Example Questions for Final Exam

CS 162 I. Stoica
Spring 2011

Store and Forward 1

Hosts A and B are connected to each other via router R. R is a store-and-forward router. The bandwidth from A to R is 10Mbps, and the bandwidth from R to B is 5Mbps. The one-way latency of each link is 22ms. Assume host A sends a 30KB file to host B.

a) Assume the file is divided into two packets, p1 and p2, where p1 has a length of 10KB, and assume the packets are sent back-to-back. What is the difference between the arrival times of the first and the second packet at host B?

b) What is the effective throughput between A and B in part (a)? (The transmission time is the time interval from the time the first bit is sent at A until the final bit is received at B).

c) Does the throughput increase or decrease if we divide the file into smaller packets? Why?

d) Now, assume each packet is acknowledged. The file is divided into 6 packets of the same size. How long would it take to send the entire file assuming that the sender cannot send a new packet before it receives an acknowledgment for the previous packet? (The transfer time is the time interval measure at source A from the time the first segment is sent until the acknowledgement of the last segment is received). Ignore the transmission time of the acknowledgements

Solution:

a) The difference is the transmission time of packet B over the second link
time: size/bw= 8*20/5000 = 160/5000 = 32ms

b) Total transfer-time: calculate how long the second packet takes and add the wait for the first packet over link A-R. A-R wait is: 8*10/10000 = 8ms
Now time to send B is 8*20/10000 + 8*20/5000 + 44ms = 16ms + 32ms + 44ms=92ms
Total time: 100ms
THROUGHPUT: 30k/100ms = 0.3MBps or 2.4 Mbits/s

c) The throughput will increase – because a smaller packet is received at R and can be forwarded immediately (don't have to wait till the entire *original* packet is transmitted over the first link to start transmitting over the slower second link)

d) Size of 1 packet = 5KB
transmit time /packet= 44 + 44ms + 5*8/10000+5*8/5000= 44ms +44ms + 4ms + 8ms = 100ms
so for 6 = 600ms

Store and Forwarding 2

Consider two packets that are sent back to back (i.e., one right after another) along the path A-B-C-D, where A, B, C, and D are store-and-forward routers. Assume the capacity of link (A, B) is 10Mbps, the capacity of (B, C) is 1Mbps, and the capacity of (C, D) is 100Mbps. The propagation delay along each link is 10 ms. Assume the size of the first packet is 1000 bits, and that the size of the second packet is 500 bits.

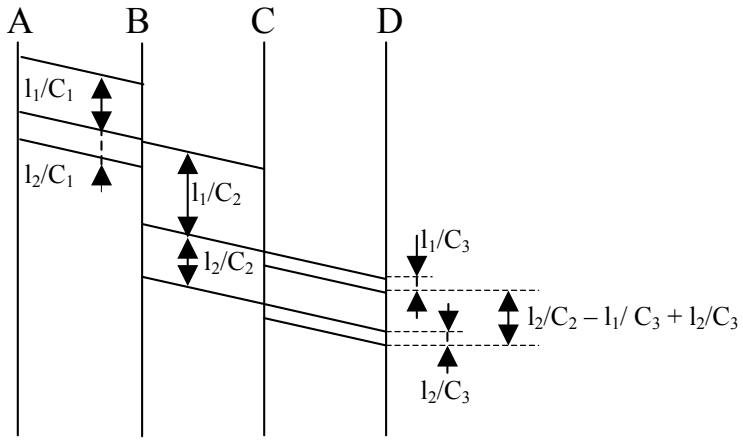
What is the inter-arrival time between the two packets at node D? The inter-arrival time is equal to the arrival time of the second packet minus the arrival time of the first packet at node D.

Notes: Assume there is no cross-traffic; the two packets are the only ones in the network. The arrival time of a packet is the time when the last bit of the packet was received. With a store-and-forward router a packet is forwarded only after the last bit of the packet was received.

