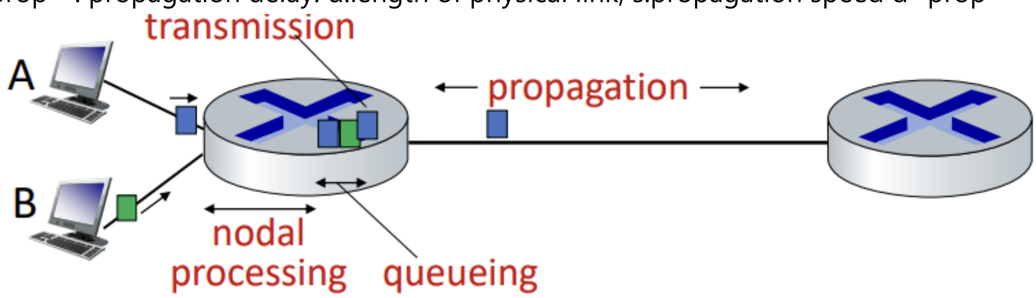
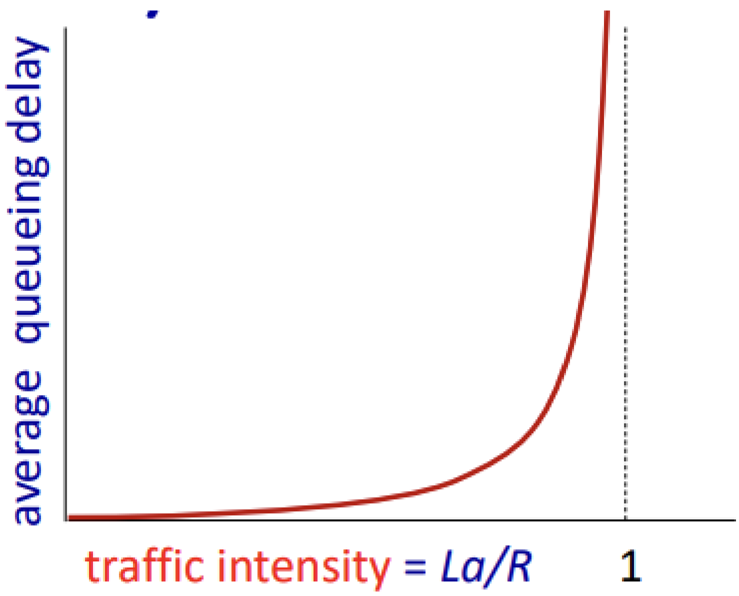
Points

Chapter 1

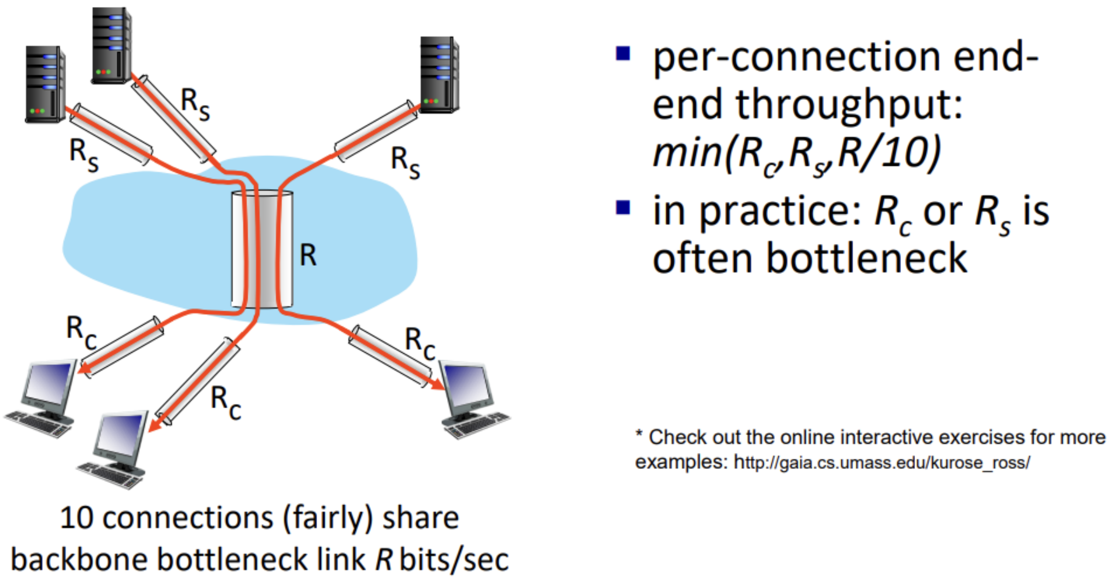
- Four sources of packet delay
    - d~nodal~ = d~proc~ + d~queue~ + d~prop~ + d~trans~
    - d~trans~ : packet transmission delay = time needed to transmit L-bit packet into link
        - L(bits)/R(bits/sec)
    - d~queue~ : Time waiting at output link for transfer (in buffer)
    - d~proc~ : nodal processing, check bit errors, determine output link
    - d~prop~ : propagation delay. d:length of physical link, s:propagation speed d~prop~ = d/s





