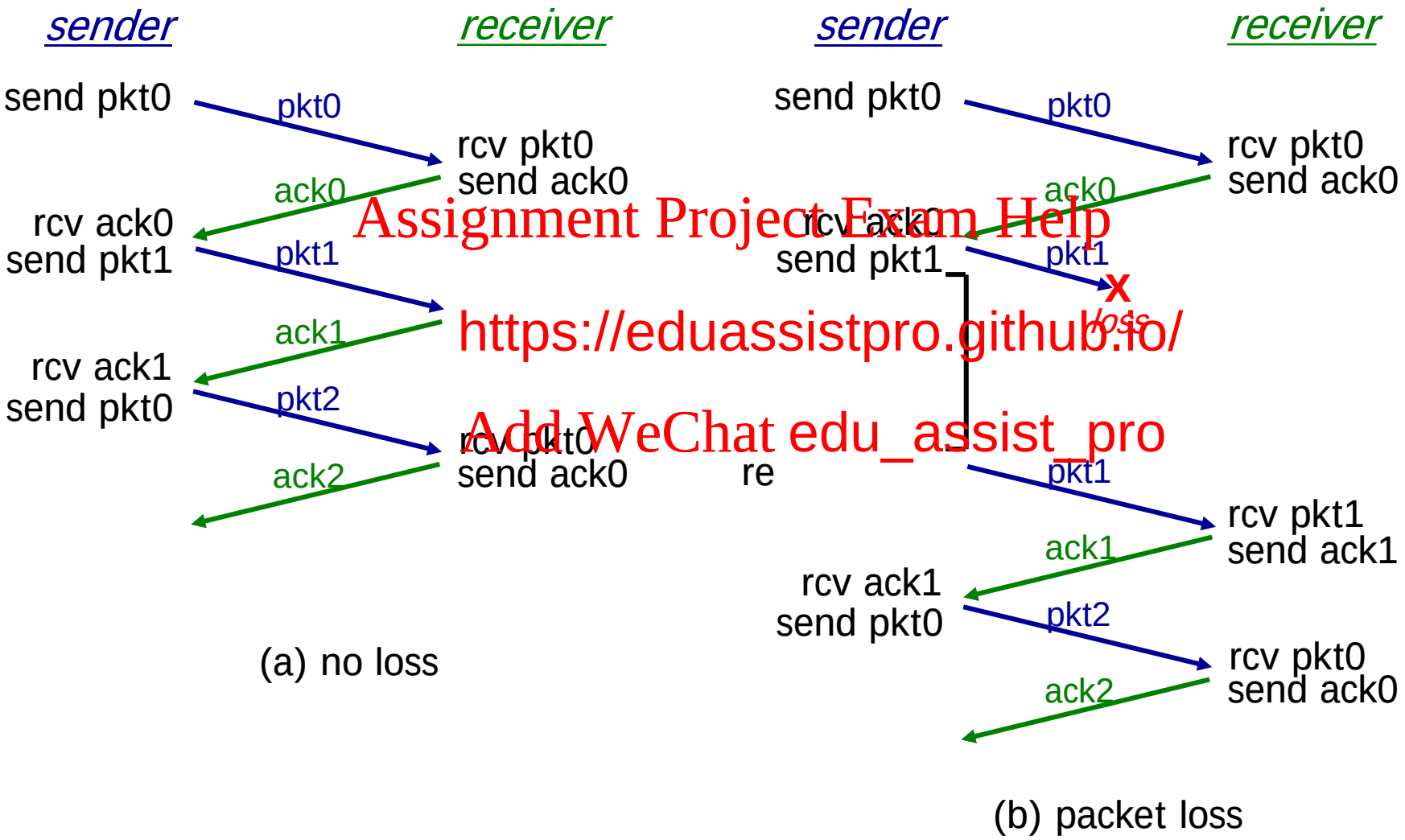
Why is there a UDP?

ction establishment
n add delay)

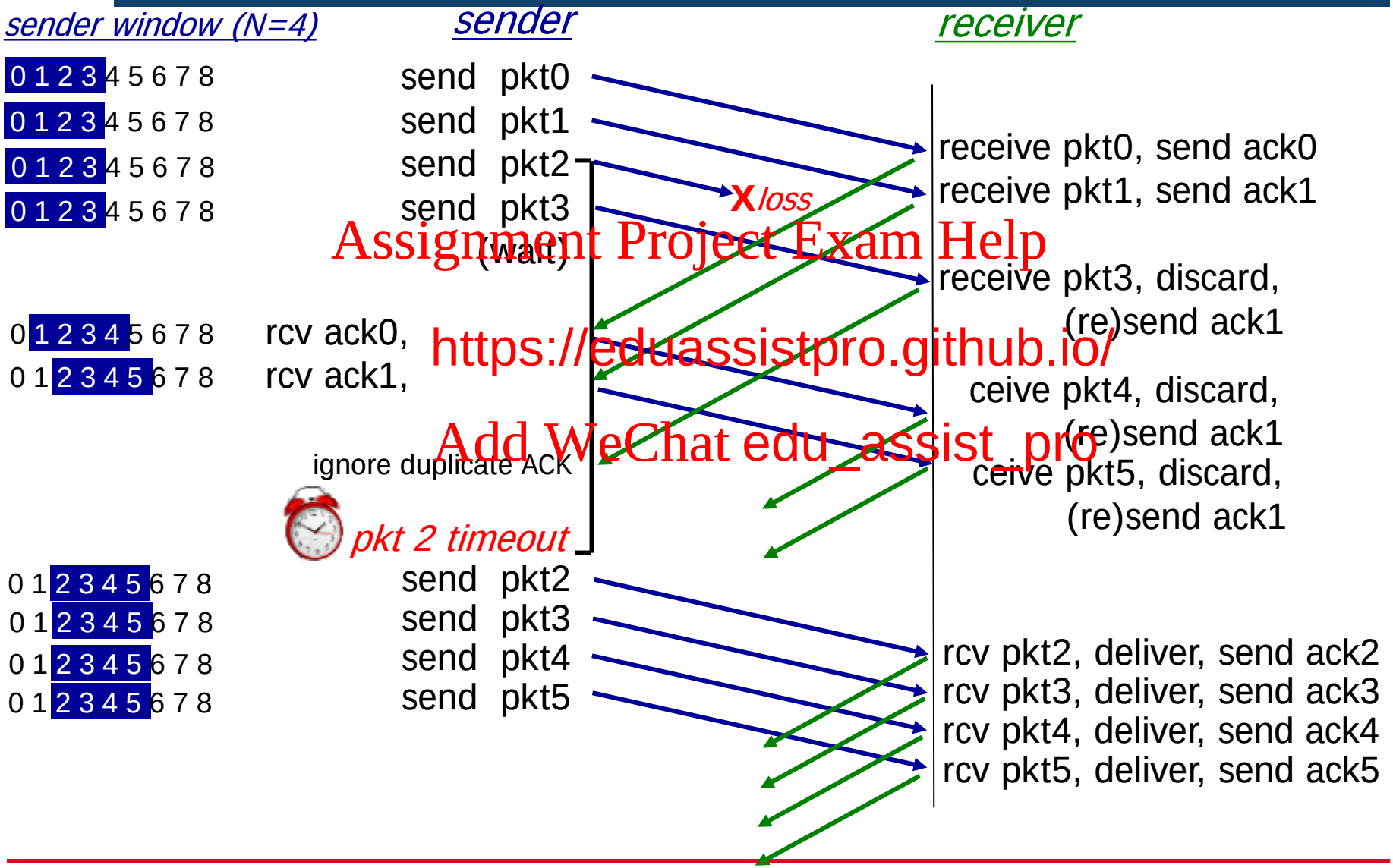
connection state at
eiver

- small header size
- no congestion control: UDP can blast away as fast as desired