Answer

See the time diagram below bellow (l_1 = 1000b, l_2 = 500b, C_1 = 10Mbps, C_2 = 1Mbps, C_3 = 100Mbps)

The inter-arrival time between the two packets at D is $l_2/C_2 - l_1/C_3 + l_2/C_3 = 500b/10^6Mbps - 1000b/10^8Mbps + 500b/10^8Mbps = 0.495$ ms

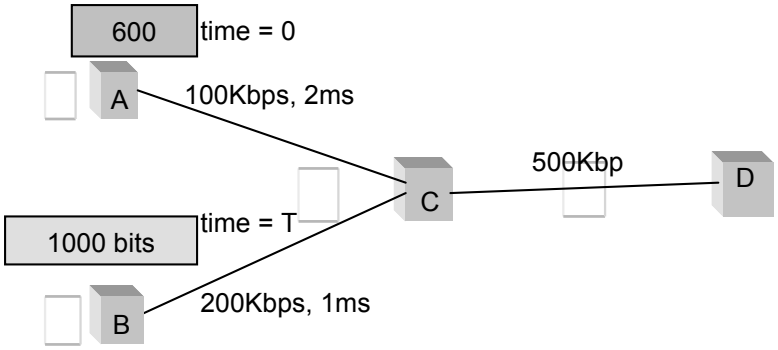


Store and Forwarding 3

Consider the network in the figure below where two source nodes A and B are connected to a destination node D through a router C. Assume that node A starts to send a 600 bit packet at time 0 and node B start to send a 1000 bit packet at time T (see figure below). Plot the inter-arrival time, denoted I, between the two packets at node D versus the starting time of B's packet, T for 0 <= T < 5.

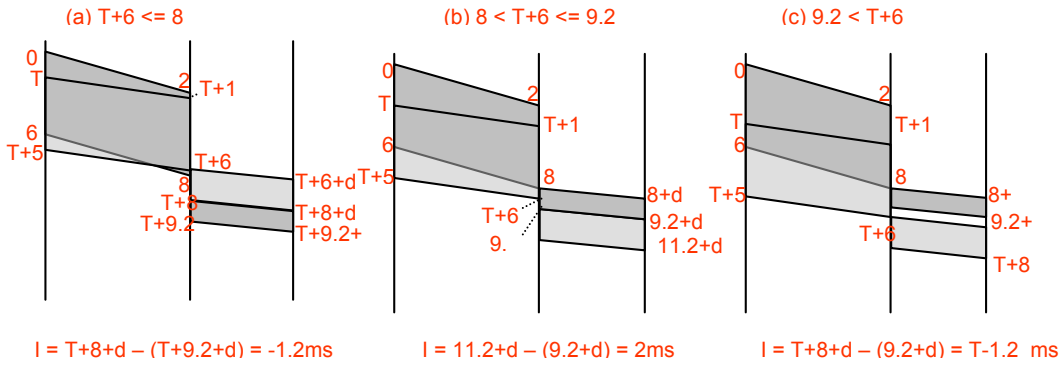
Notes: Ignore the processing time at C. The arrival time of a packet is the time when the last bit of the packet has arrived at node D. The inter-arrival time is

I = (arrival time of packet sent by B at D) – (arrival time of the packet sent by A at D).

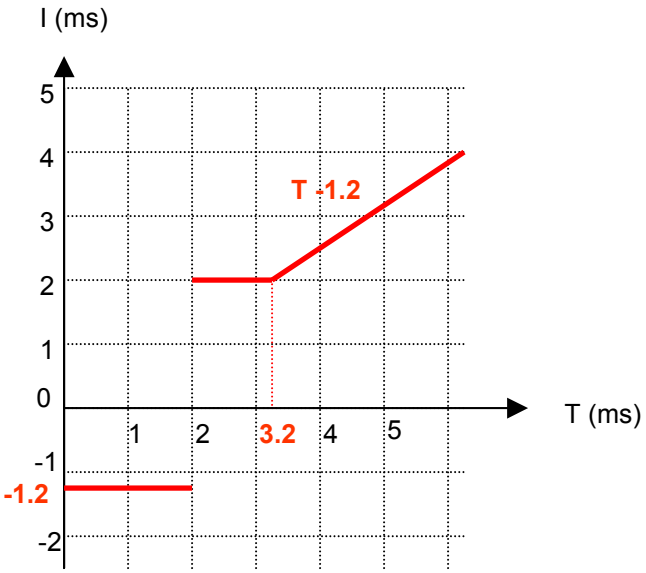


Hint: Based on the value of T, there are three cases you might want to consider. The diagram in Figure (a) shows the first case and depicts the messages sent by A and B arriving at C. You can use these diagrams to solve the problem, i.e., finish diagram (a) and fill in the other two diagrams (b) and (c).