- Packet queueing dealy
    - R: link bandwidth
    - L: packet length
    - a: average packet arrival rate
    - La/R ~ 0 : avg.queueing delay small
    - ->1 avg. queueing delay large
    - > 1 more "work" arriving is more than can be serviced - average delay infinite
- Throughput

    - rate at which bits are being sent from sender to receiver



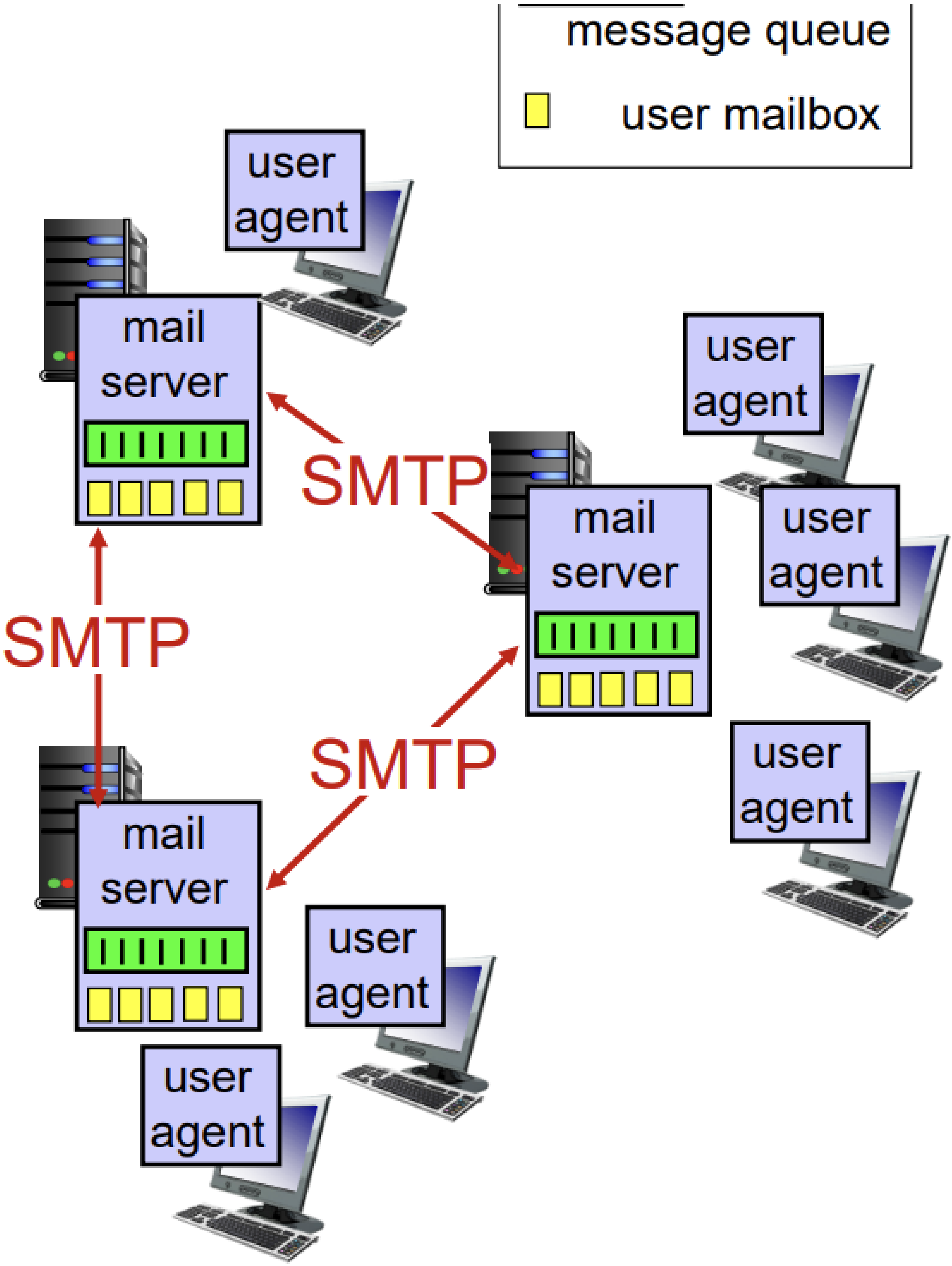
- Two key network-core functions
    - routing: determines source-destination route taken by packets
    - **forwarding: move packets from router's input to appropriate router output(Q)**