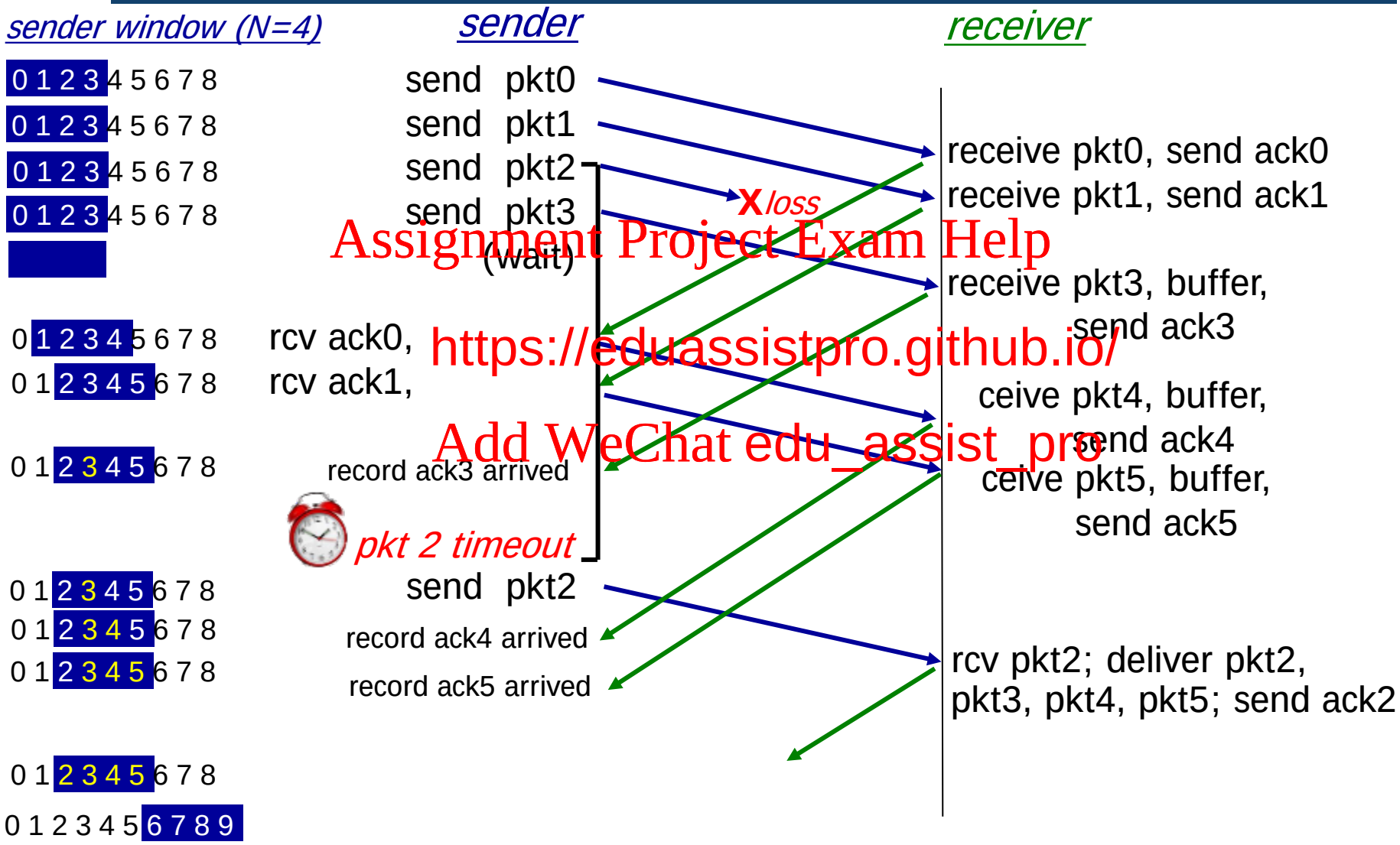
Stop and wait



GBN

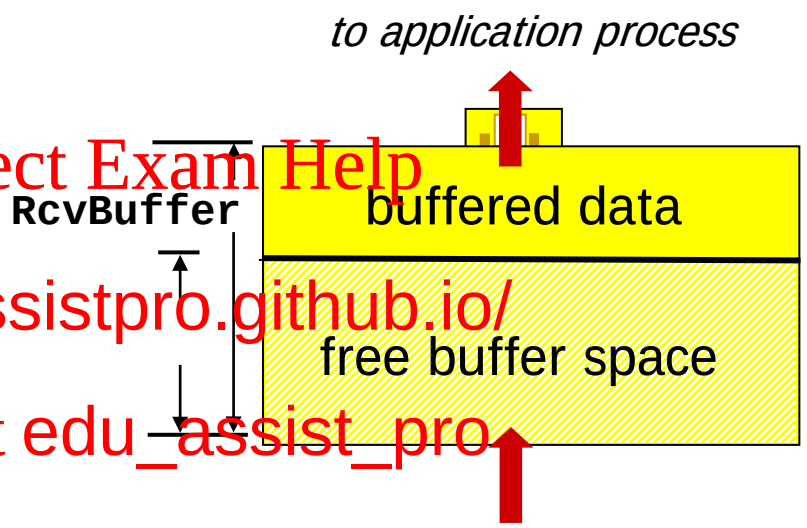


Selective repeat



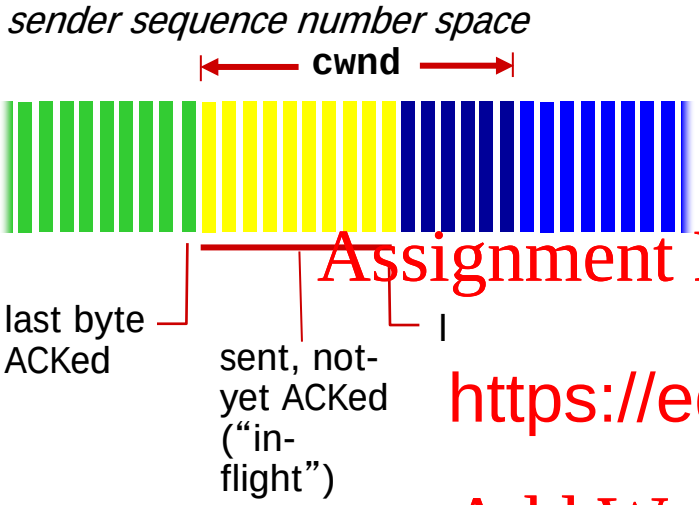
TCP flow control

- receiver "advertises" free buffer space by including **`rwnd`** value in TCP header of receiver-to-sender segments
  - **`RcvBuffer`** size s options (typical defau bytes)
  - many operating systems autoadjust **`RcvBuffer`**
- sender limits amount of unacked ("in-flight") data to receiver's **`rwnd`** value
- guarantees receive buffer will not overflow



receiver-side buffering

TCP Congestion Control



TCP sending rate:

› *roughly:* send cwnd bytes, wait RTT for ACKs, then send more

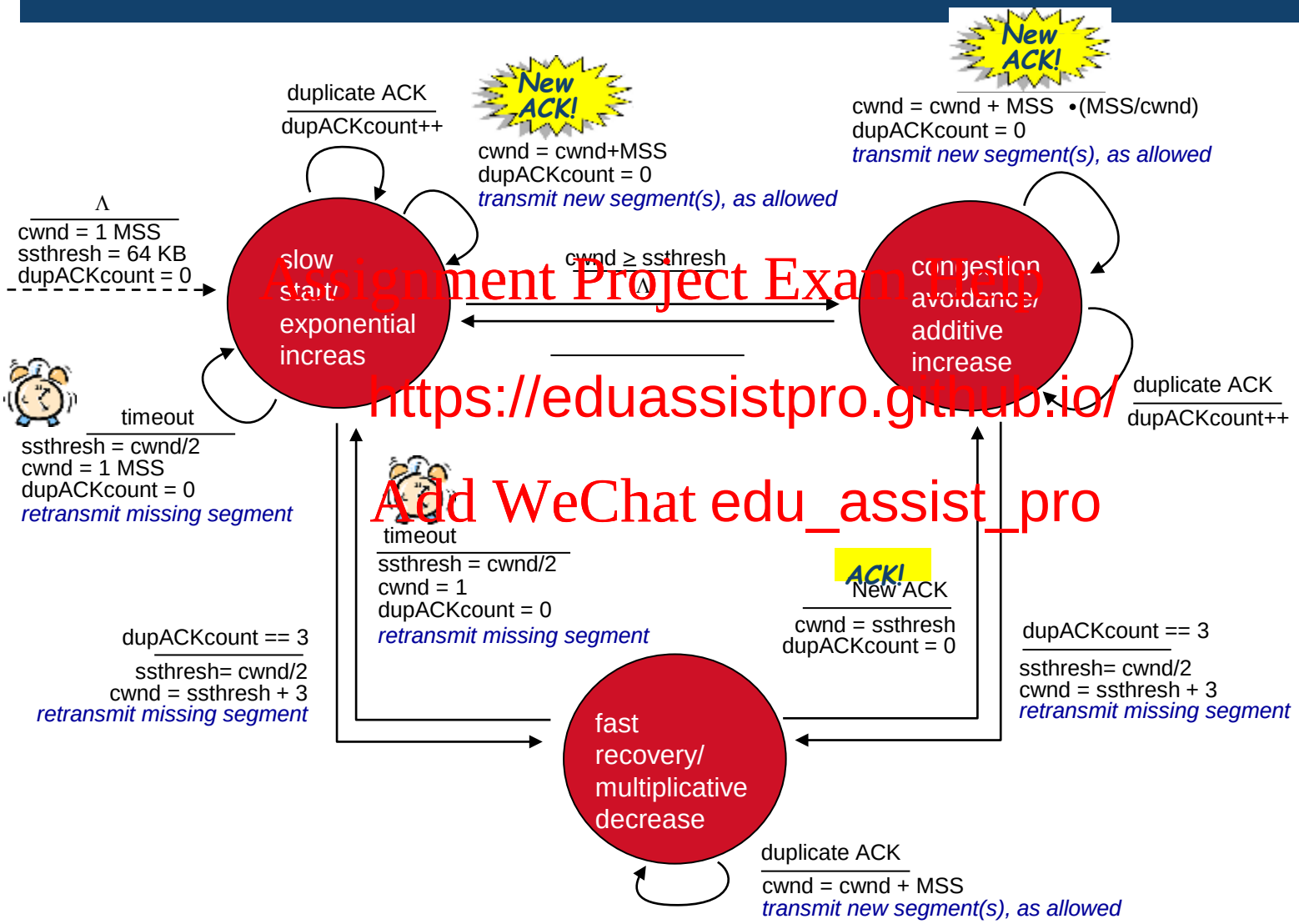
› sender limits transmission:

LastByteSent - LastByteAcked ≤ cwnd

› **cwnd** is dynamic, function of perceived network congestion

$$\text{rate} \approx \frac{\text{cwnd}}{\text{RTT}} \text{ bytes/sec}$$

TCP Congestion Control



TCP round trip time, timeout

› timeout interval: **EstimatedRTT** plus "safety margin"

