

Advanced Network Technologies

Week 2:

Network performance

Network Application Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Dr. Wei Bao | Lecturer
School of Computer Science



THE UNIVERSITY OF
SYDNEY



Network Performance: Throughput

Assignment Project Exam Help

<https://powcoder.com>

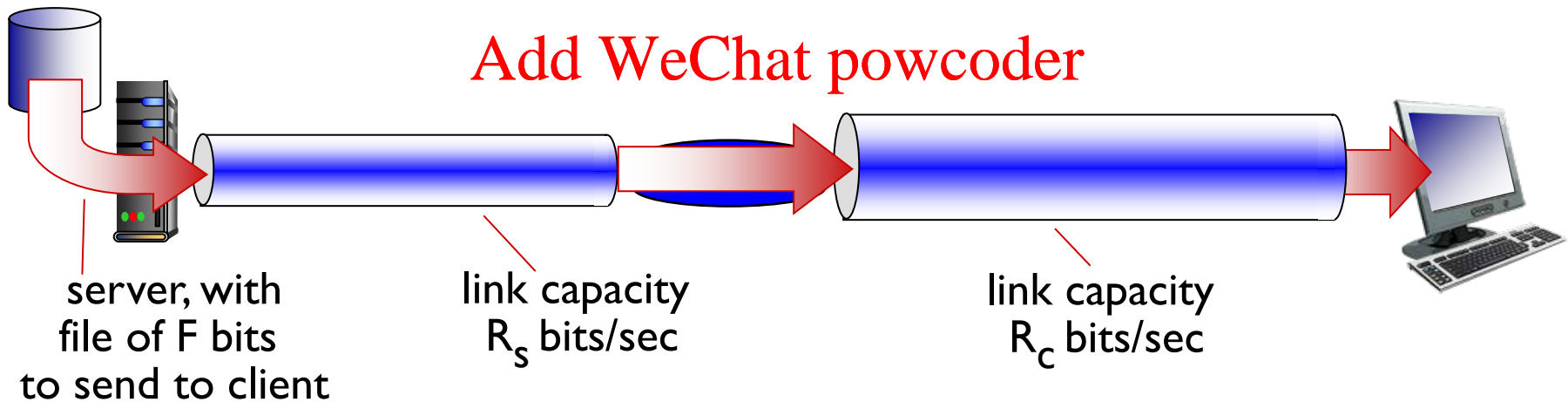
Add WeChat powcoder

- › **throughput**: rate (bits/time unit) at which bits transferred between sender/receiver
 - **instantaneous**: rate at given point in time
 - **average**: rate over longer period of time

Assignment Project Exam Help

<https://powcoder.com>

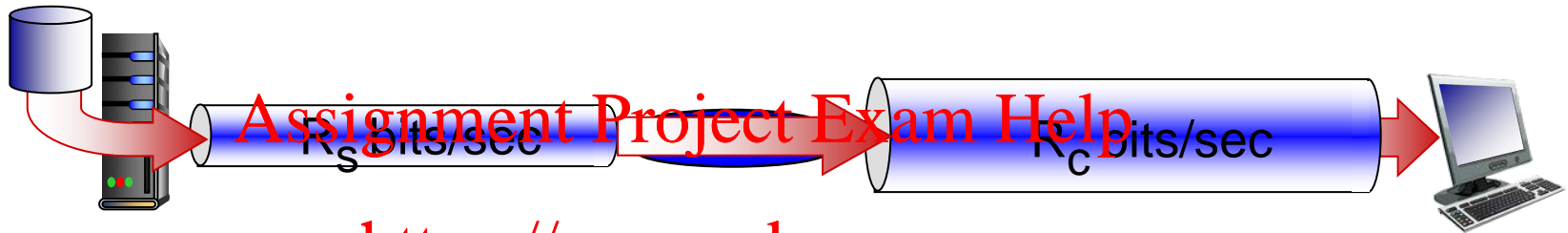
Add WeChat powcoder



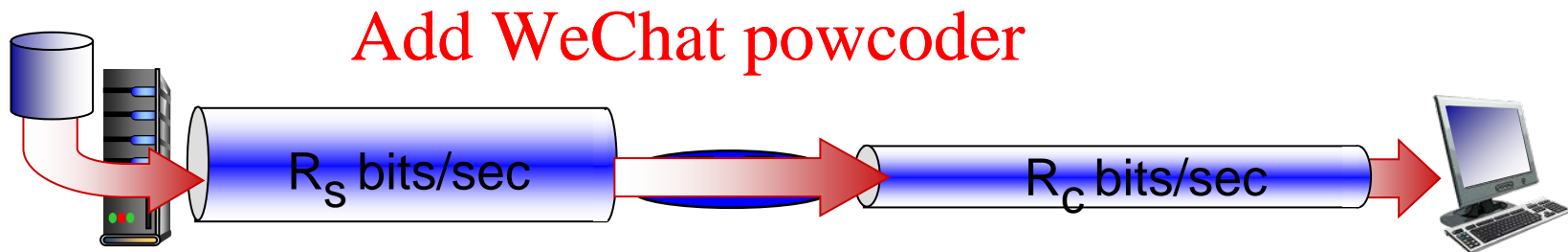


Throughput (cont'd)

- › $R_s < R_c$ What is average end-end throughput?



- › $R_s > R_c$ What is average end-end throughput?



bottleneck link

link on end-end path that constrains end-end throughput

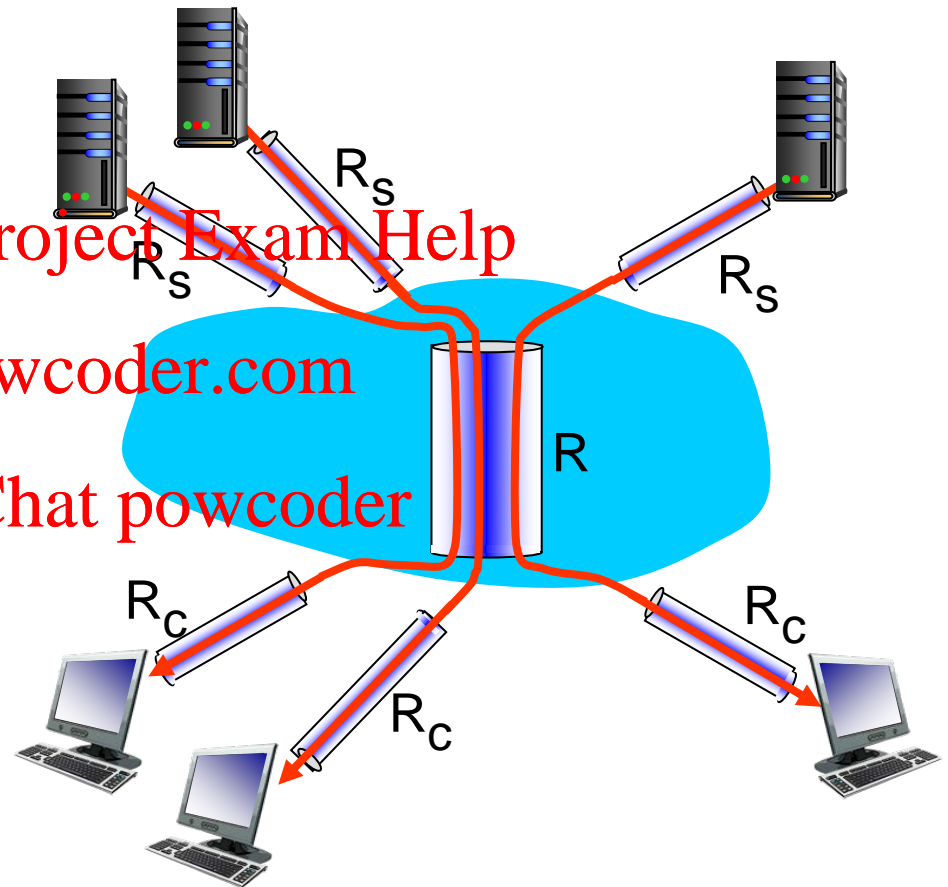
Internet Scenario

- › per-connection end-end throughput: $\min(R_c, R_s, R/10)$

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



10 connections (fairly) share
backbone bottleneck link R bits/sec



Network Performance: Assignment Project Exam Help Fairness

<https://powcoder.com>

Add WeChat powcoder

In reality: two considerations

› Efficiency

› Fairness

Assignment Project Exam Help

<https://powcoder.com>

› However, they are contradicting!

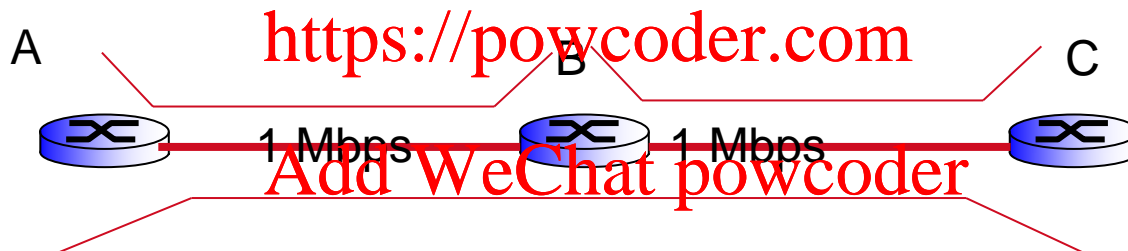
Add WeChat powcoder



Network Fairness, Bandwidth allocation

Three flows: A-B, B-C, A-C

Assignment Project Exam Help



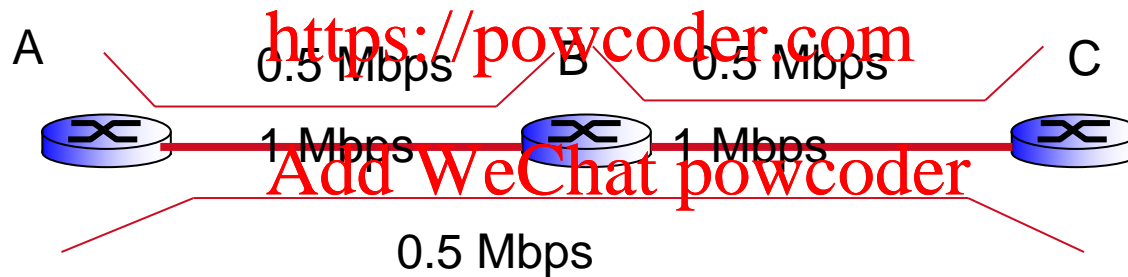
Q: How can we allocate the link bandwidths to the three flows?