Application layer

- Internet transport protocols services
    - TCP
        - connection-oriented: set up required between client and server process
        - reliable transport: to deliver all data sent with out error and in the proper order
        - **flow control: sender won't overwhelm receiver**
        - congestion control: throttle sender when network overloaded
        - does not provide: timing, minimum throughpu guarantee, security
    - UDP
        - connectionless: there is no handshaking before the two processes start to communicate
        - unreliable data transfer
        - does not provide: reliability, flow control, congestion control, timing, throughput guarantee, security, or connection setup.

1. 低延迟：相对于TCP，UDP具有更低的传输延迟。TCP通过使用确认机制和重传机制来确保数据的可靠性，这些额外的机制会增加传输延迟。而UDP不使用这些机制，因此可以更快地将数据传输到目的地，适用于对实时性要求较高的应用。
2. 较少的开销：相比TCP，UDP在数据包头部的开销更小。TCP的头部包含了许多控制信息和确认字段，而UDP的头部相对较小，仅包含源端口、目的端口和数据长度等基本信息。这使得UDP在资源受限的环境中更加高效。
3. 简单性：UDP的设计更为简单，实现和使用也相对容易。它不需要像TCP那样维护连接状态、顺序控制和拥塞控制等复杂的机制。这使得UDP成为一些简单应用或特定场景下的首选，如DNS（域名系统）、音频/视频传输和实时游戏等。
4. 广播和多播：UDP支持广播和多播功能。广播是将数据包发送到网络中的所有主机，而多播是将数据包发送到特定的多播组。这使得UDP在需要向多个主机传输相同数据的应用中更为适用，例如视频流的组播传输。
5. 自定义性：由于UDP的灵活性，应用程序可以根据自己的需求进行定制和优化。UDP不提供内置的错误检测和纠正机制，这意味着应用程序可以自行处理错误检测和处理逻辑，从而实现更高效的数据传输。

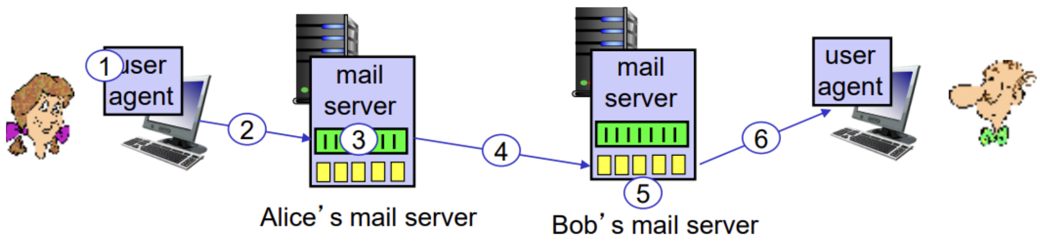
- Electronic mail



  - Three major components
    - user agents
    - mail servers
      - mailbox contains incoming messages for user
      - message queue of outgoing mail messages
    - **simple mail transfer protocol: SMTP**
      - A protocol used to send and reveice email messages between mail servers
      - SMTP uses TCP as its **underlying** transport protocol to provide the reliable data transfer service
      - Process