EstimatedRTT = (1-α)*EstimatedRTT + α*SampleRTT

typical value: α = .125

DevRTT = (1-β)*DevRTT + β*|SampleRTT-EstimatedRTT|

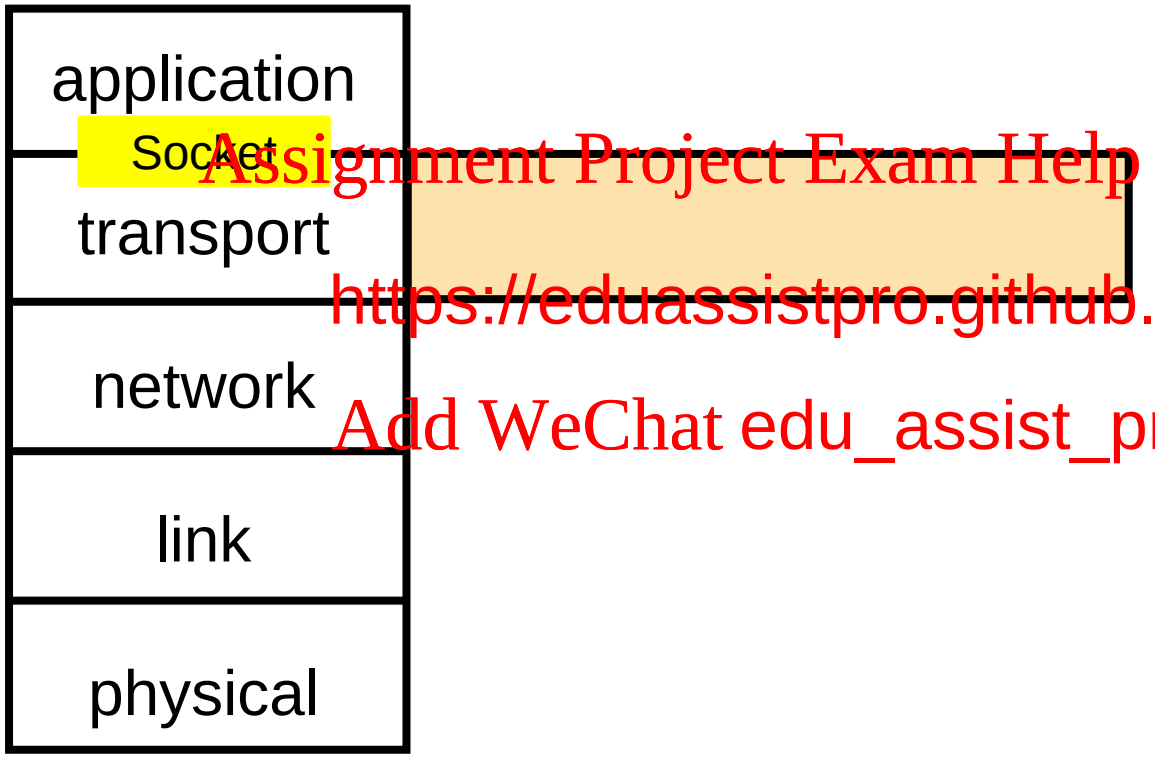
(typical β = .25)

TimeoutInterval = EstimatedRTT + 4*DevRTT

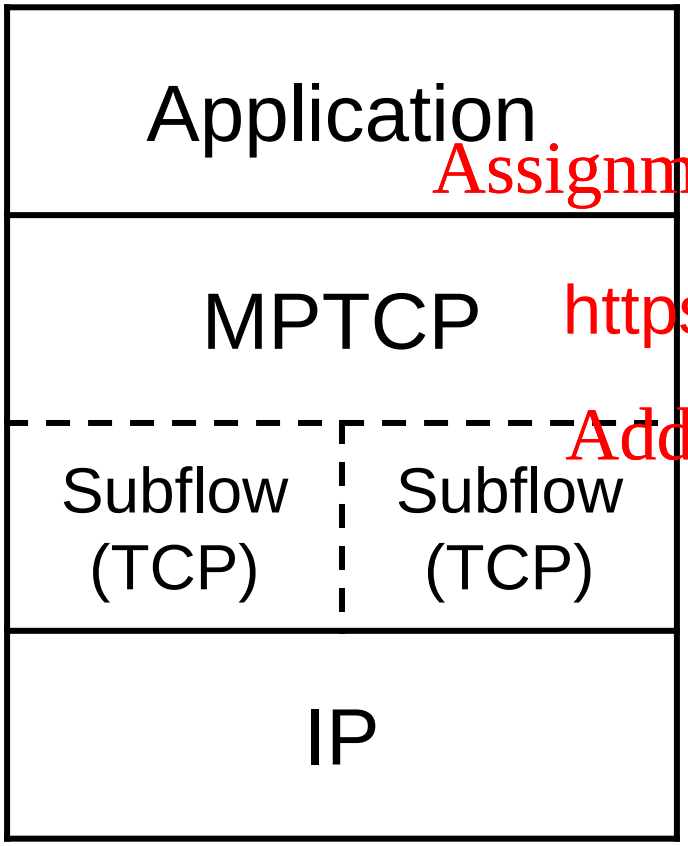
estimated RTT

"safety margin"