Network Fairness, Bandwidth allocation

Three flows: A-B, B-C, A-C

Assignment Project Exam Help



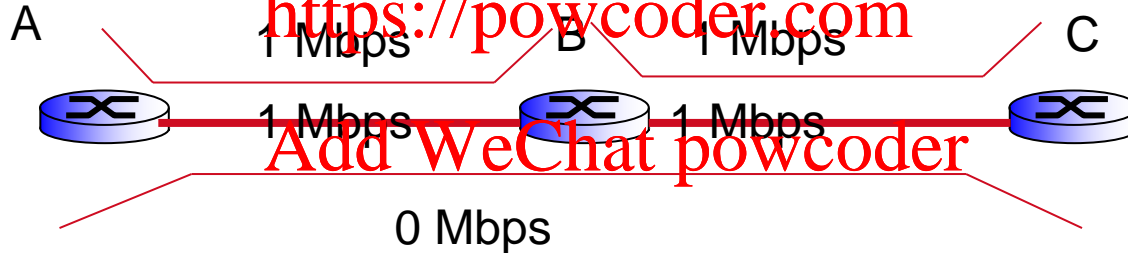
Very fair!

However: Network throughput, only 1.5Mbps

Three flows: A-B, B-C, A-C

Assignment Project Exam Help

~~https://powcoder.com~~



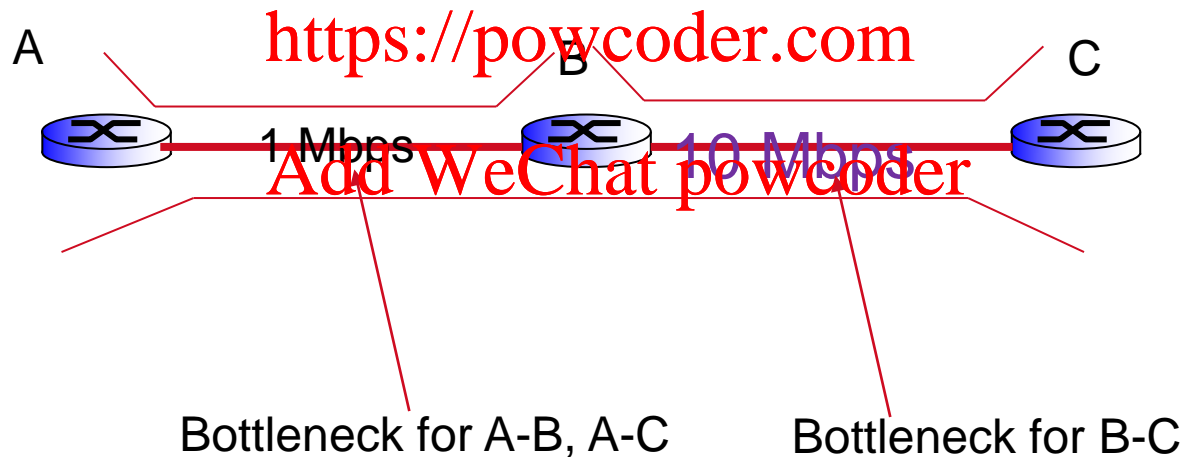
Very unfair!

However: Network throughput, 2Mbps



Bottleneck for a flow: The link that limits the data rate of the flow

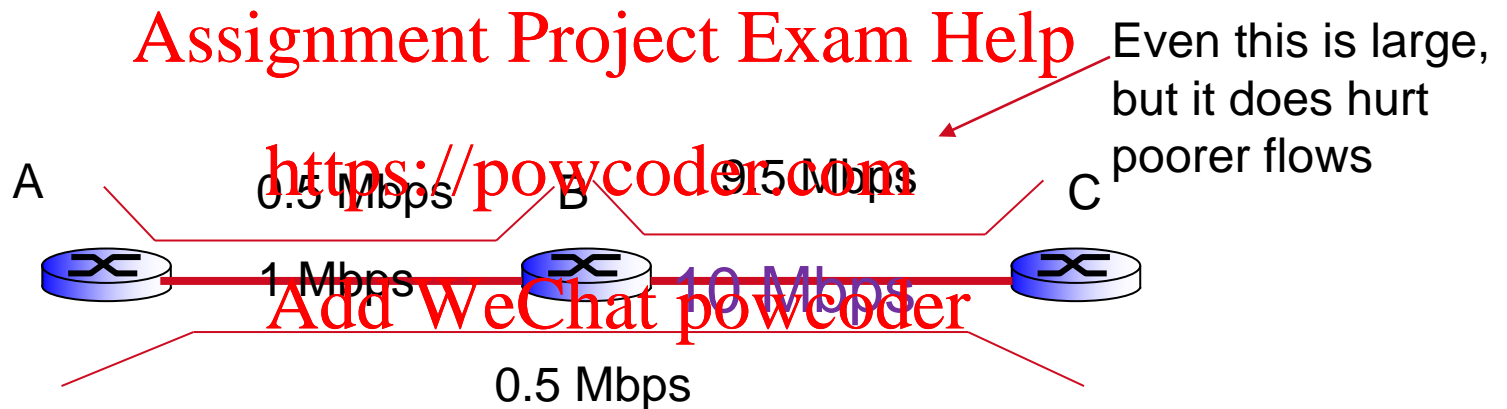
Assignment Project Exam Help



- › Maximize the minimum
- › Try to increase the “poorest” as much as possible
 - A richer flow can be sacrificed.
- › Try to increase the second “poorest” as much as possible
 - A richer flow can be sacrificed.
 - A poorer flow cannot be sacrificed.
- › Try to increase the third “poorest” as much as possible
- › ...
- › Max-min Fairness criteria: if we want to improve one flow, we can only achieve this by sacrificing a poorer or equal flow.



Bottleneck for a flow: The link limits its data rate



Bottleneck approach

- › 1 Start with all zero flows, potential flow set = {all flows}
- › 2 Slowly increase flows in the potential flow set until there is a (new) link saturated
 - “Pouring water in the network”
- › 3 Hold fix the flows that are bottlenecked, remove them from the potential flow set
- › 4 If potential flow set is not empty, go to step 2 (still has potential to increase)

[Assignment Project Exam Help](https://powcoder.com)

<https://powcoder.com>

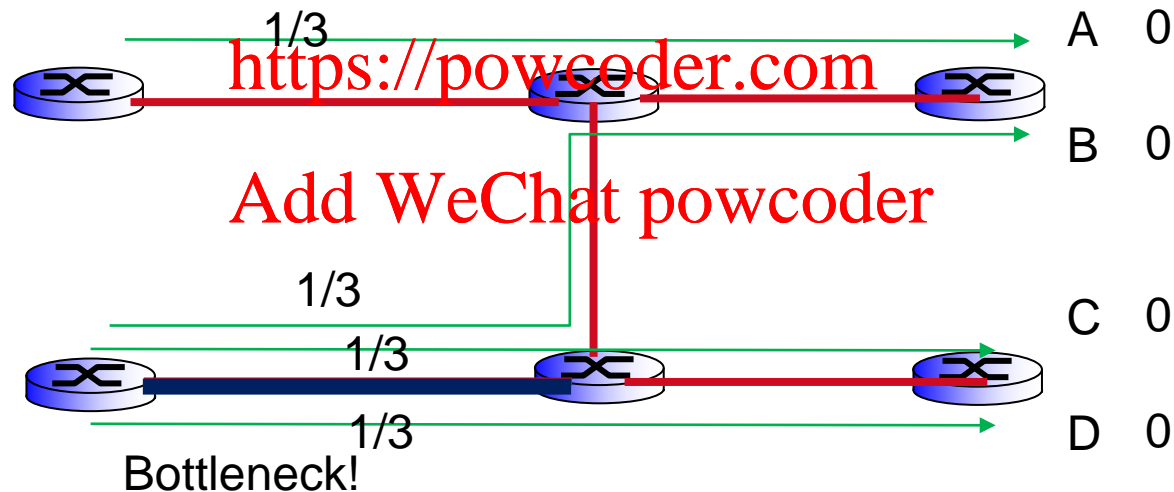
[Add WeChat powcoder](https://powcoder.com)



Bottleneck approach

Each link between two routes with capacity 1

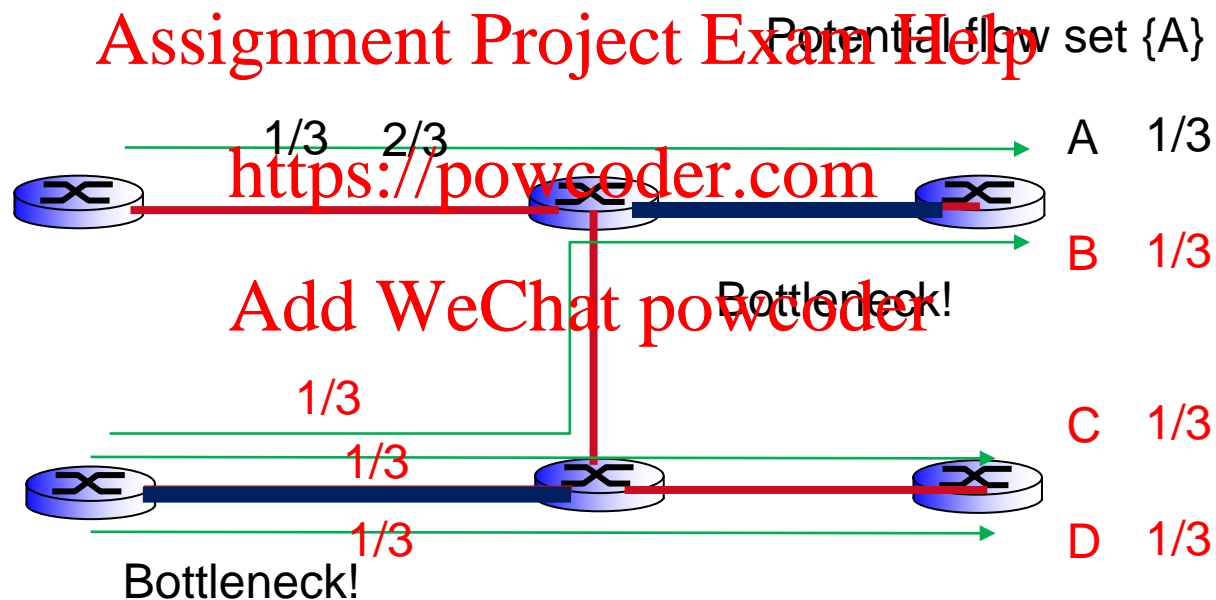
Assignment Project Exam Help Potential flow set {A, B, C, D}