        - Client's SMTP mail server estabilishes a TCP connection to the recipients SMTP server using Port25
        - **3 phases of transfer: handshaking, transfer of messages, closure**
        - **messages must be in 7-bit ASCII**
    - Example

1) Alice uses UA to compose message "to" `bob@someschool.edu`
2) Alice's UA sends message to her mail server; message placed in message queue
3) client side of SMTP opens TCP connection with Bob's mail server
4) SMTP client sends Alice's message over the TCP connection
5) Bob's mail server places the message in Bob's mailbox
6) Bob invokes his user agent to read message



    - SMTP uses persistent connections
        - If the sending mail server has several messages to send to the same receiving mail server, it can send all of the messages over the same TCP connection
    - SMTP server uses CRLF.CRLF to determine end of message
- Comparison of HTTP and SMTP

Common characteristics

- Both are client-and-server Model
- Both use the reliable data transfer service of TCP
- Use persistent connection

Difference

- HTTP: pull protocol-someone loads information on a Web server and users use HTTP to pull the information from the server.
- SMTP: push protocol-the sending mail server pushes the file to the receiving mail server.
- SMTP has 7-bit ASCII restriction. HTTP does not have this kind of restriction.
- HTTP: each object encapsulated in its own response message
- SMTP: place all of message's objects into one message

Transport layer

- Checksum

1. Checksum field is used to check the bit-error occurring during the data transmission. Assuming that two 16-bit binary words are given as 1001000110110000 and 1110110101000110. Please calculate the checksum of the two 16-bit words. Write the detailed calculation steps.

Answer:

```
               1001000110110000
             + 1110110101000110
       Carry 1 0111111011110110
               0111111011110111;
Checksum is    1000000100001000;
```

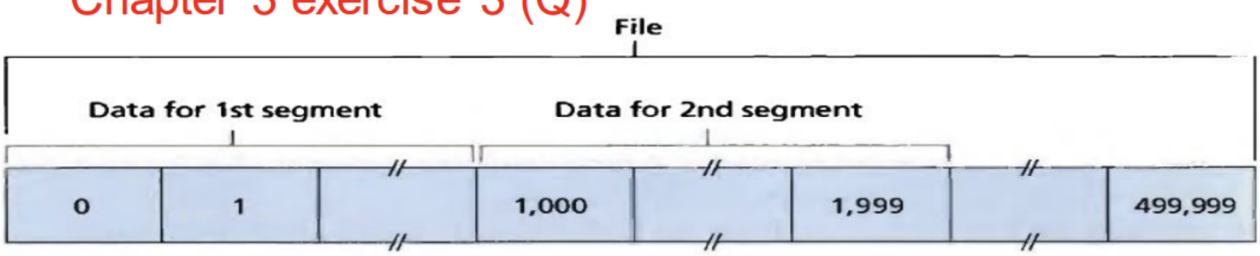
- Sequence Number

3. Suppose Host A sends two TCP segments back to back to Host B over a TCP connection. The first segment has sequence number X; the second has sequence number Y. How much data is in the first segment?

How to Decide the Sequence No.

- TCP views data as unstructured, but ordered stream of bytes.
- We label these bytes with integer numbers.
- Sequence number is the number of the first data in the segment in unit of bytes. Sequence numbers are over bytes, not segments.
- Example:
  - The data file consisting of 500,000 bytes, MSS is 1000bytes, the initial sequence number is 0.
  - TCP constructs 500 segments; the sequence number set in the first, second, third segments is 0, 1000, 2000, respectively.

Chapter 3 exercise 3 (Q)



- No 4

4. Consider sending a large file from one host to another over a TCP connection.
   a) Assume the initial congestion window is 1MSS (MSS=500 bytes), and approximately constant round-trip times is 0.5 second. Calculate the initial sending rate of the sender?

Answer:

Sending Rate $= \left\lceil \frac{500 \times 8}{0.5} \right\rceil = 8kbps$

   b) In slow start field, how long does it take for CongWin to increase from 1 MSS to 16 MSS in terms of RTT (assuming no loss events and constant RTT)? Give the reason.

Answer:

In the slow-start state, the value of cwnd begins at 1 MSS and grows exponentially.

So it takes totally 4RTTs to increase from 1 MSS to 16 MSS in terms of RTT (assuming no loss events and constant RTT).