Protocol Stack



MPTCP in Stack

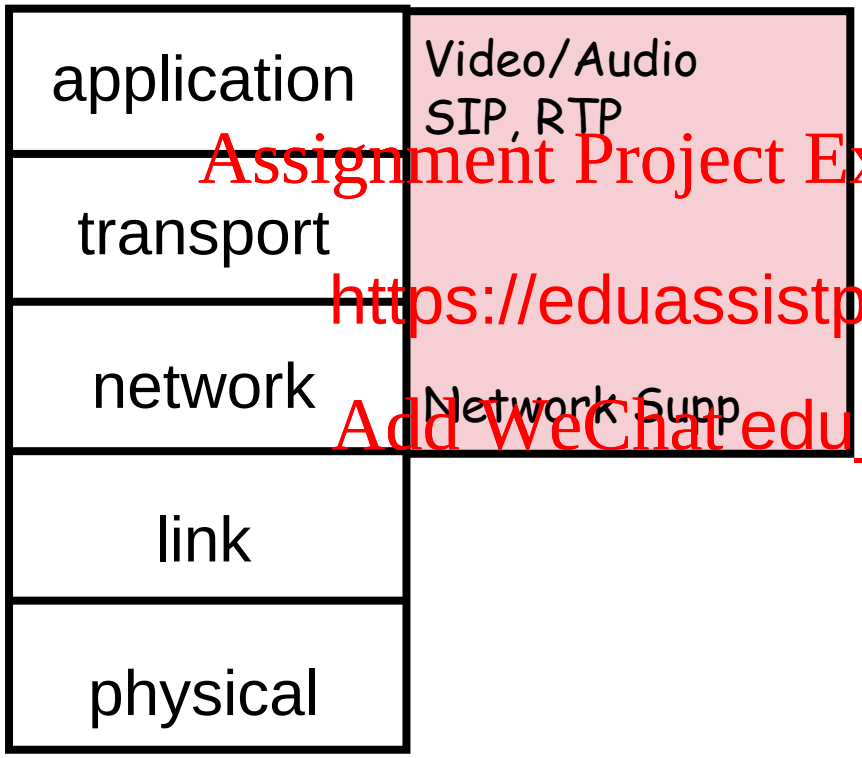


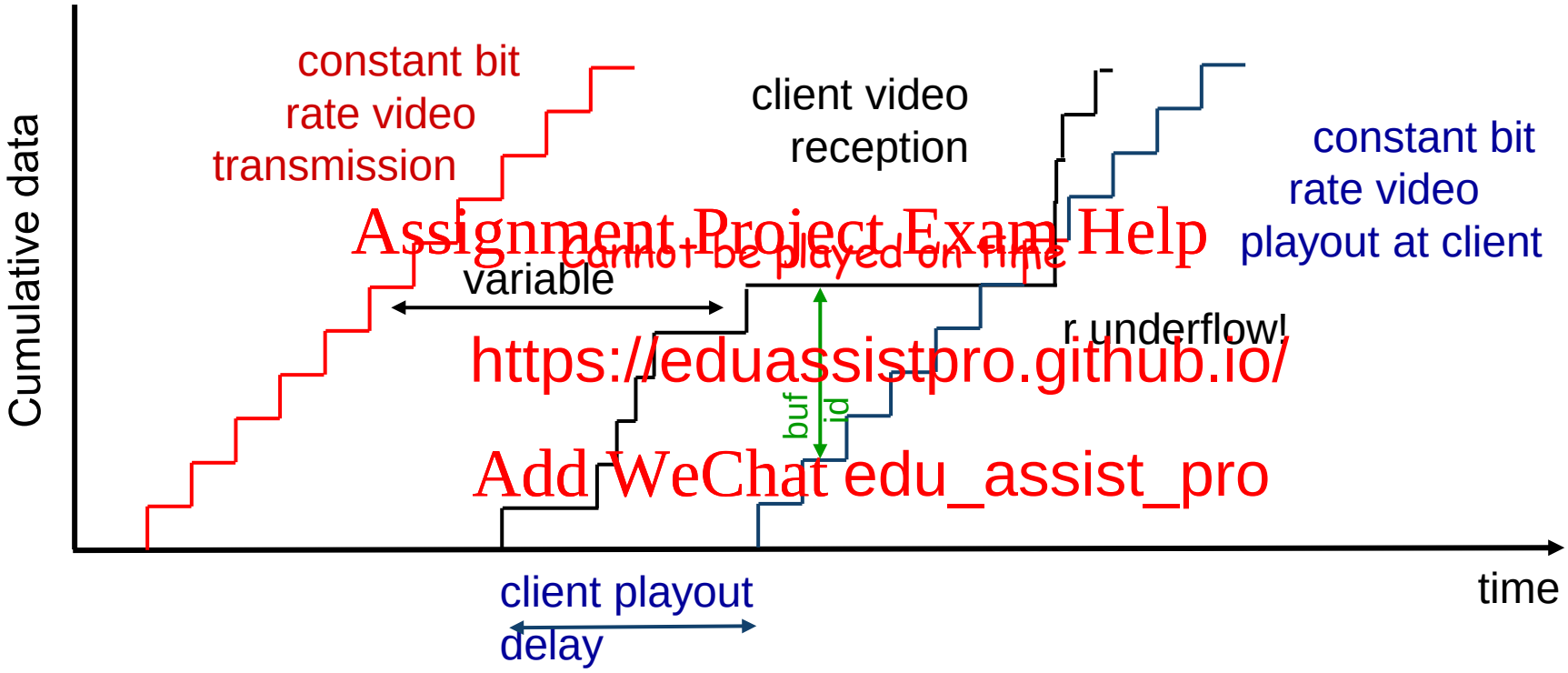
https://pocketnow.com/multipath-tcp

MPTCP in Stack

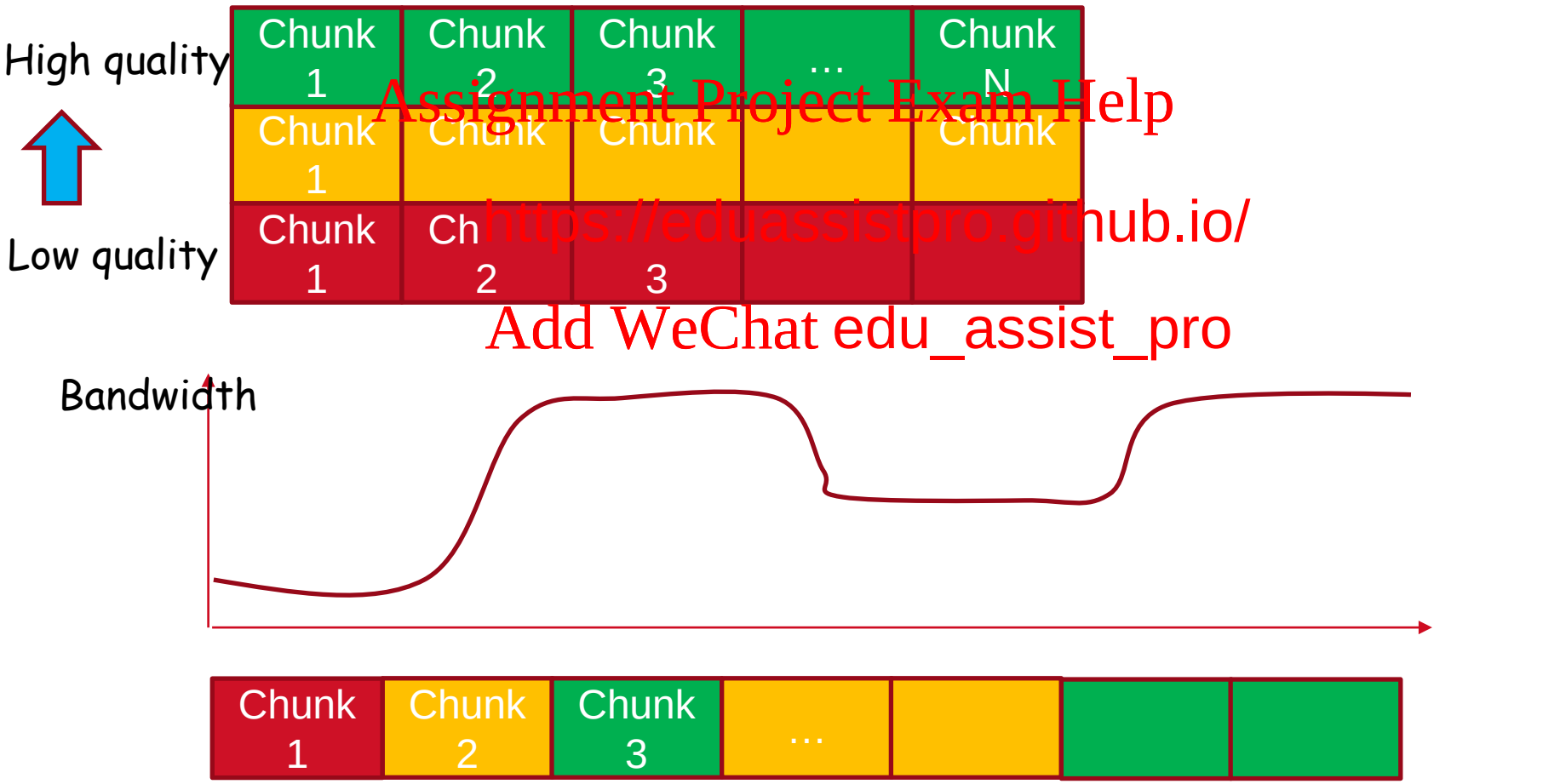
- *Initialization: MP_CAPABLE, JOIN, Token*
- *Sequence number: Subflow sequence number + data sequenc*
- *Flow control: is for all subflows.*
- *Congestion control: Upcat or fairness.*

Internet Protocol Stack





Streaming multimedia: DASH



Adaptive playout delay

- *goal:* low playout delay, low delay loss rate
- *approach:* adaptive playout delay adjustment:
  - estimate network delay, adjust playout delay at beginning of each talk spurt
  - silent periods compressed and elongated
- adaptively estimate [illegible] ometially weighted moving average):

$$d_i = (1-\alpha)d_{i-1} + \alpha\,(r_i - t_i)$$

delay estimate after ith packet

small constant,

time received - time sent (timestamp)

measured delay of ith packet

Adaptive playout delay

- also useful to estimate average deviation of delay, v_i

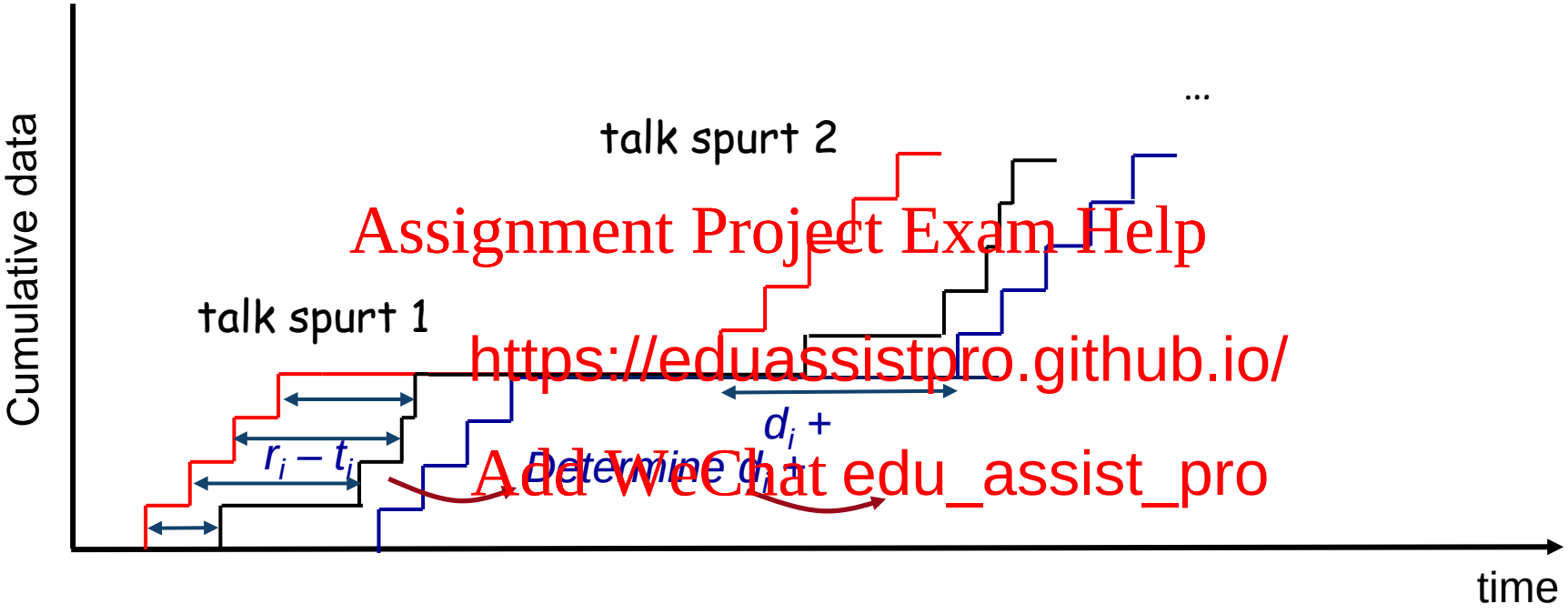
$$v_i = (1-\beta)v_{i-1} + \beta\,|r_i - t_i - d_i|$$