Bottleneck approach

Each link between two routes with capacity 1

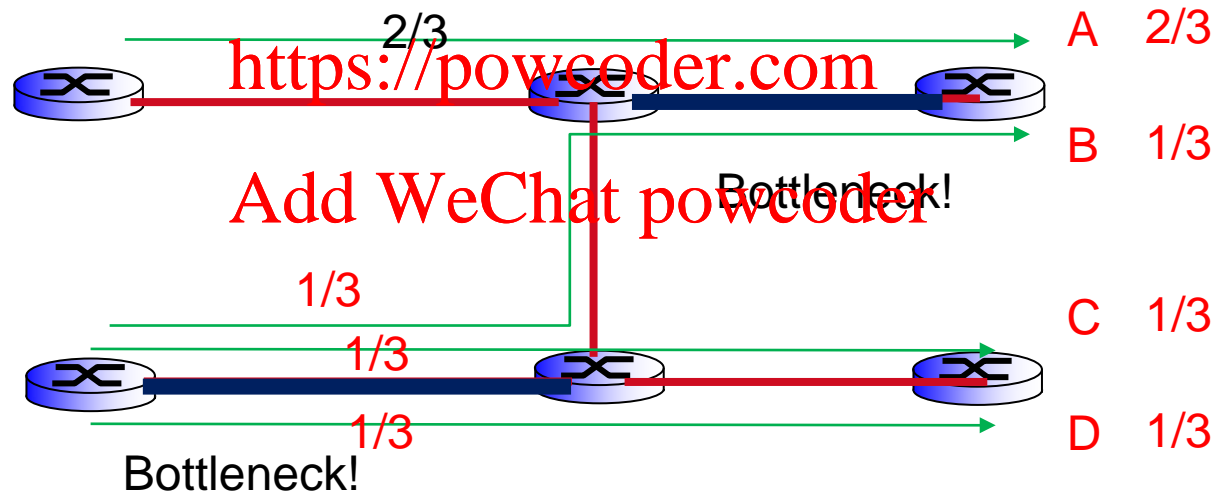




Bottleneck approach

Each link between two routes with capacity 1

Assignment Project Exam Help Potential flow set {}





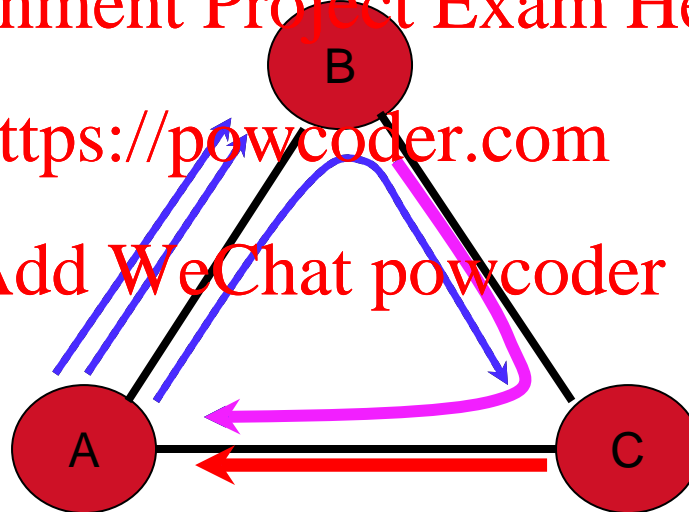
Can you solve the following problem?

link rate: $AB=BC=1$, $CA=2$

Assignment Project Exam Help

<https://powcoder.com>

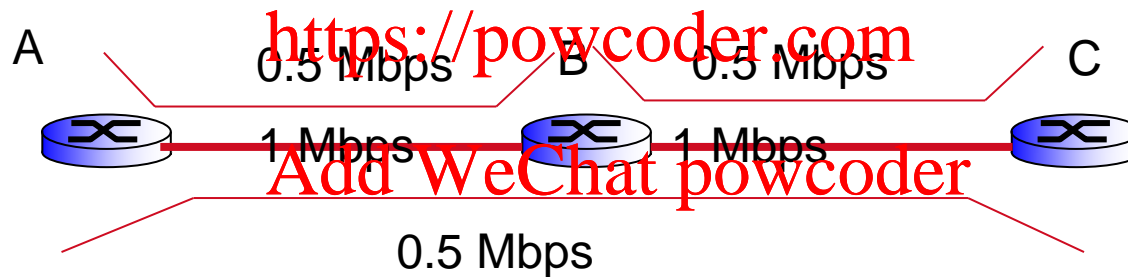
Add WeChat powcoder





More comment: Max-min fairness is too fair!

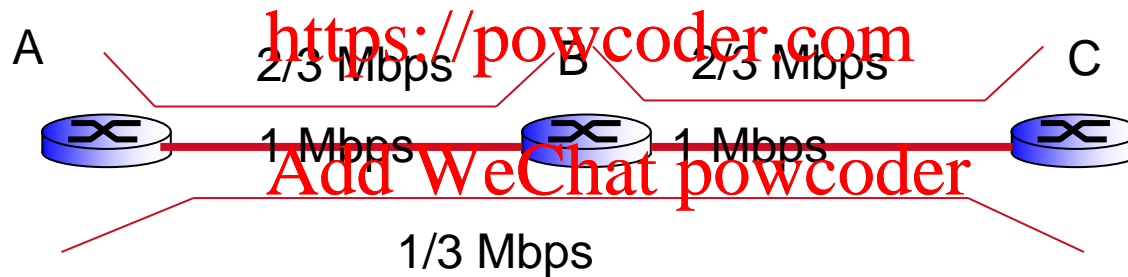
Assignment Project Exam Help



You are using two links. How can we get a same share?

Another form of fairness
proportional fairness

Assignment Project Exam Help



Longer routes are penalized



Can you solve the following problem?

link rate: $AB=BC=1$, $CA=2$

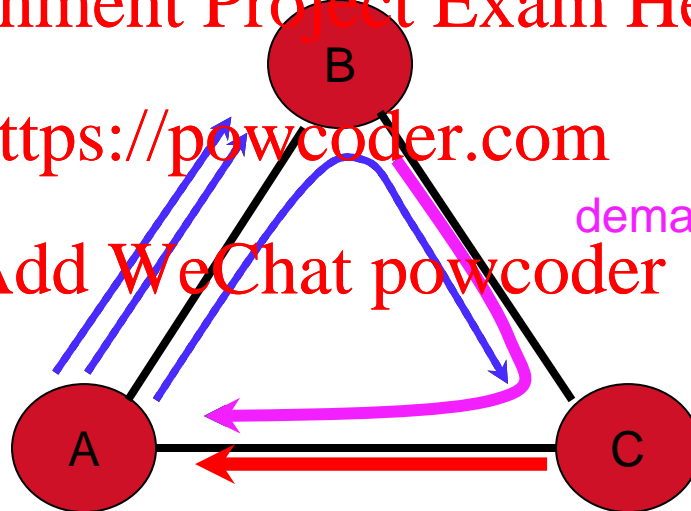
Assignment Project Exam Help

<https://powcoder.com>

demand 1,2,3 = $1/3$

demand 4 = $2/3$

Add WeChat powcoder



demand 5 = $4/3$



Assignment Project Exam Help The Application Layer

<https://powcoder.com>

Add WeChat powcoder

Some network applications

- › e-mail
- › voice over IP (e.g., Skype)
- › web
- › real-time video conferencing
- › text messaging
- › social networking
- › remote login
- › search
- › P2P file sharing
- › ...
- › multi-user network games
- › Add WeChat powcoder
- › streaming stored video
(YouTube, Netflix)

Assignment Project Exam Help

<https://powcoder.com>

Creating a network app

write programs that:

- › run on (different) *end systems*
- › communicate over network
- › e.g., web server software communicates with browser software