- Datagrams

5. Suppose datagrams are limited to 1,500 bytes (including header) between source Host A and destination Host B. Assuming a 20-byte IP header and a 20-byte TCP header, how many datagrams would be required to send a MP3 file consisting of 2.5 million bytes? Assume the data is carried in TCP segments. Explain how you computed your answer.

Answer:

MP3 file size = 2.5 million bytes. Assume the data is carried in TCP segments, with each TCP segment also having 20 bytes of header. Then each datagram can carry 1500-40=1460 bytes of the MP3 file. $\frac{2.5 \times 10^6}{1460} = 1712.3$, so the number of datagrams required is 1712+1=1713.

- TCP Congestion Control
  - Congestion window(cwnd): a parameter to limit the transmission rate in sender
    - Sender limits transmission: the amount of unacknowledged data at a sender may not exceed the minimum of cwnd and rwnd

LastByteSent-LastByteAcked ≤ min{CongWin, rwnd}

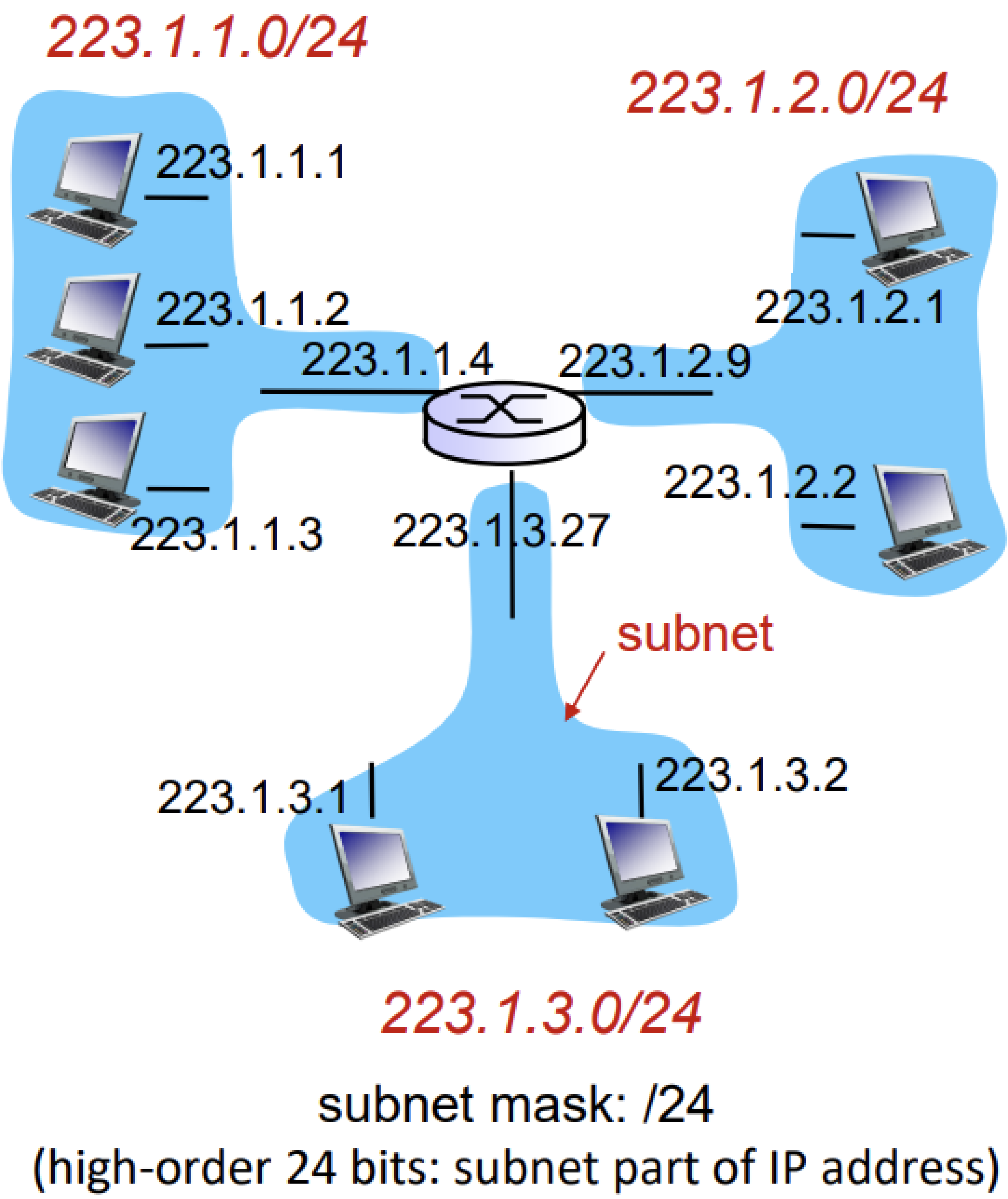
Roughly, sender's send rate is (Q)