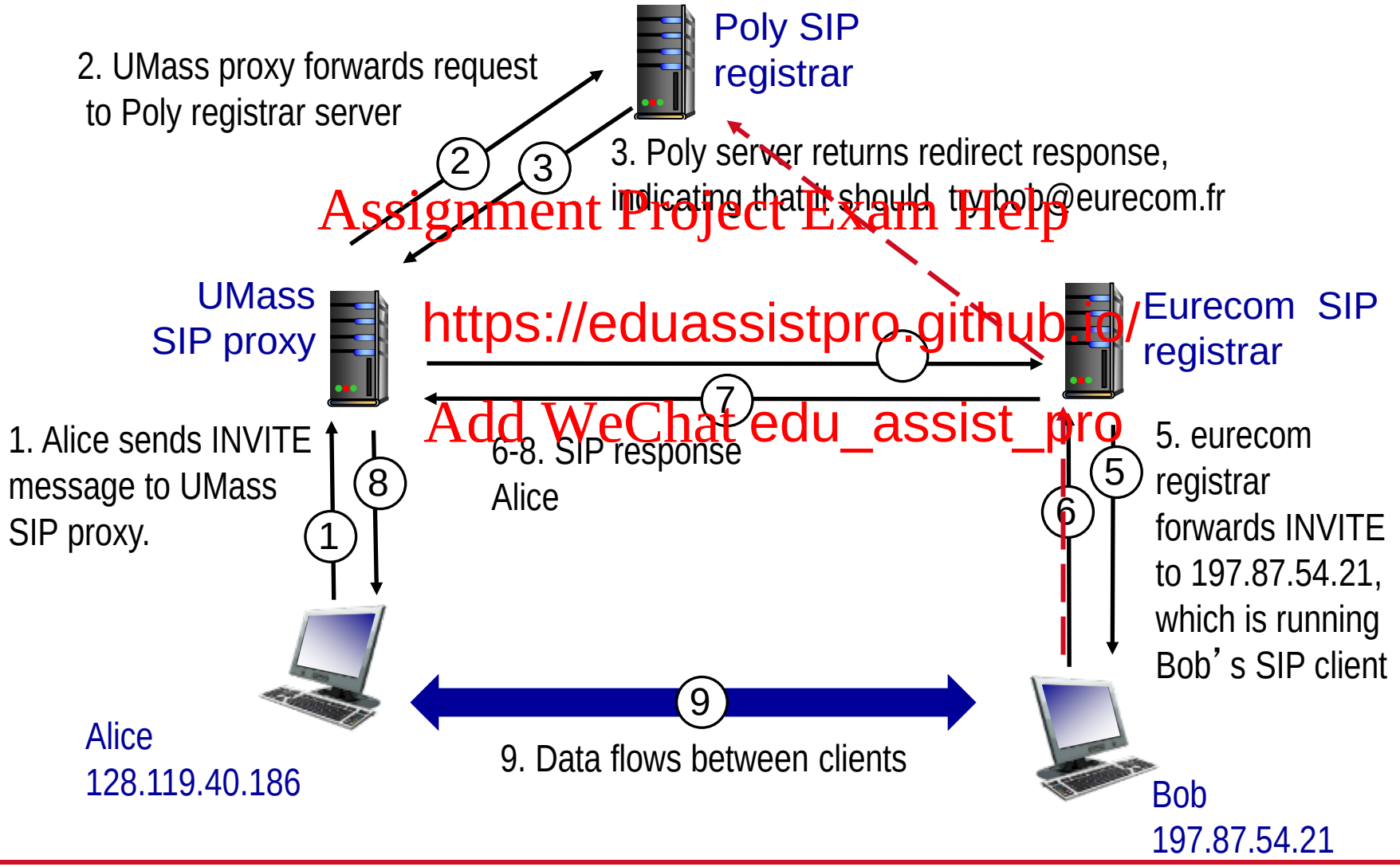
  - estimates d_i, v_i calculated for every received packet, but used only at start of talk spurt

  - for first packet in talk spurt, playout

$$playout\text{-}time_i = t_i + d_i + Kv_i$$

Delay jitter





RTP

payload type	sequence number	time stamp	Synchronization Source ID (SSRC)	Miscellaneous fields

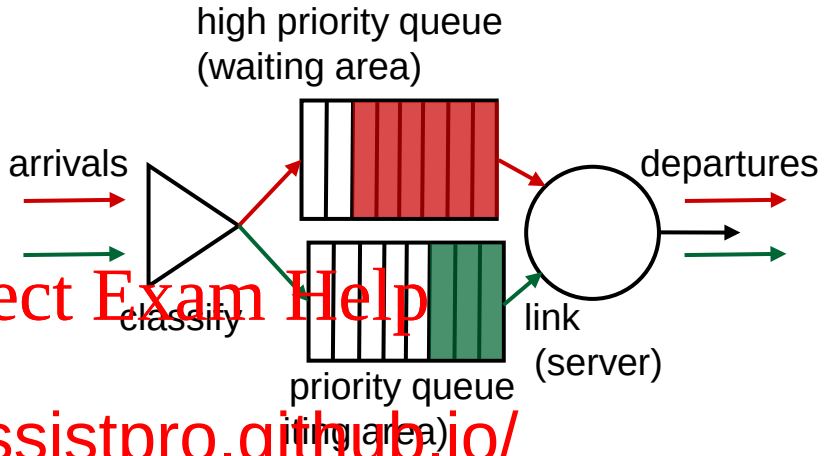
- payload type (7 bits): indicates type of encoding currently being used.
- sequence # (16 bits): increment by one for each RTP packet sent
- *timestamp field* instant o in this RTP data packet

- Sequence + timestamp： packet loss or new talk spurt.

Scheduling policies: priority

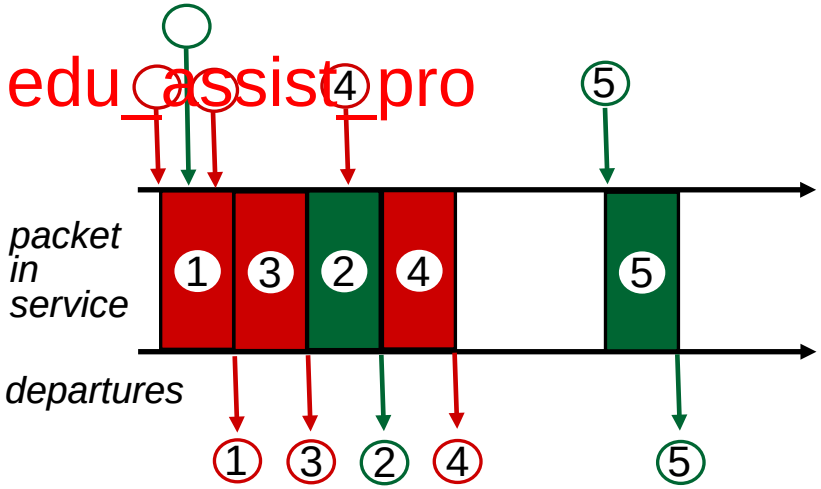
priority scheduling: send highest priority queued packet

non-preemptive



› multiple *classes*, w priorities

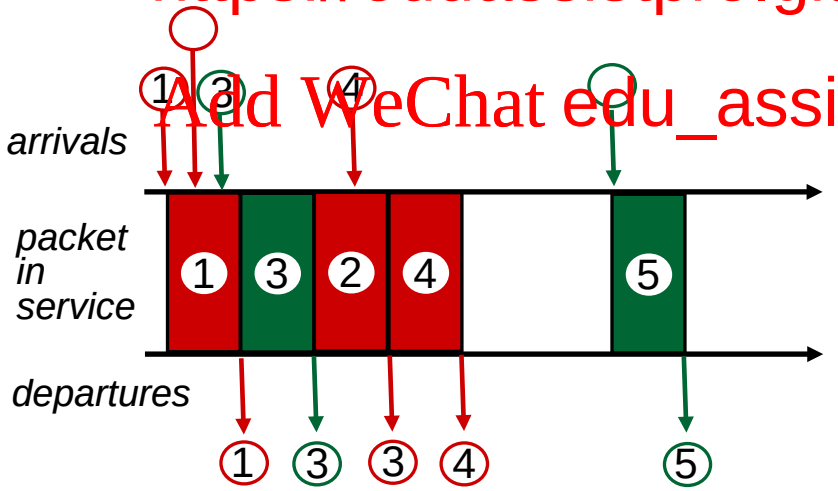
- class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc.
- real world example?



Scheduling policies (con't)

Round Robin (RR) scheduling:

- multiple classes, with equal priority
- cyclically scan class queues, sending one complete packet from each class (if available)



Scheduling policies (con't)

Weighted Fair Queuing (WFQ):

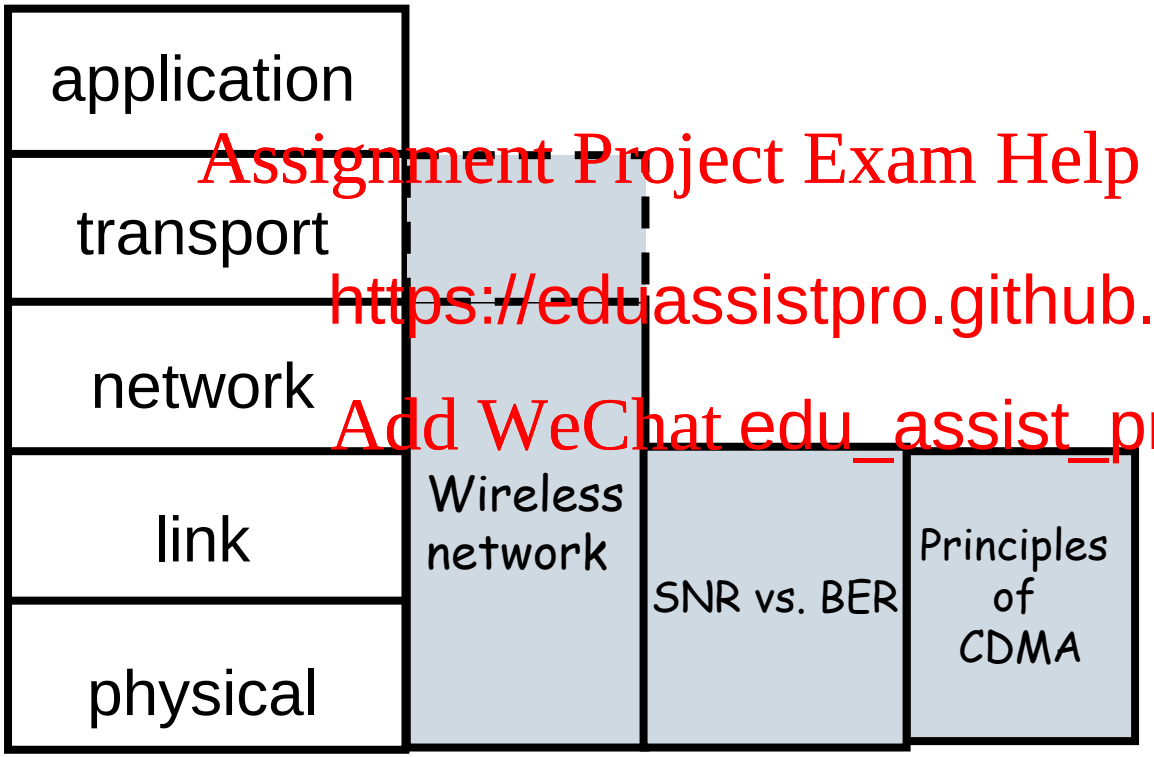
- Each class *i* is assigned a weight w_i
- *Guarantee*: if there are class *i* packets to send (during some interval) then class *i* receives a fraction of service which is $w_i / (\sum w_j)$
- On a link with transmi roughput $Rw_i / (\sum w_j)$