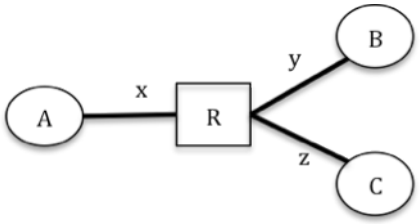
Solution:



Use the following coordinates to plot the inter-arrival time between the two packets at D (I) versus the starting time of B's packet (T), for 0 <= T <= 5.


I (ms)
5
4
3
2
1
0
-1
-2
T -1.2
1
2
3.2
4
5
T (ms)
-1.2

Store and Forward 4 (Measuring link capacities)
Consider the network below where three hosts A, B, and C, respectively, are connected to router R. The capacity of the links are x Kbps, y Kbps, and z Kbps, respectively.

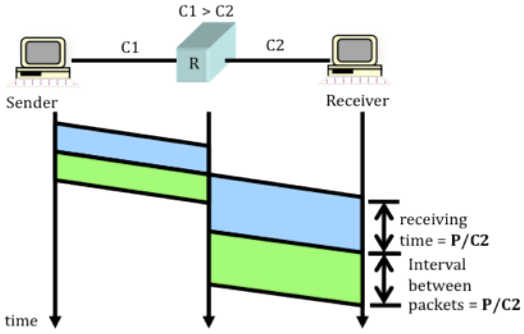


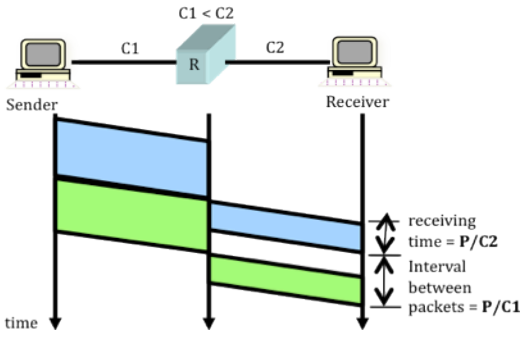
Assume the users of hosts A, B, and C, respectively, want to measure the capacities of the three links. To achieve this, they send the following messages:

- From A to B: Send two back-to-back 1Kbit packets; the interval between receiving the first and the second packet at B is 100ms; the time it takes B to receive each packet (i.e., the interval between receiving the first bit of the packet and the last bit of the packet) is 100ms.
- From B to C: Send two back-to-back 1Kbit packets; the interval between receiving the first and the second packet at C is 200ms; the time it takes B to receive each packet is 200ms.
- From C to A: Send two back-to-back 1Kbit packets; the interval between receiving the first and the second packet at A is 200ms; the time it takes B to receive each packet is 50ms.

Assume router R is store-and-forward, and the receiving time of a packet represents the time the last bit of the packet is received. Assume no other traffic in the network and a 0 packet processing delay.

a) (5 points) Based on the above measurements, compute capacities x, y, and z, respectively.





The above figure illustrates the impact of the relationship between link capacities on (i) the inter-arrival between receiving the two packets, and (ii) the time it takes to receiving a packet. The packet length is P bits.

Along path A—R—B, the bottleneck link is 1Kbit/0.1s = 10Kbps, and the capacity of the last link (i.e., R—B) is 1Kbit/0.1s = 10Kbps. Thus R—B is the bottleneck link and has the capacity of 10Kbps, while A—R has a capacity which is at least 10Kbps.