$$\boxed{\text{rate} = \frac{\text{CongWin}}{\text{RTT}}\ \text{Bytes/sec}}$$

At the beginning of every RTT, the constraint permits the sender to send cwnd bytes of data into the connection; at the end of the RTT the sender receives acknowledgments for the data.

- Chapter 3, 4(Q), 5(Q)

Networklayer



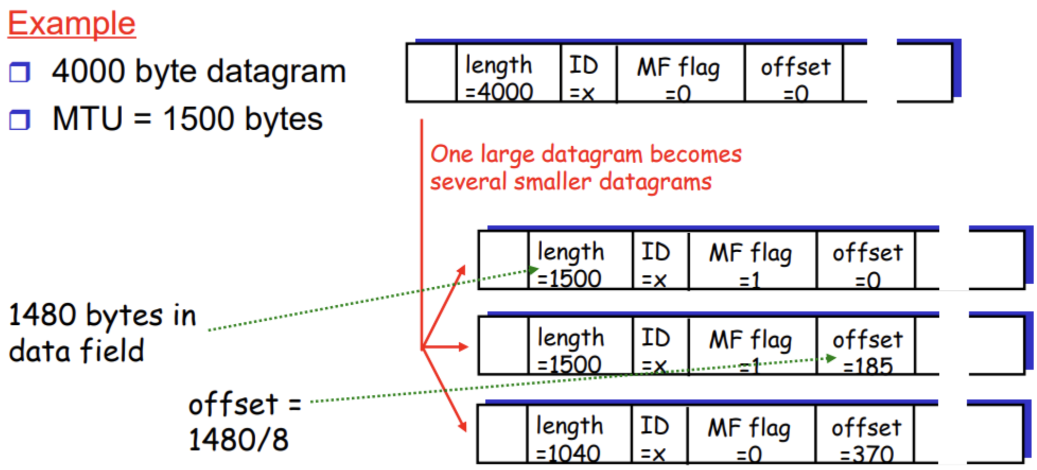
- Subnets
  - A subnet is a network where interfaces can physically reach each other **without passing through an intervenning router**
  - Dotted-decimal notation: a.b.c.d/x, where the notation /x is known as **subnet mask**, it indicate that **x leftmost bits of the 32-bit IP address** is the subnet part, and remaining (32-x) bits is the host part
- Longest prefix matching
  - when looking for forwarding table entry for given adestination address, use longest address prefix that matches destination address

Destination Address Range	Link interface
11001000 00010111 00010*** ********	0
11001000 00010111 00011000 ********	1
11001000 00010111 00011*** ********	2
otherwise	3

11001000 00010111 00010110 10100001 which interface?

11001000 00010111 00011000 10101010 which interface?

- IP fragmentation & Ressembly
    - Fragmentation
        - Divide a large IP datagram into 22 or more smaller IP datagrames
        - Encapsulate each of these smaller IP datagrams in a separate link-layer frames
        - Send these frames over the outgoing link
        - The process is called as fragmentation, and each of these smaller datagrams is referred to as a fragment
    - Reassembly
        - When these fragments arrive at their destination, the destination ressembles these fragments to reconstruct the original larger size datagram