Policing mechanisms: implementation

token bucket: limit input to specified *burst size* and *average rate* (useful to police the flow)

- bucket can hold b tokens
- a packet must remove a token from bucket to be transmitted into the network
- tokens generated at rate *r token/sec* unless bucket full (token ignored)
- *over interval of length t: number of packets admitted less than or equal to (rt + b)*
- *Token-generation rate r limits the rate at which packets enter the network*

Internet Protocol Stack



Wireless Physical layer

› *SNR versus BER tradeoffs*

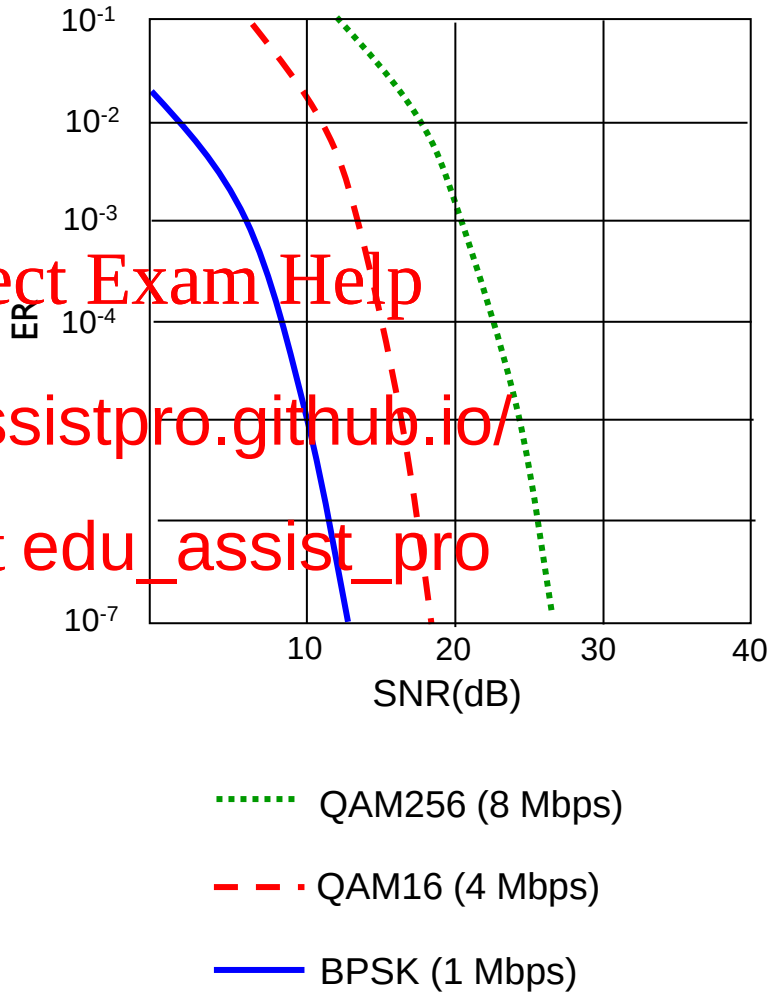
- *Different physical layer modulation:*

What is the meaning o

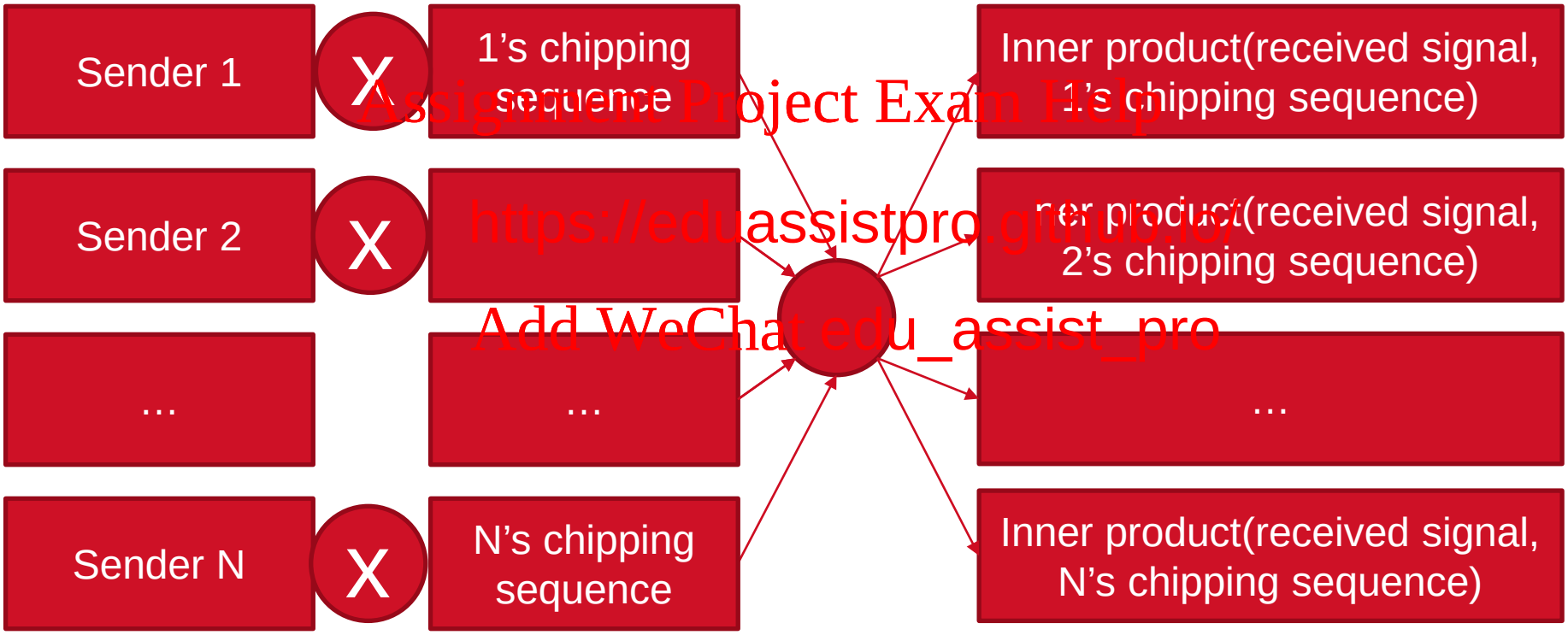
Normal/Gaussian distribution, Q fucntion

Week 9, Assignment 2 Q4

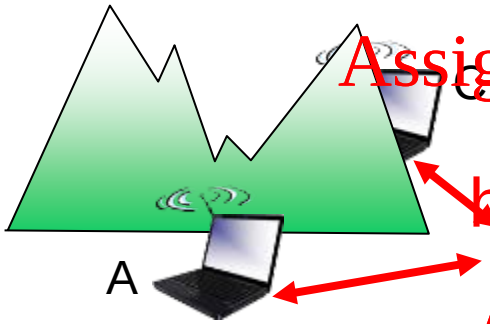
In final exam.