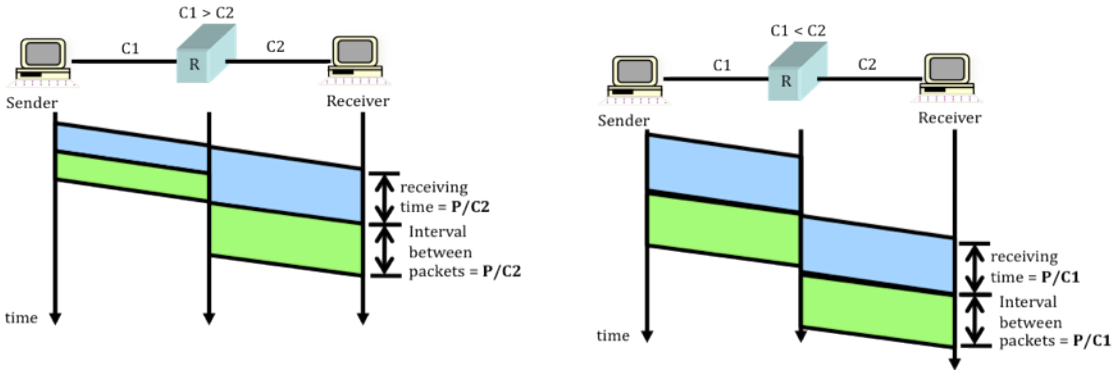
Along path B—R—C, the bottleneck link is 1Kbit/0.2s = 5Kbps, and the capacity of the last link (i.e., R—C) is 1Kbit/0.2s = 5Kbps. Thus R—C is the bottleneck link and has the capacity of 5Kbps.

Finally, along path C—R—A, the bottleneck link is 1Kbit/0.2s = 5Kbps, and the capacity of the last link (i.e., R—A) is 1Kbit/0.05s = 20Kbps.

Thus, A—R: 20Kbps, A—B: 10Kbps, A—C: 5Kbps

b) (5 points) Answer (a) assuming R is a cut-through router. Neglect the header size. (Note in the

class: For this point you ignore the times to receive a packet.)



The above figure illustrates the impact of the relationship between link capacities in the case of a cut-through router on (i) the inter-arrival between receiving the two packets, and (ii) the time it takes to receiving a packet. The packet length is P bits.

All the links along a path must transmit at the same rate, and thus transmission rate along a path is dictated by a bottleneck link. So,

min(X, Y) = 1Kbits / 100msec = 10Kbps
min(Y, Z) = 1Kbits / 200msec = 5Kbps
min(Z, X) = 1Kbits / 200msec = 5Kbps

Thus, Z = 5Kbps.
Note we cannot know exact values of X and Y though we know they must meet min(X, Y) = 10Kbps

Sliding Window 1

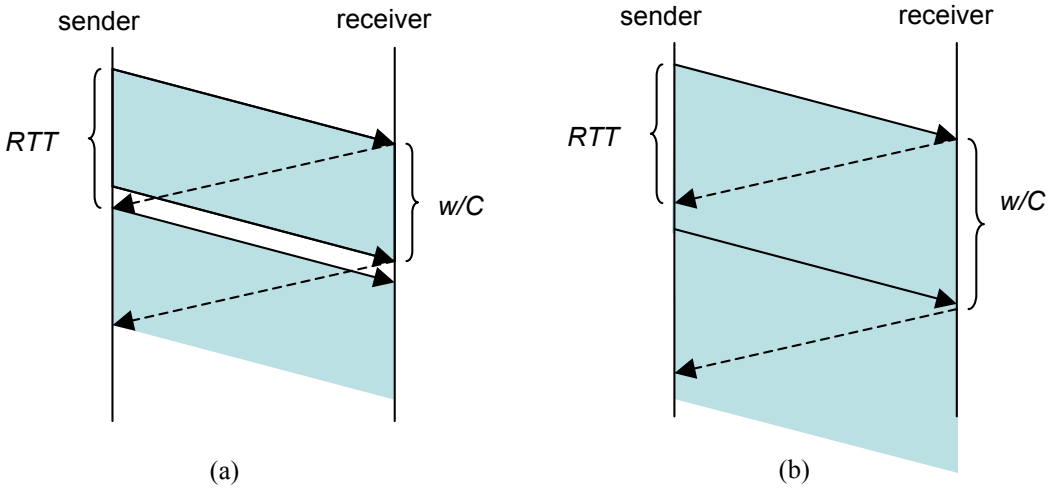
Assume two end-hosts communicate using the sliding window protocol. Assume the receiver window is always smaller than the sender's window and the size of the receiver window is *w* bits. Let *C* be the link capacity between the two end-hosts in bps, and RTT be the roundtrip time between the two end-hosts in sec. What is the maximum throughput achieved by the two end-hosts?