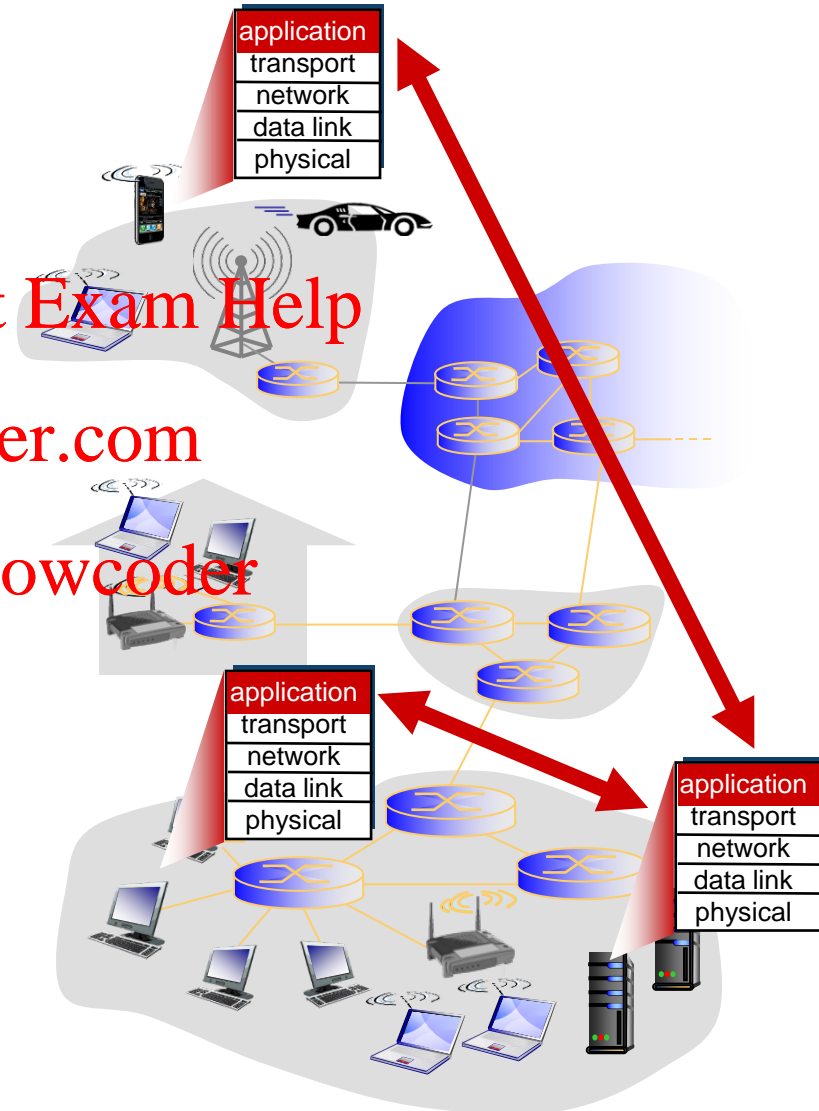
no need to write software for network-
core devices

- › network-core devices do not run user applications
- › applications on end systems allows for rapid app development, propagation

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Possible structure of applications

- › Client-server
- › Peer-to-peer (P2P)

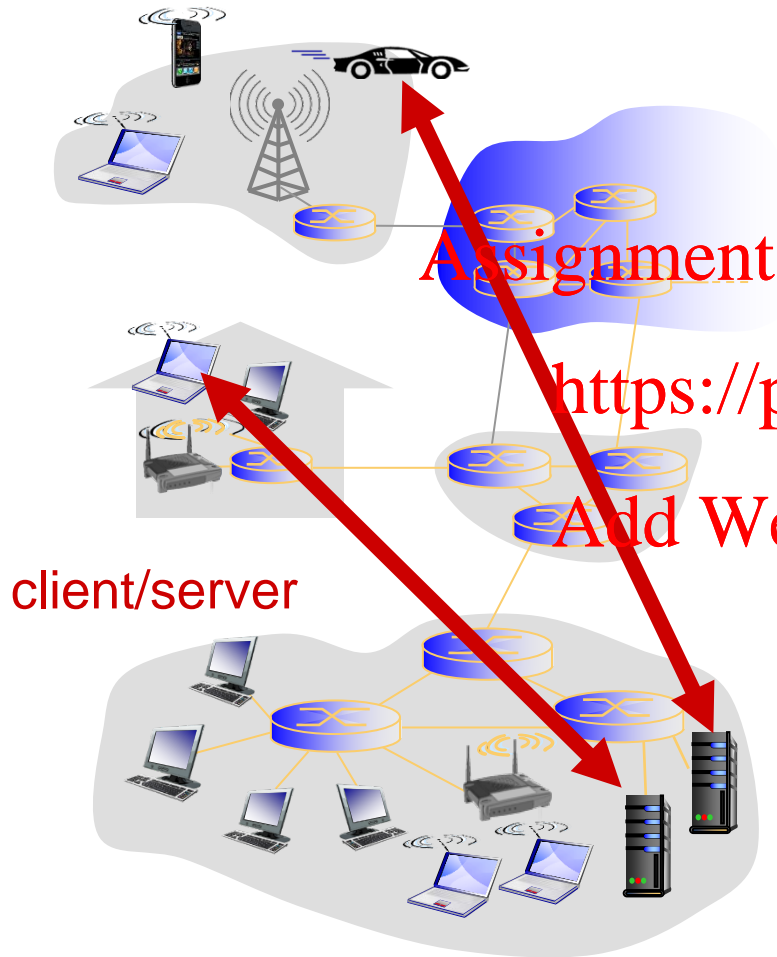
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Client-server architecture



server:

- › always-on
- › permanent IP address
- › data centers for scaling

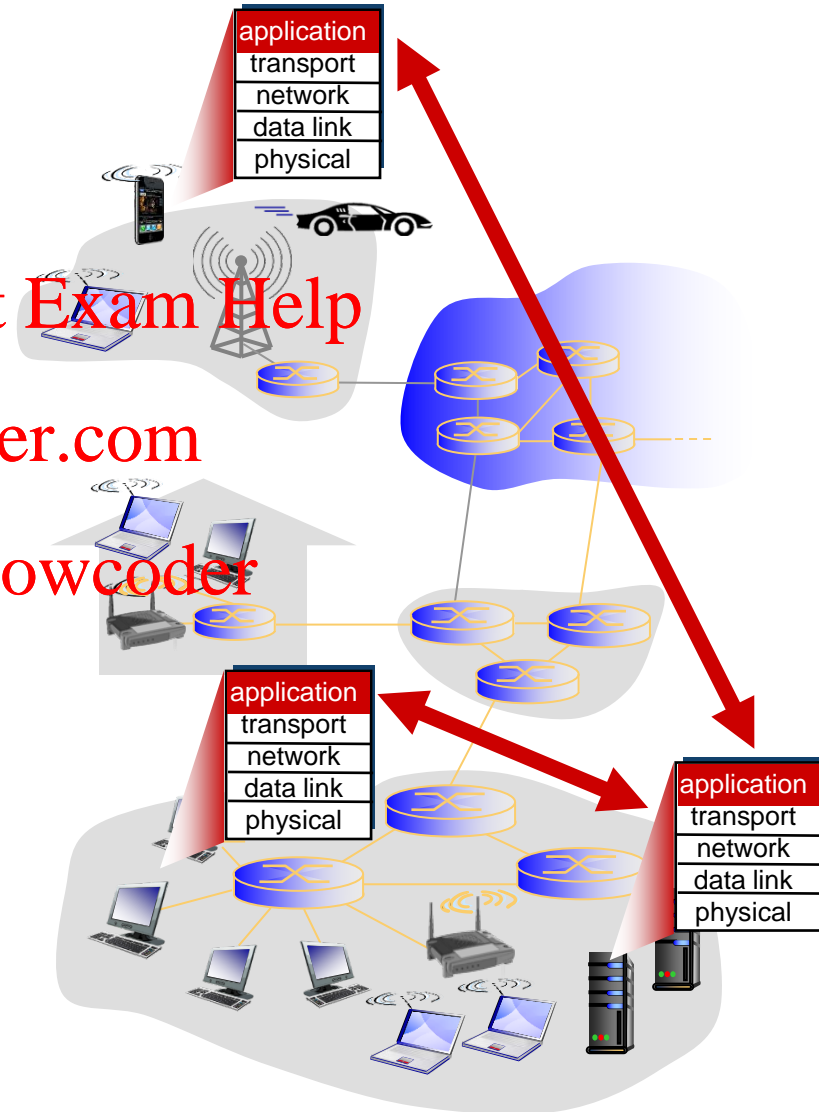
<https://powcoder.com>

clients:

Add WeChat [powcoder](https://powcoder.com) communicate with server

- › may be intermittently connected
- › may have dynamic IP addresses
- › do not communicate directly with each other

- › no always-on server
- › arbitrary end systems directly communicate
- › peers request service from other peers, provide service in return to other peers
 - *self scalability* – new peers bring new service capacity, as well as new service demands
- › peers are intermittently connected and change IP addresses
 - complex management



Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

process: program running
within a host

- › within same host, two processes communicate using **inter-process communication** (defined by OS)

- › processes in different hosts communicate by exchanging **messages**

clients, servers

client process: process that initiates communication

server process: process that waits to be contacted

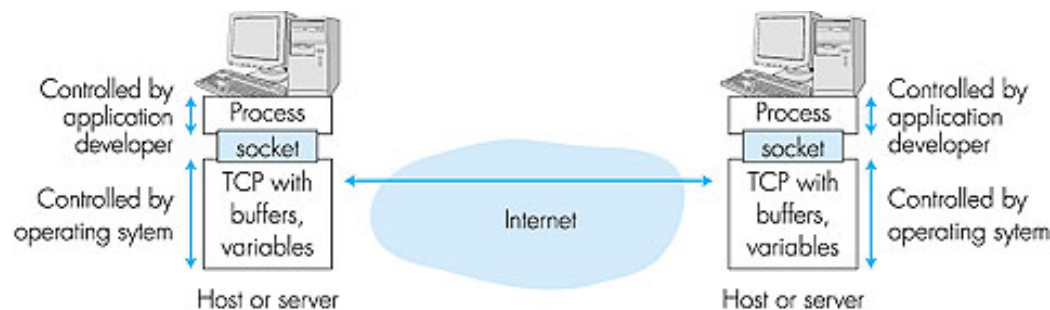
- ❖ aside: applications with P2P architectures have client processes & server processes

- › process sends/receives messages to/from its **socket**
- › socket analogous to door
 - sending process shoves message out door
 - sending process relies on transport infrastructure on other side of door to deliver message to socket at receiving process

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



- › to receive messages, process must have *identifier*
- › host device has unique 32-bit IP address (or 128 in IPv6)
- › Q: does IP address of host on which process runs suffice for identifying the process?
- A: no, *many* processes can be running on same host
- › *identifier* includes both **IP address** and **port numbers** associated with process on host.
- › example port numbers:
 - HTTP server: 80
 - mail server: 25
- › to send HTTP message to gaia.cs.umass.edu web server:
 - **IP address**: 128.119.245.12
 - **port number**: 80
- › more shortly...