CDMA



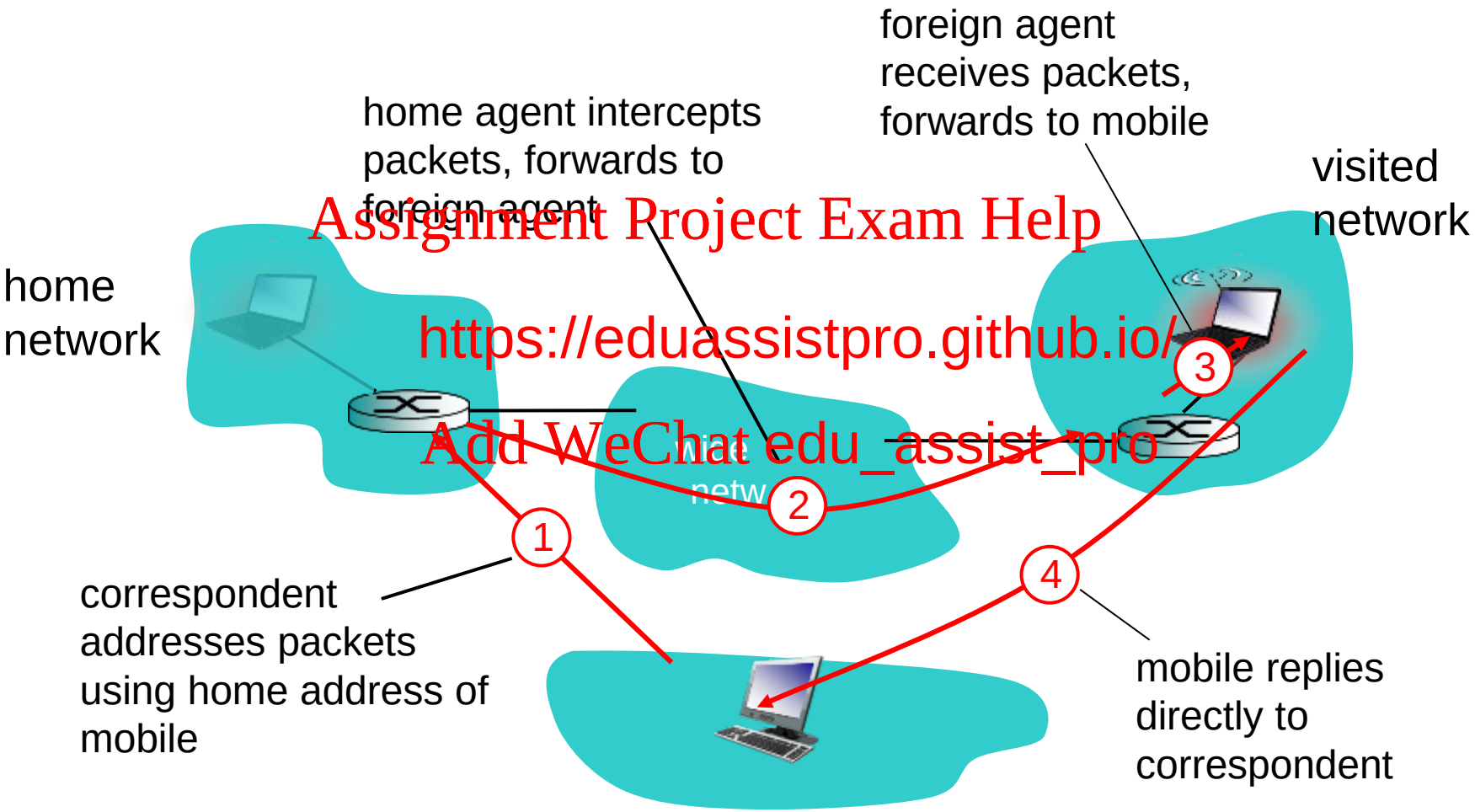
Hidden terminal and exposed terminal



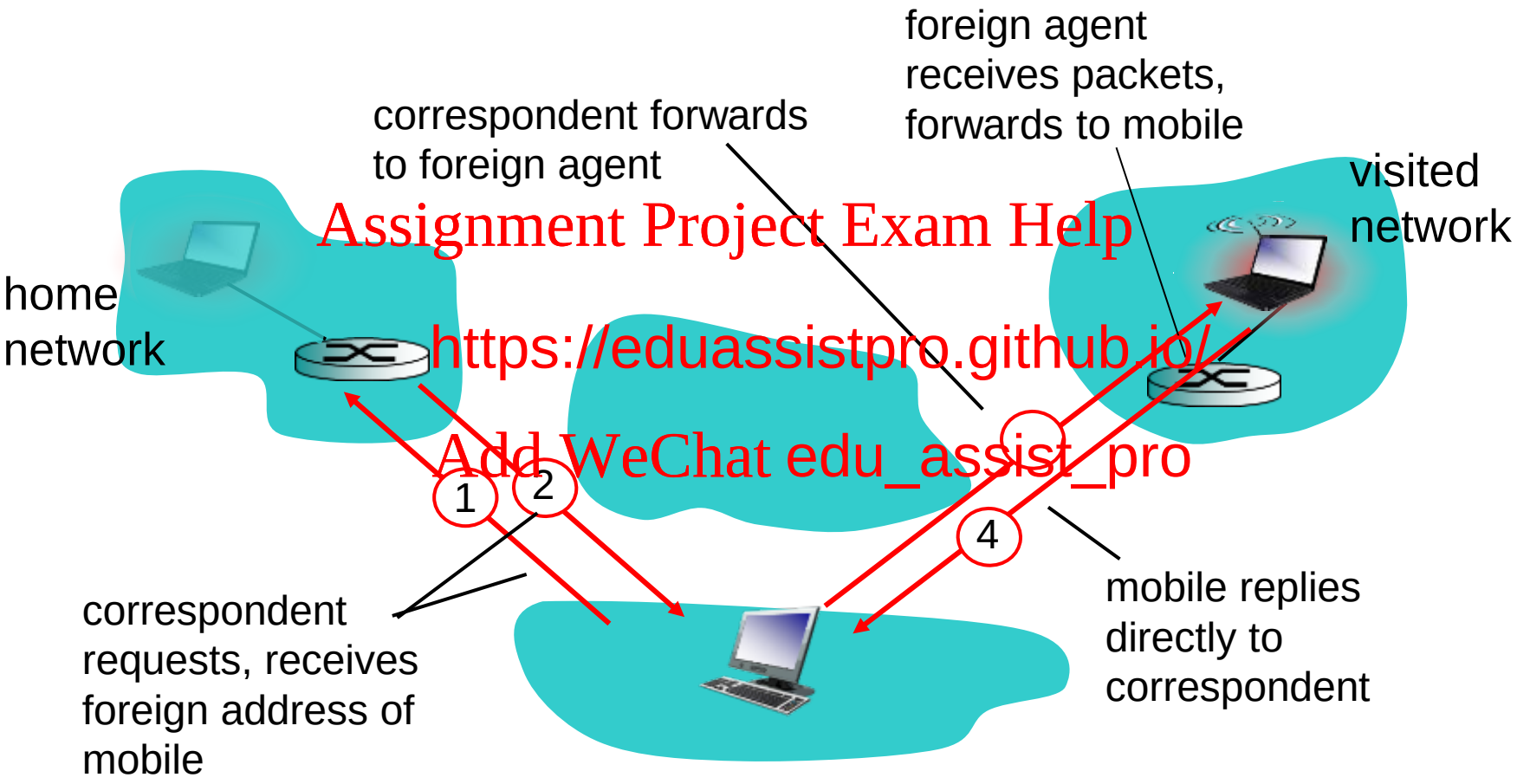
Hidden terminal problem

Exposed terminal problem

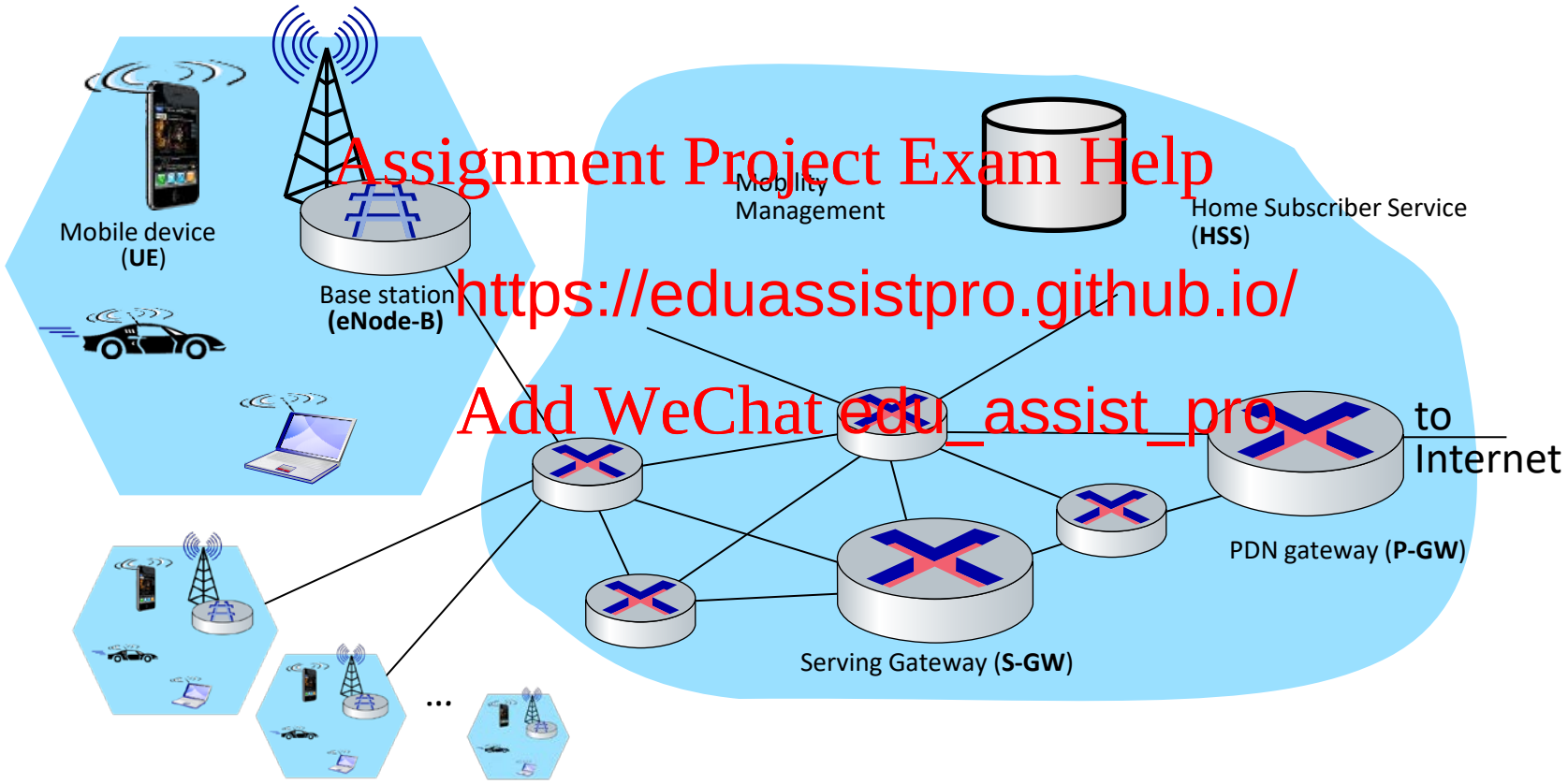
Mobility via indirect routing



Mobility via direct routing



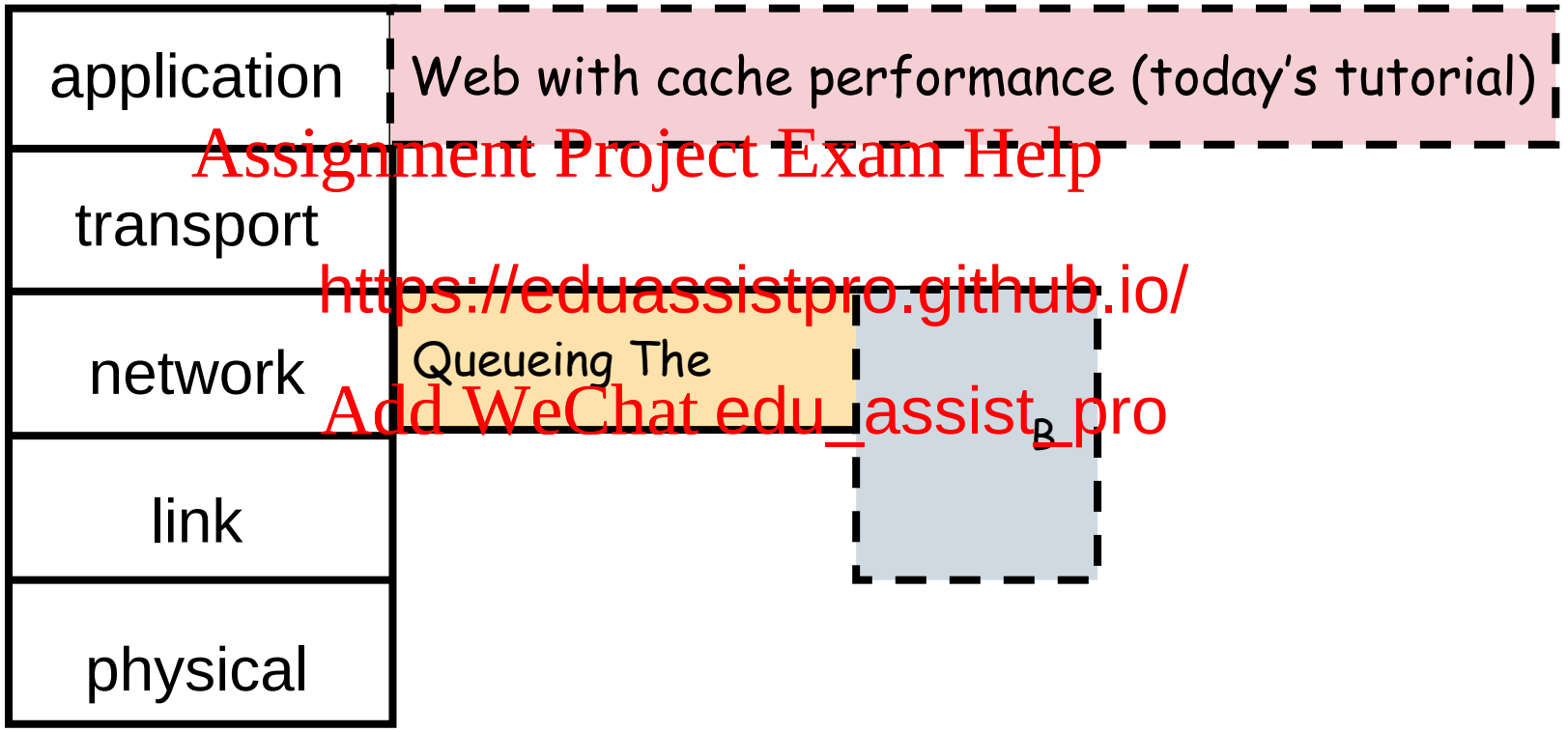
4G LTE



Wireless, mobility: impact on higher layer protocols

- logically, impact *should* be minimal …
  - best effort service model remains unchanged
  - TCP and UDP can (and do) run over wireless, mobile
- … but performance-wise:
  - packet loss/delay due t … elays for link-layer retransmissions), and h
  - TCP interprets loss as congestion, will decr … window un-necessarily

Internet Protocol Stack: Theory



$$\text{Number} = \text{Time} \times N$$

- W: average waiting time in queue
- X: average service time
- T: average time spent in system ($T = W + X$)
- N_Q = average number of customers in queue
- ρ = utilization = average number of customers in service
- N = average number of customer in system ($N = N_Q + \rho$)
- Want to show later: $N = \lambda T$ (Little's theorem)
- λ Average arrival rate

Stationary Distribution Derivation

Transition diagram and balance

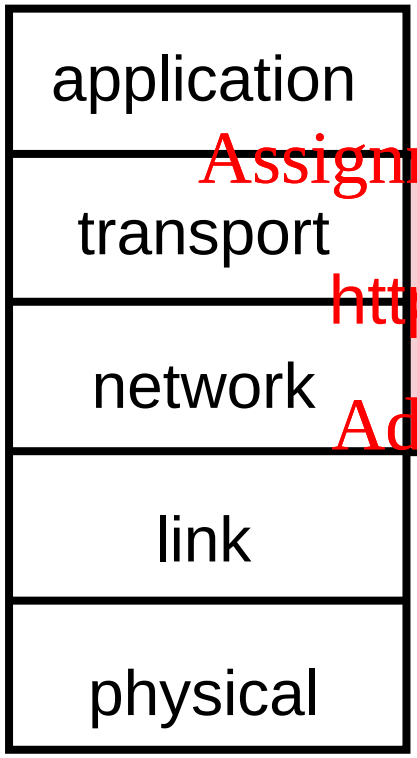
Stationary distribution

Average # of users

Average waiting time

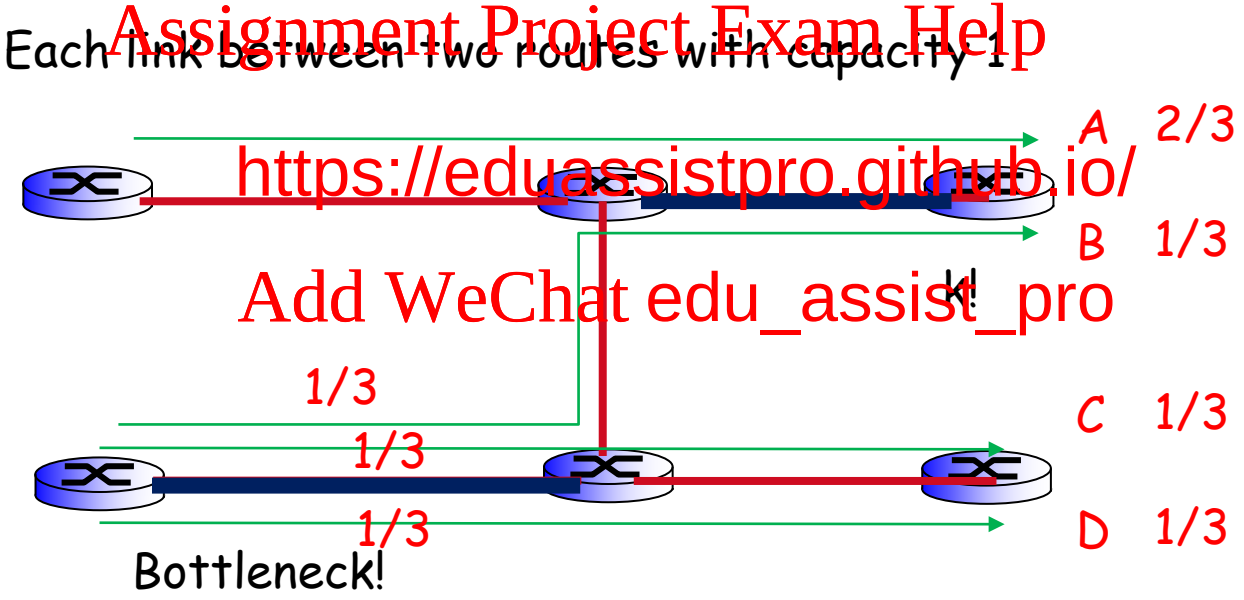
In final exam.

Internet Protocol Stack



Max-min fairness

How to judge if max-min fairness is satisfied.
How to find max-min fairness: Bottleneck approach



THE UNIVERSITY OF SYDNEY

Assignment 1 common mistake

Correct solution

Common mistake

Why is it wrong?

Common mistake

These are not independent events!

P(A and B) = P(A) P(B) is true for independent events.