Note: Assume every bit is acknowledged.

Answer: There are two cases (see figure below)

case a: RTT > w/C, throughput = w/RTT

case b: RTT <= w/C, throughput = C



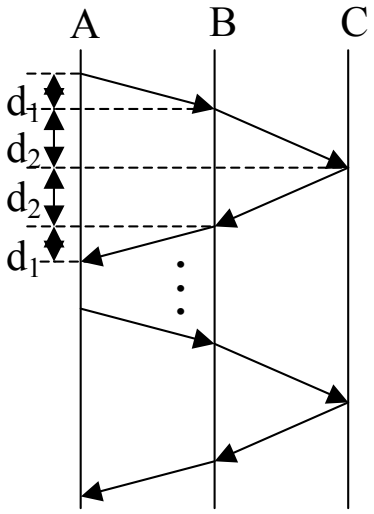
Sliding Window 2

Consider two links, (A,B) and (B,C), with propagation delays of d1 and d2, respectively. Assume that host A sends M packets to host C using a sliding window flow control protocol with a window of size W.

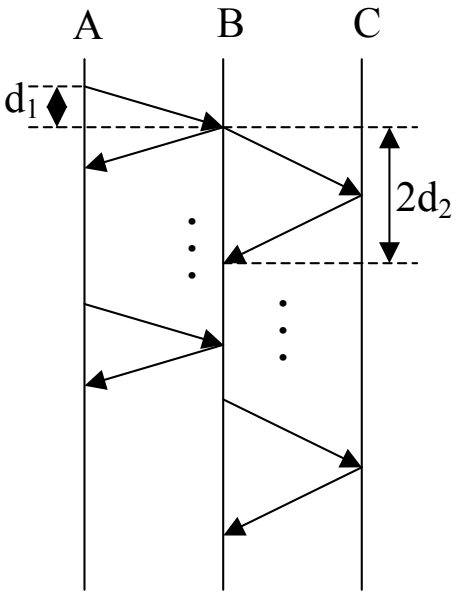
a) How much does it take to send all packets from A to C when the flow control protocol is implemented end-to-end between A and C? (5 pt)
b) How much does it take to send all packets from A to C when the flow control protocol is implemented on each link instead of end-to-end? (5 pt)

Notes: Ignore queueing delay and transmission times. The time it takes to transfer all packets is the difference between the time when the last acknowledgement is received and the time the sender sent the first packets.

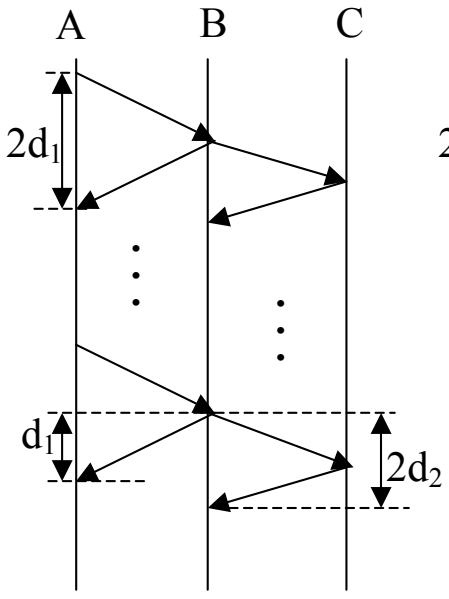
Answer:

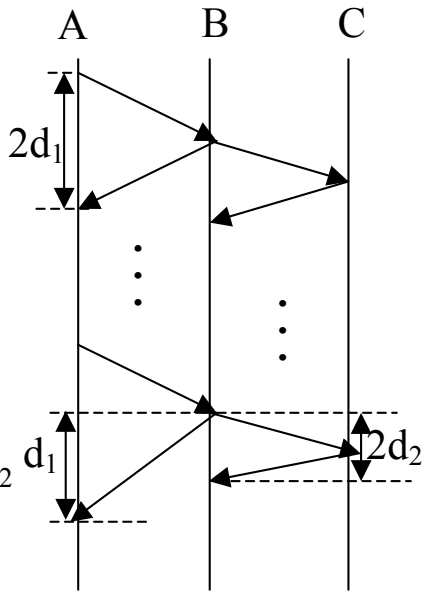


$$a) \quad 2 \times (d_1 + d_2) \times \left\lceil \frac{M}{W} \right\rceil$$



$$b)\ d1 \le d2 \Rightarrow d_1 + 2 \times d_2 \times \left\lceil \frac{M}{W} \right\rceil$$