App-layer protocol defines

- › types of messages exchanged,
 - e.g., request, response
 - › message syntax:
 - what fields in messages & how fields are delineated
 - e.g. First line: method. Second line: URL
 - message semantics
 - meaning of information in fields
 - e.g. 404 means “not found”
 - › rules for when and how processes send & respond to messages
- open protocols:
- › defined in RFCs
 - › allows for interoperability
- e.g., HTTP, SMTP
- proprietary protocols:
- › e.g., Skype

What transport service does an app need?

data integrity

- › some apps (e.g., file transfer, web transactions) require 100% reliable data transfer
- › other apps (e.g., audio) can tolerate some loss

timing

- › some apps (e.g., Internet telephony, interactive games) require low delay to be “effective”

throughput

- ❖ some apps (e.g., multimedia) require minimum amount of throughput to be “effective”
- ❖ other apps (“elastic apps”) make use of whatever throughput they get

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

TCP service:

- › *reliable transport* between sending and receiving process
- › *flow control*: sender won't overwhelm receiver
- › *congestion control*: throttle sender when network overloaded
- › *does not provide*: timing, minimum throughput guarantee
- › *connection-oriented*: setup required between client and server processes

UDP service:

- › *unreliable data transfer* between sending and receiving process
- › *does not provide*: reliability, flow control, congestion control, timing, throughput guarantee, or connection setup,

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Internet apps: application, transport protocols

application	application layer protocol	underlying transport protocol
email	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
file transfer	FTP [RFC 959]	TCP
streaming multimedia	HTTP RTP [RFC 1889]	TCP or UDP
Internet telephony	SIP, RTP, proprietary (e.g., Skype)	TCP or UDP



Assignment Project Exam Help
Web and HTTP

<https://powcoder.com>

Add WeChat powcoder

First, a review...

- › web page consists of *base HTML-file* which includes *several referenced objects*

Assignment Project Exam Help

- *HTML: HyperText Markup Language*

- › object can be JPEG image, Java applet, audio file,...
- › each object is addressable by a *URL (Uniform Resource Locator)*, e.g.,

<https://powcoder.com>

Add WeChat powcoder

`www.someschool.edu/someDept/pic.gif`

host name

path name



File: usually base-html file
(HyperText Markup Language)

xxxxxxxxxx
www.aaa.edu/Obj1.jpg
yyyyyyyyyyyyyy
www.aaa.edu/Obj2.jpg
zzzzzzzzzz

Browser shows



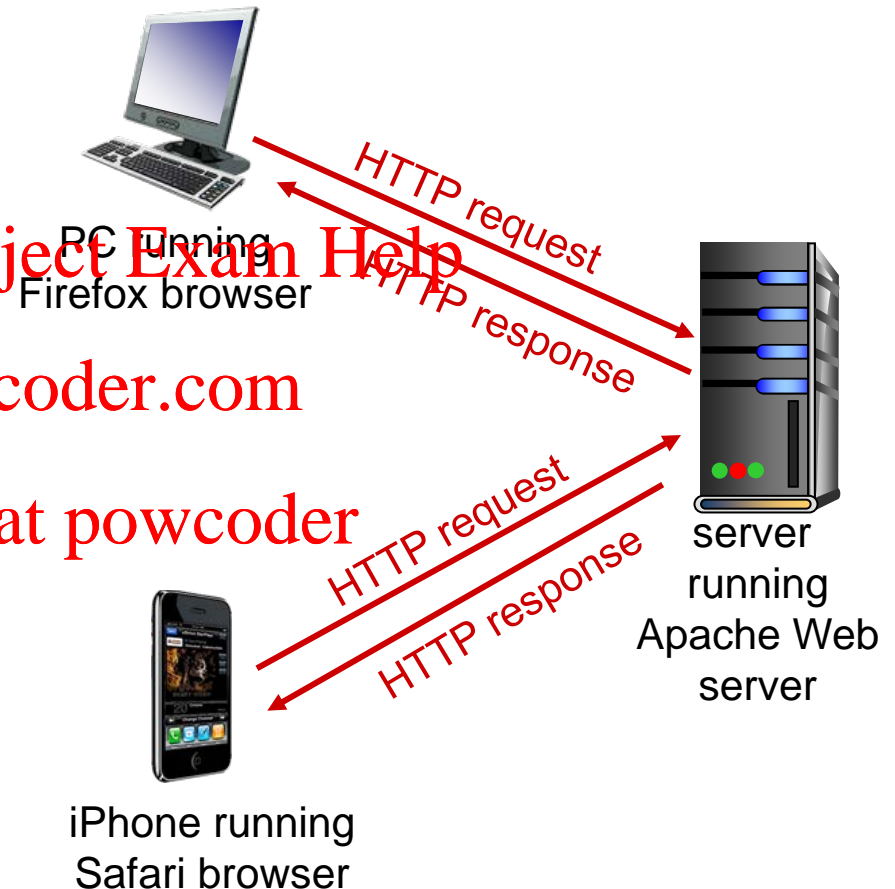
Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

HTTP: hypertext transfer protocol

- › Web's application layer protocol
- › client/server model
 - **client**: browser that requests, receives, (using HTTP protocol) and “displays” Web objects
 - **server**: Web server sends (using HTTP protocol) objects in response to requests



Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

uses *TCP*:

- › client initiates TCP connection (creates socket) to server, port 80

- How to know IP address?
- DNS (Domain Name System)

- › server accepts TCP connection from client

- › HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)

- › TCP connection closed

HTTP is “stateless”

- › server maintains no information about past client requests

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

protocols that maintain “state” are complex!

- ❖ past history (state) must be maintained
- ❖ if server/client crashes, their views of “state” may be inconsistent, must be reconciled

aside

non-persistent HTTP

- › at most one object sent over TCP connection

- connection then closed

- › downloading multiple objects required multiple connections

persistent HTTP

- › multiple objects can be sent over single TCP connection

- connection between client, server

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

suppose user enters URL: `www.someSchool.edu/someDepartment/home.index` (contains text, references to 10 jpeg images)

1a. HTTP client initiates TCP connection to HTTP server (process) at `www.someSchool.edu` on port 80

1b. HTTP server at host `www.someSchool.edu` waiting for TCP connection at port 80. "accepts" connection, notifying client

2. HTTP client sends HTTP *request message* into TCP connection socket. Message indicates that client wants page `someDepartment/home.index`

3. HTTP server receives request message, forms *response message* containing requested page, and sends message

5. HTTP client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects to download

4. HTTP server closes TCP connection.

time



suppose user enters URL: `www.someSchool.edu/someDepartment/home.index` (contains text, references to 10 jpeg images)

1a. HTTP client initiates TCP connection to HTTP server (process) at `www.someSchool.edu` on port 80

1b. HTTP server at host `www.someSchool.edu` waiting for TCP connection at port 80. "accepts" connection, notifying client

2. HTTP client sends HTTP request message into TCP connection socket. Message indicates that client wants object `someDepartment/object1.jpg`

3. HTTP server receives request message, forms response message containing requested object, and sends message

5. HTTP client receives response message containing object, displays the object.

4. HTTP server closes TCP connection.

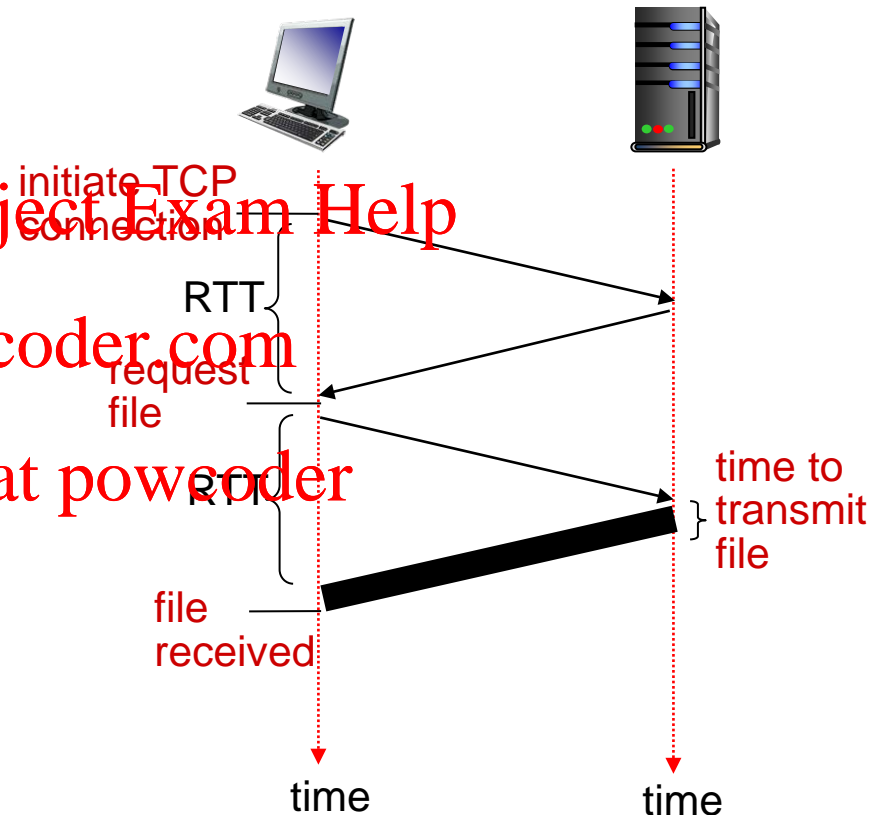
time ↓
6. Steps 1-5 repeated for each of 10 jpeg objects

RTT (definition): time for a small packet to travel from client to server and back

HTTP response time:

- › one RTT to initiate TCP connection
- › one RTT for HTTP request and first few bytes of HTTP response to return
- › file transmission time
- › non-persistent HTTP response time =

$2RTT + \text{file transmission time}$



suppose user enters URL: `www.someSchool.edu/someDepartment/home.index` (contains text, references to 10 jpeg images)

1a. HTTP client initiates TCP connection to HTTP server (process) at `www.someSchool.edu` on port 80

1b. HTTP server at host `www.someSchool.edu` waiting for TCP connection at port 80. "accepts" connection, notifying client

2. HTTP client sends HTTP *request message* into TCP connection socket. Message indicates that client wants page `someDepartment/home.index`

3. HTTP server receives request message, forms *response message* containing requested page, and sends message

5. HTTP client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects to download

TCP is still on

time





suppose user enters URL:

`www.someSchool.edu/someDepartment/home.index`

(contains text,
references to 10
jpeg images)

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

2. HTTP client sends HTTP *request message* into TCP connection socket. Message indicates that client wants object
`someDepartment/object1.jpg`

3. HTTP server receives request message, forms *response message* containing requested object, and sends message

4. HTTP client receives response message containing object, displays the object.

Repeated for each of 10 jpeg objects

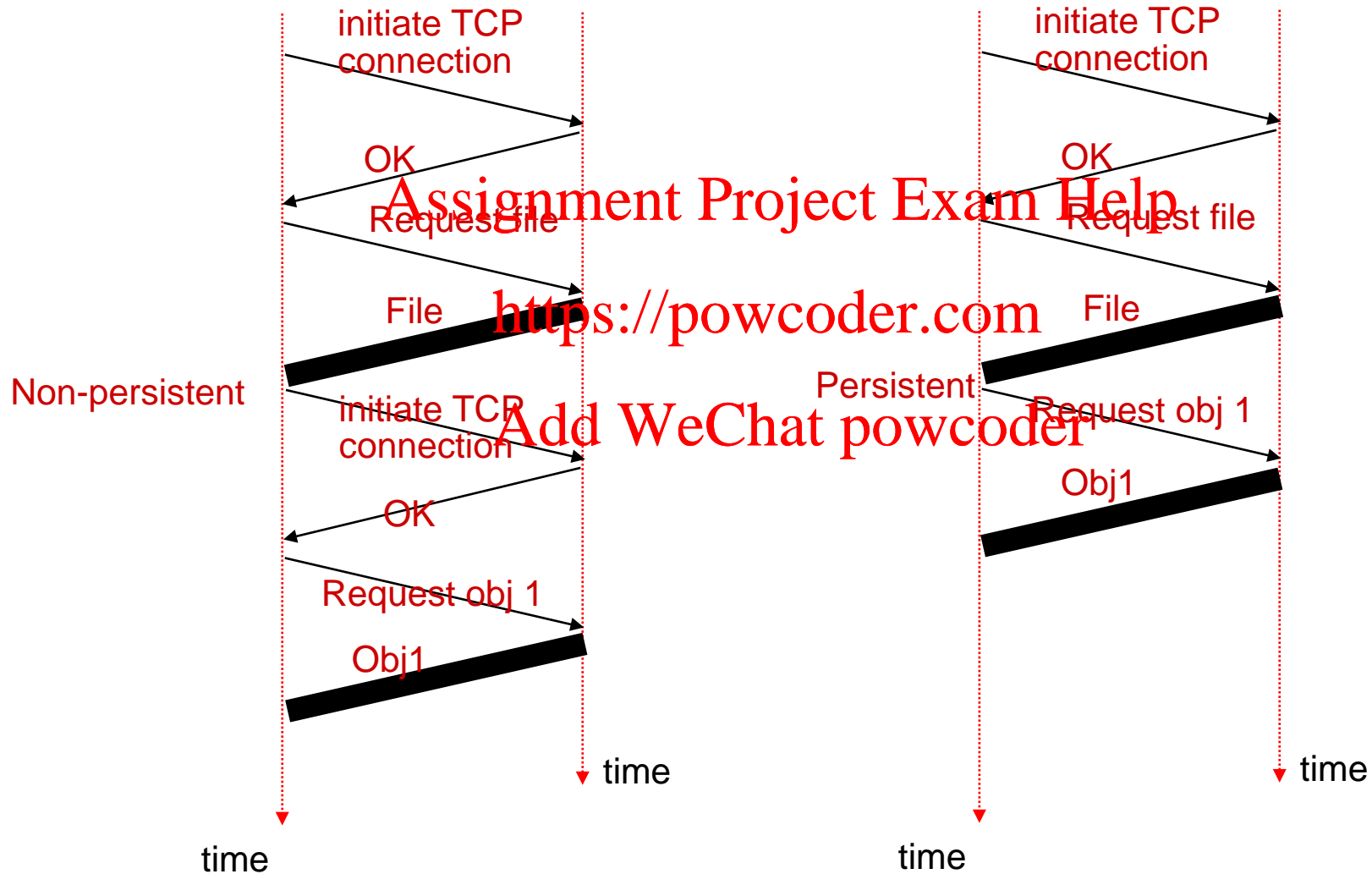
10 rounds later HTTP server closes TCP connection.

time





Non-persistent vs. persistent



non-persistent HTTP issues:

- › requires 2 RTTs + file transmission time per object

persistent HTTP:

- › server leaves connection open after sending response
- › subsequent HTTP messages between same client/server sent over open connection
- › client sends requests as soon as it encounters a referenced object
- › as little as one RTT + file transmission time for all the referenced objects

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat, powcoder

› two types of HTTP messages: *request, response*

› HTTP request message:

- ASCII (human-readable format)

Assignment Project Exam Help

request line
(GET, POST,
HEAD commands)

header
lines

carriage return,
line feed at start
of line indicates
end of header lines

<https://powcoder.com>

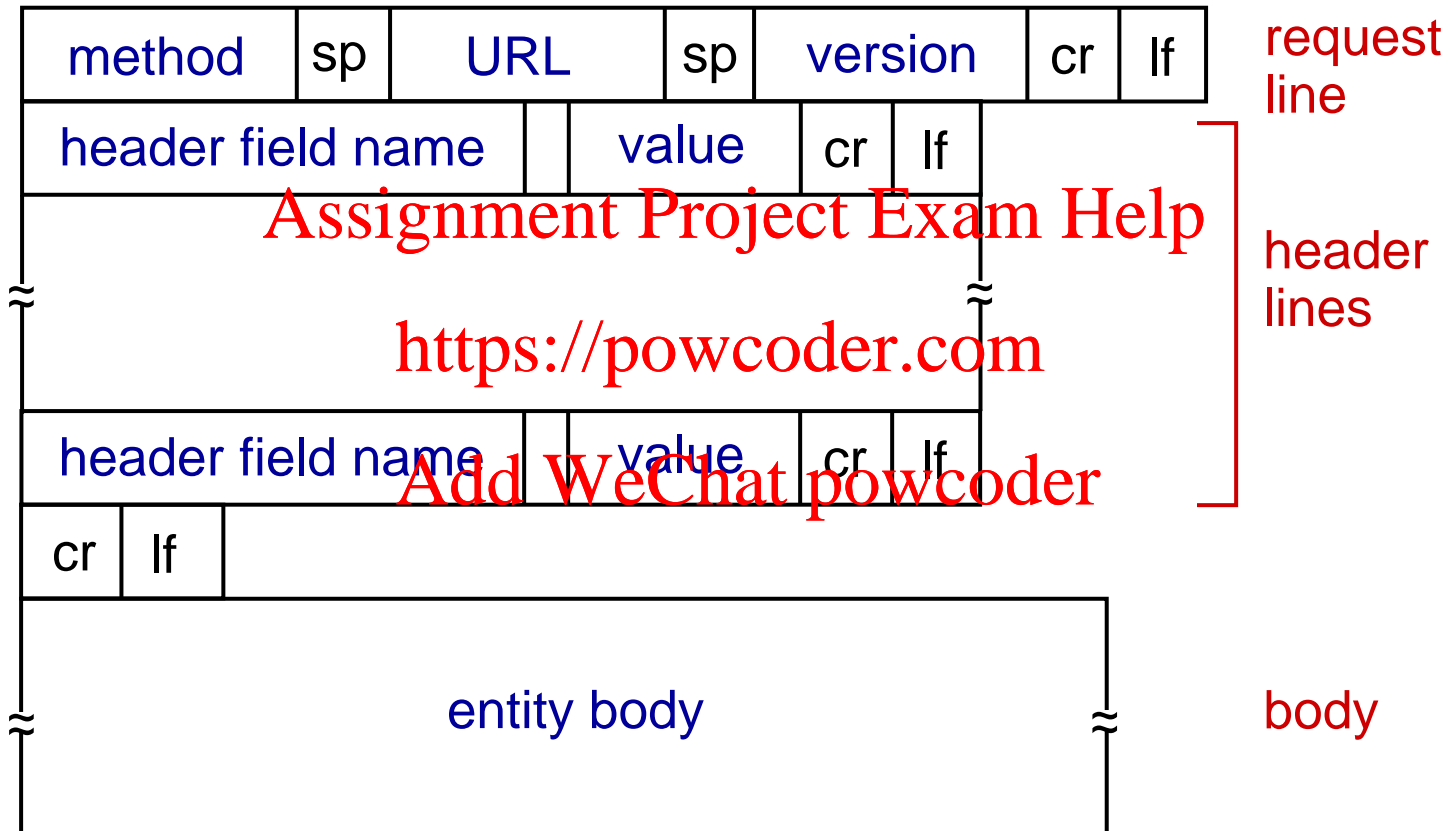
```
GET /index.html HTTP/1.1\r\n
Host: www-net.cs.umass.edu\r\n
User-Agent: Firefox/3.6.10\r\n
Accept: text/html,application/xhtml+xml\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip,deflate\r\n
Accept-Charset: ISO-8859-1,utf-8;q=0.7\r\n
Keep-Alive: 115\r\n
Connection: keep-alive\r\n
\r\n
```

carriage return character

line-feed character



HTTP request message: general format





GET method

Assignment Project Exam Help

<https://powcoder.com>

POST method: Add WeChat powcoder

- › web page often includes form input
 - › input is uploaded to server in entity body
-

HTTP/1.0:

- › GET
- › POST
- › HEAD

- asks server to leave requested object out of response

HTTP/1.1:

- › GET, POST, HEAD
- › PUT

<https://powcoder.com>

Add WeChat powcoder

uploads file in entity body to path specified in URL field

- › DELETE

- deletes file specified in the URL field



HTTP response message

status line
(protocol
status code
status phrase)

header
lines

data, e.g.,
requested
HTML file

```
HTTP/1.1 200 OK\r\n
Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n
Server: Apache/2.2.12 (CentOS)\r\n
Last-Modified: Tue, 30 Oct 2007 17:00:02
GMT\r\n
ETag: "17dc6-a5c-bf716880"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 2652\r\n
Keep-Alive: timeout=10, max=100\r\n
Connection: Keep-Alive\r\n
Content-Type: text/html; charset=ISO-8859-
1\r\n
\r\n
data data data data data ...
```