$d_1'>r_{ab}$ is happens, -> more likely r_{ab} is small -> more likely $d_2'>r_{ab}$ is also true.

Q: Could you give an example when the above approach is correct?
A: r_{ab} is a constant. d_1', d_2', d_3', d_4' are independent and thus $d_1'>r_{ab}$, $d_1'>r_{ab}$, $d_3'>r_{ab}$, $d_4'>r_{ab}$ are independent!

Another correct way

Because r_{ab}, d_1', d_2', d_3', d_4' are continuous random variable, and **independent** and they follow the **same distribution**, (i.i.d. independent and identically distributed), so that they have the same probability, i.e., 1/5, to be the smallest one.

Therefore $P(r_{ab} > \min$

Simple way to verify 4/5

Final exam

- The marks of final exam sum up to 100 and it is worth **60%** of your overall mark.
- Online, open book, (type C)
- 130 minutes + buffer time + upload time
- Double-pass po
- 7 questions in total.
  - Calculation, short answer and response
- Type your answers in the blank below, or write down, scan/photograph, and upload in the end.
- Spend time wisely. Question 1 doesn't mean easiest.

Final exam

- No programming questions
- No Wireshark

Office hour

- By appointment
  - wei.bao@sydney.edu.au
  - zhengjie.yang@sydney.edu.au
  - zwan5430@uni.sydney.edu.au
- Assignment 2 commo
  - 4-Dec-2020 (Fri), 3pm
  - Non-compulsory, no recording
- Last-chance office hour
  - 7-Dec-2020 (Mon), 3pm (tentative), Zoom
  - Non-compulsory, no recording

Survey

- **Unit of Study Surveys (USS) for Semester 2 are now open!**
- Login to the University's Student Survey System now to complete a survey:
- **https://student-surveys.sydney.edu.au/students**
- Survey completed will ize draw to win a range of Apple products incl an Apple Watch and JB HiFi Gift Cards