$$d1 > d2 \Rightarrow (2 \times d_1) \times \left(\left\lceil \frac{M}{W} \right\rceil - 1 \right) + d_1 + \max(d_1, 2 \times d_2)$$

End-to-end argument
Use no more than three sentences to answer the questions below:

(a) State the “end-to-end arguments”.
(b) Error detection techniques detect corrupt bits. Give an example of an error that can be detected by the transport layer that the link layer cannot.
(c) Given that the transport layer can detect errors that the data link layer cannot, why would you implement error detection in the lower layers?

Solution:
(a) A lower layer should not implement functionality that can be correctly and completely implemented only by a higher layer. The only exception is when partial functionality in a lower layer optimizes performance significantly, and doesn't adversely affect applications at the higher layer that do not require

this functionality.

(b) The transport layer can detect errors that occur after a packet has passed the data link layer. For example, a packet can be corrupted in the buffer or the switching backplane of a router.

Another example is when a stronger error detection technique is used at the transport layer. Since the transport layer is implemented only at end-hosts and not on every router along the path, typically more computational resources are available at the transport layer.

Note: You have to give an example that some how involves corrupt bits. If you talk about lost or corrupt packets, you receive partial credit only.

(c) Implementing error detection at a lower layer can improve performance. The transport layer can only detect errors at the destination, and hence the packet with the error is carried over the entire path wasting bandwidth. The retransmission has to be done over the entire path too.

Another example is that different data link layer technologies have different error characteristics, so a high error rate technology (wireless) can have strong error detection, while a low error rate technology (optical) can have weak error detection. We can thus avoid the overhead of having strong error detection over the entire path.

Cryptography

(a) What is the main advantage of the public key cryptography (e.g., RSA) over the symmetric key cryptography (e.g., DES)? (5 pt)

(b) Assume that two hosts A and B use public key cryptography to ensure confidentiality. Enumerate the steps followed by the two hosts to send a message from A to B. (You don't need to give any formulas) (5 pt)

Answer:

(a) The main advantage of public key over symmetric key cryptography is that the public key cryptography does not require sending any secret key over the network.

(b)

1) The receiver (B) generates a public/private key pair and sends the public key to the sender.
2) The sender A encrypts the message using B's public key.
3) The receiver B decrypts the message using its private key.

Worm Propagation

Consider a network consisting of N nodes and the following worm propagation process. At time t=0, only one machine is infected. In each round every infected machine contacts k other machines and infects them. Assume that the worm is "smart" so that an infected machine never tries to infect another infected machine.

(a) If N=128 and k=2, how many rounds does it take to infect all the nodes in the system? (4 points)

Initially, there is only one infected node.
In the 1st round there are 1+2 infected nodes.
In the 2nd round there are 3+2*3 = 9 infected nodes.
In the 3rd round there are 9+9*2 = 27 infected nodes.
In the 4th round there are 27+27*2 = 78 infected nodes.
In the 5th round all nodes will be infected.

(b) Let r be the number of rounds it takes for all nodes in the system to get infected. Compute r as a function of N and k. (Note: this is a generalization of point (a)) (8 points)

Let x(i) be the number of infected nodes in round i. The number of nodes infected in round x(i+1) is then

$x(i+1) = x(i) + x(i)*k = x(i)*(1+k)$

Since x(0) = 1, we have

$x(i) = x(1)*(1+k)^i = (1+k)^i$

All nodes will be infected in round r such that x(r) >= N, that is,

$r = \lceil \log(N)/\log(k+1) \rceil$

Note that for N=128, k=2, we get r=5.