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

- › status code appears in 1st line in server-to-client response message.
- › some sample codes:

200 OK Assignment Project Exam Help

- request succeeded, requested object later in this msg

301 Moved Permanently

- requested object moved, new location specified later in this msg (Location:)

400 Bad Request

- request msg not understood by server

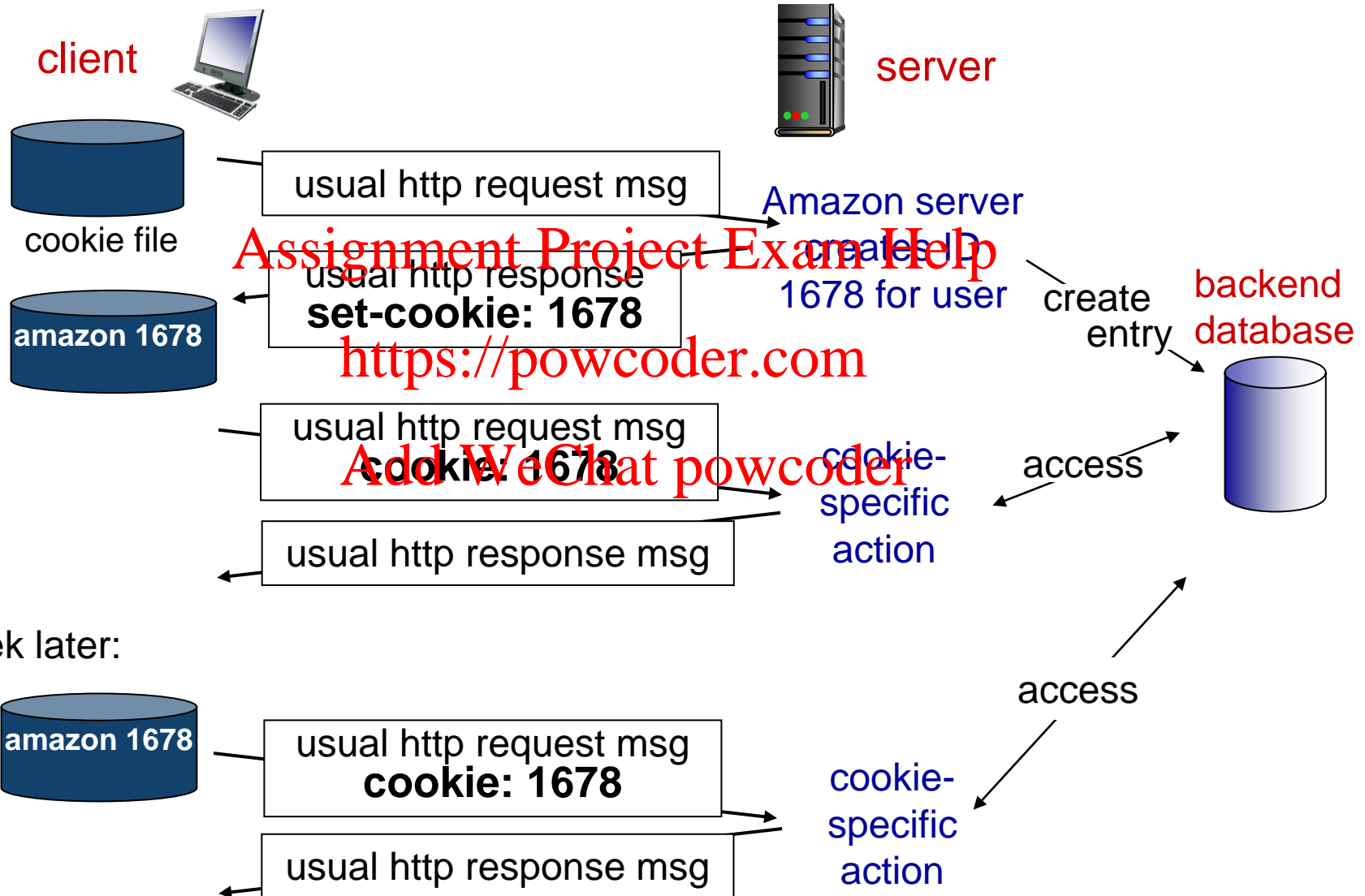
404 Not Found

- requested document not found on this server

505 HTTP Version Not Supported



Cookies: keeping “state” (cont’d)



many Web sites use cookies

four components:

- 1) cookie header line of HTTP response message
- 2) cookie header line in next HTTP request message
- 3) cookie file kept on user's host, managed by user's browser
- 4) back-end database at Web site

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

what cookies can be used for:

- › authorization
 - › shopping carts
 - › recommendations
 - › user session state (Web, e-mail)
- <https://powcoder.com>

Assignment Project Exam Help

Add WeChat powcoder

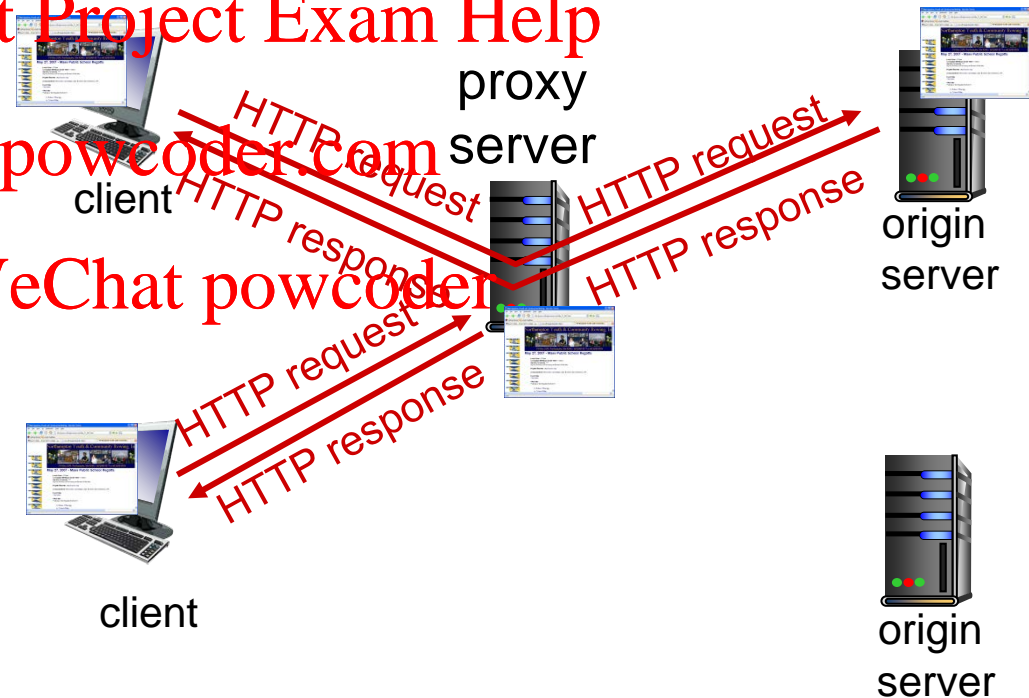
how to keep “state”:

- protocol endpoints: maintain state at sender/receiver over multiple transactions
 - cookies: http messages carry state
-

Web caches (proxy server)

goal: satisfy client request without involving origin server

- › user sets browser: Web accesses via cache
- › browser sends all HTTP requests to cache
- › if object in cache:
 - then cache returns object
 - else cache requests object from origin server, then returns object to client





› Q: Does the cache act as a client or a server?

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

› R: cache acts as both
client and server

- server for original requesting
client

- client to origin server

› typically cache is
installed by ISP
(university, company,
residential ISP)

why Web caching?

› reduce response time for
client request

› reduce traffic on an
institution's access link

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

assumptions:

- ❖ avg object size: 100K bits
- ❖ avg request rate from browsers to origin servers: 15/sec (1.5 Mbps service)
- ❖ RTT from institutional router to any origin server: 2 sec
- ❖ access link rate: 1.54 Mbps

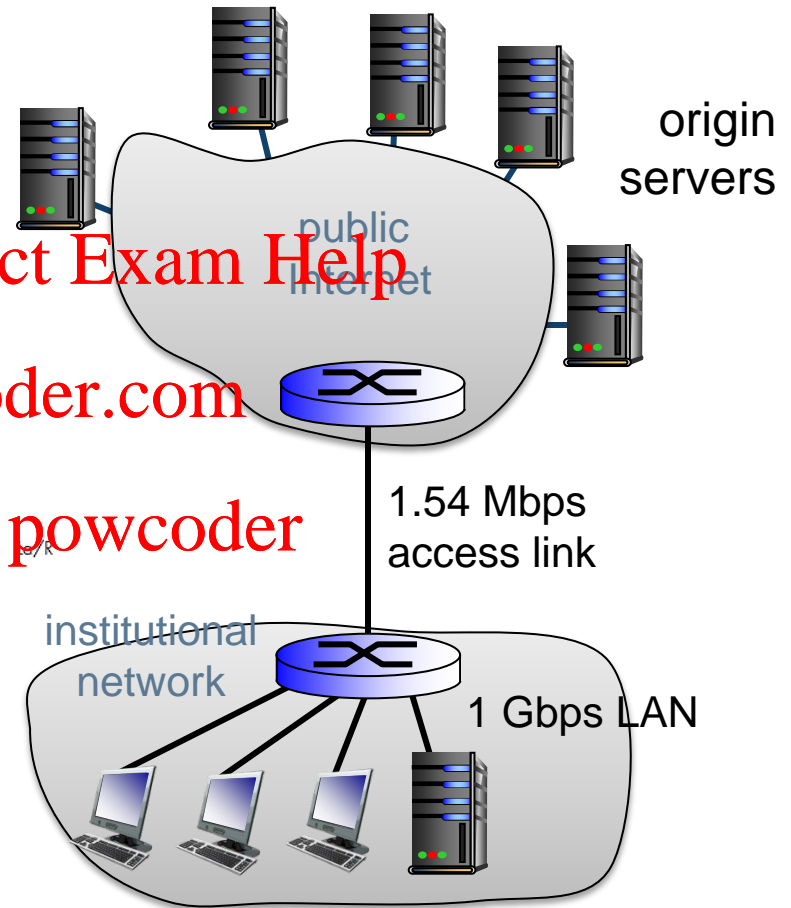
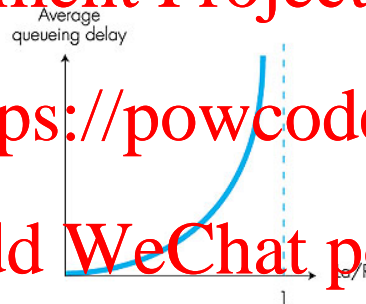
consequences:

- ❖ LAN utilization: 0.15%
- ❖ $LANU = \text{avg req rate} * \text{size} / \text{link bandwidth}$
- ❖ access link utilization = **99%** *problem!*
- ❖ $ALU = \text{avg req rate} * \text{size} / \text{link bandwidth}$
- ❖ total delay = 2 sec + minutes + usecs

Q: what happens with fatter access link?