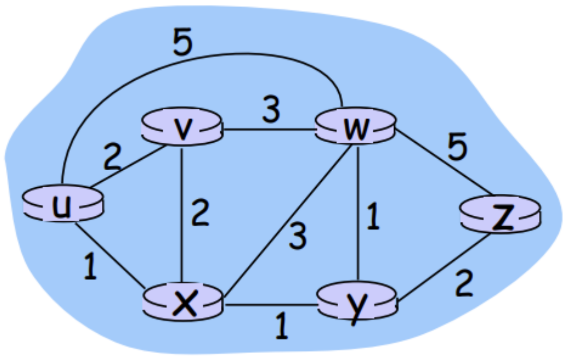
Steps:
1. Subtract 20 from original length: 4000 -20 = 3980 (bytes of "IP payload")
2. Subtract 20 from new MTU: 1500- 20 = 1480 (max. bytes of data in each fragment)
3. Divide "maximum data bytes" by 8 **to get offset increment (# chunks)**: 1480/8 = 185
4. Offset of each fragment "n" (n = 1, 2, 3, ...) = (n-1) x "offset increment": 0, 185, 370. ...
5. Length of each fragment (except the last fragment) = MTU = 1500 bytes.
   Length of last fragment = 20 + remaining data bytes = 20 + 3980 - 2 x 1480 = 1040.

        - MTU means max size

- Graph abstraction costs
    - Least-cost path: a path with the least cost
    - Shortest path: The path with the smallest number of links

- If all edges in the graph have the same cosst, the least-cost problem is also the shortest path problem

- Dijkstra's algorithm

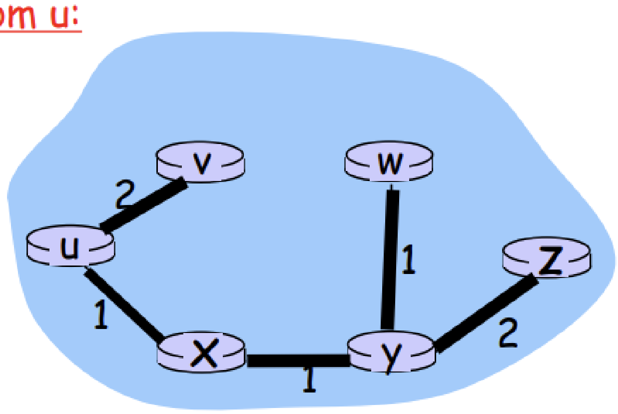


Questions:

Using Dijkstra's algorithm, list the shortest path with the cost from node * to each other node Note that you do **NOT** need to list the process of each iteration.

And show the forwarding Table

Resulting shortest-path tree from u:



Resulting forwarding table in u:

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

Link layer

- Multiple access protocols
  - MAC protocol

        - It is designed to coordinate the transmission of different nodes in order to minimize/avoid collission
    - **Goal**
        - Efficient and fair
            - When one node wants to transmit it can send at rate R, where R is the rate of broadcast link
            - When N nodes want to transmit, each can send at average rate R/N
        - Simple
            - Simple and easy to implement
        - decentralized
            - No special node is needed to coordinate transmissions
            - no synchronizzation among all nodes

- Slotted ALOHA
    - Assumptions
        - All frames consist of exactly fixed-size bits
        - time divided into equal size slots
        - If 2 or more nodes transmit in slot, all nodes detect collision
        - nodes are synchronized
    - Operation
        - when node obtains fresh frame, transmits in next slot
            - if no collision: node can send new frame in next slot
            - if collision: node retransmits frame in each subsequent slot with prob. p until success
    - Advantages:
        - Single active node can continuously transmit at full rate of channel
        - highly decentralized
            - nodes detect collision independently
            - node decides when to retransmit independently
        - simple
    - Disadvantages
        - collisions, wasting slots
        - idle slots
        - clck synchronization

- at best , channel used for useful transmissions 37% of time

Slotted ALOHA

Assume:

- Each node has an infinite number of packets to transmit with probability p

Questions:

1. What is the success probability of a particular node in a time slot？
2. What is the success probability of any node in a time slot？(Efficiency of Slotted ALOHA)
3. What is the failure probability of all nodes in a time slot？
4. What is the probability that the first success occurs in the k-th slot?

- Q1: answer: p*(1-p)^(N-1) (N is the total nodes number)
- Q2: answer: (p*(1-p)^(N-1))^N
- Q3: answer: 1 - P(success)
- Q4: answer: (1 - p)^(k-1) * p

- Switches vs. routers
  - Both are store-and-forward:
    - routers: network-layer devices(examine network-layer headers)
    - switches: link-layer devices(examine link-layer headers)
  - Both have forwarding tables:
    - routers: compute tables using routing algorithms, IP addresses
    - switches: learn forwarding table using flooding, learning, MAC addresses