<https://powcoder.com>

Add WeChat powcoder



Caching example: fatter access link

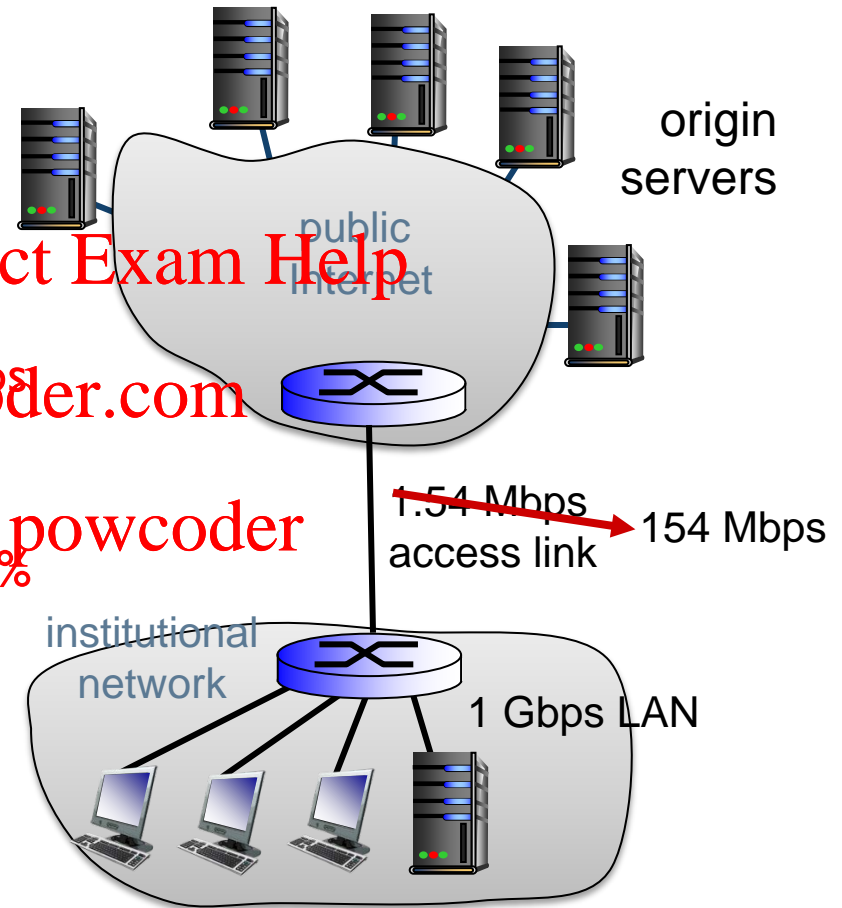
assumptions:

- ❖ avg object size: 100K bits
- ❖ avg request rate from browsers to origin servers: 15/sec
- ❖ RTT from institutional router to any origin server: 2 sec
- ❖ access link rate: 1.54 Mbps

consequences:

- ❖ LAN utilization: 0.15%
- ❖ access link utilization = 99%
- ❖ total delay = 2 sec + minutes + usecs

msecs



Cost: increased access link speed (not cheap!)

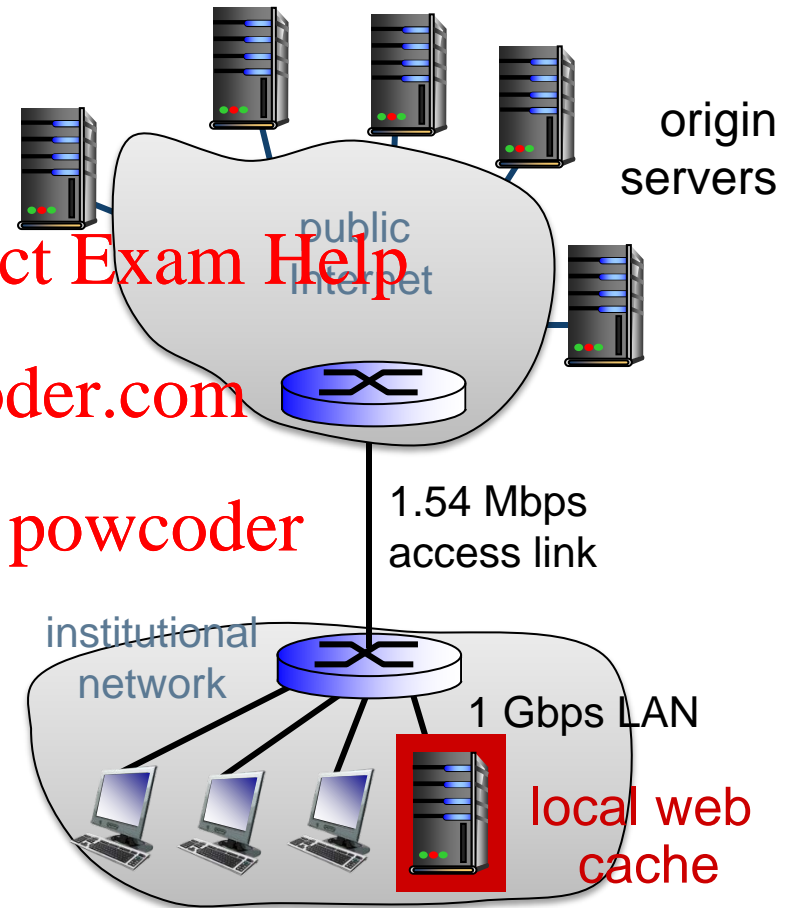
Caching example: install local cache

assumptions:

- ❖ avg object size: 100K bits
- ❖ avg request rate from browsers to origin servers: 15/sec
- ❖ RTT from institutional router to any origin server: 2 sec
- ❖ access link rate: 1.54 Mbps

consequences:

- ❖ LAN utilization: 0.15%
- ❖ access link utilization = 0%
- ❖ total delay = usecs



Cost: web cache (cheap!)

Caching example: install local cache

Calculating access link utilization, delay with cache:

- › suppose cache hit rate is 0.4
 - 40% requests satisfied at cache,
 - 60% requests satisfied at origin
- › access link utilization:
 - 60% of requests use access link
- › average total delay
 - = $0.6 * (\text{delay from origin servers}) + 0.4 * (\text{delay when satisfied at cache})$

Link utilization is around 60%, queueing delay is small enough

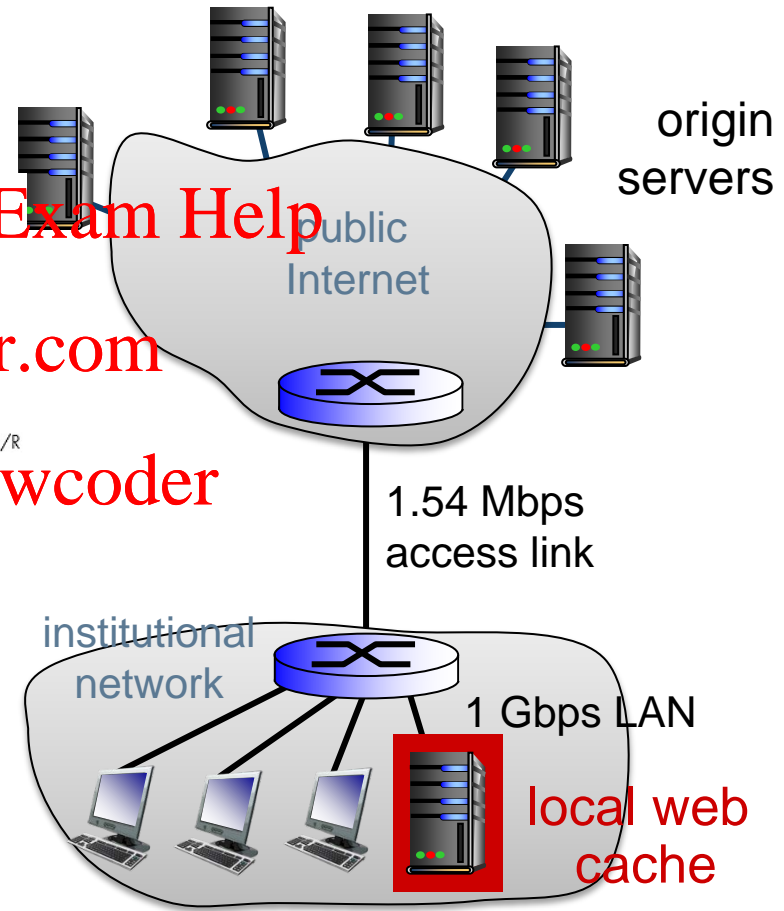
$$= 0.6 (\sim 2.x \text{ second}) + 0.4 (\sim \text{usecs})$$

less than with 154 Mbps link (and cheaper too!)

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



- › **Goal:** don't send object if client has up-to-date cached version

- no object transmission delay
- lower link utilization

- › **client:** specify date of cached copy in HTTP request

If-modified-since:
<date>

- › **server:** response contains no object if cached copy is up-to-date:

HTTP/1.0 304 Not
Modified