(a) Assume that the network can carry at most M packets in a round, where k < M < N. How many rounds r does it take to infect the entire network in this case as a function of k, M, N? (8 points)

It takes m rounds to infect the first M nodes, where

$m = \lceil \log(M)/\log(k+1) \rceil$

Since no more than M new nodes can be infected during every subsequent round, it takes n rounds to infect the rest of the nodes, where

$n = \lceil (N-M)/M \rceil$

Thus

$r = m + n = \lceil \log(M)/\log(k+1) \rceil + \lceil (N-M)/M \rceil$

Chord/DHT

How does the Chord lookup protocol achieve *efficiency* (i.e. packets reach their destination in as few steps as possible)?

By use of finger table. At each step the distance to the destination reduces by half, which means that the lookup terminates in a logarithmic number of steps.

a) How does Chord lookup protocol achieve *robustness* in the presence of random node failures?

Each node contains approximately log(N) successors, where N is the number of nodes in the system. The ring gets disconnected only if a node loses its successors, which is a relatively small probability.

b) Imagine a Chord overlay network is formed by connecting a few nodes in UC Berkeley and a few nodes in MIT. Consider the two following approaches for assigning node IDs.
   i) The IDs of the UCB nodes have the first bit set as 1, where the IDs of the MIT nodes have the first bit set as 0.
   ii) Node IDs are generated by taking a hash of the IP address and port number at which the nodes run.

   Name the advantages and disadvantages of these two approaches?

   i) Advantage: Faster lookups, as nodes in the same geographic region can find other nodes in the same geographic region contacting only close by nodes. Disadvantage: Less resilience, as the failures of nodes in the same geographic region of nodes at MIT will very likely disconnect the network.
   ii) The other way around.

c) In the basic Chord protocol, node n maintains fingers $f_i = \text{succ}(n + 2^{i-1})$, for $i = 1, 2, \ldots, m$, where m represents the number of bits of an ID. Now assume a variant of Chord where node n keeps a fingers $f_i = \text{succ}(n+4^{i-1})$, for $i = 1, 2, \ldots, m/2$. How does the number of hops of a lookup in the modified Chord protocol compare to the number of hops of a lookup in the basic Chord? Explain your answer.

Let d be the distance between the source and the destination node. In the case of basic Chord, every hop you reduces the distance to the destination by at least half (1/2) so it will take at most $\log_2 d$. In contrast, in the modified Chord, each hop reduces the distance to the destination by at least one quarter (1/4), so in the worst case there are still three quarters (3/4) of the distance to the destination. Thus, in this case it will take at most $\log_{(4/3)} d$ to reach the destination.

Security – Programming Bugs

Consider the following simple C code:

```
#include <stdio.h>

void manipulate(char *buffer) {
  char newbuffer[80];
  strcpy(newbuffer,buffer);
}

int main() {
  char ch, buffer[256];
```

```
  int i=0;

  while ((ch = getchar()) != '\n') {
    buffer[i] = ch;
    i++;
  }

  manipulate(buffer);

  printf("The value of i is : %d\n",i);
  return 0;
}
```

(a) Identify two bugs in the code above and explain their implications.

i) strcpy() in manipulate() needs to check for length of the buffer before copying in newbuffer.
ii) The while() in main() should make sure that i doesn't exceed buffer length.

(b) Explain how you can fix the two bugs.

i) strcpy() should make sure that length of buffer doesn't exceed the size of newbuffer, i.e., 80.
ii) Replace while() in main with

```
while (((ch = getchar()) != '\n') && i < 256)
```

Network Performance Metrics

Complete each statement with a short "fill in the blank" phrase (1 point per fill-in):

a. File transfer, Email, and web access require two properties from the network:

1. **In order packet delivery**
2. **Reliable delivery**

b. Consider the network performance metrics Delay, Bandwidth, and Loss. For the following applications, which of the three is the most critical metric and why?

1. Distribution of web pages to remote web caches: **Loss**

Rationale: **Low loss insures rapid dissemination of small files through the network; otherwise need to retransmit to many packets which takes a long time.**

2. Two-way "Voice over IP" packet telephony: **Delay**

Rationale: **Interactive responsiveness is needed, audio streams are loss tolerant;**

3. Remote copying of large databases for disaster management: **Bandwidth**

Rationale: **Must move large bulks of data as quickly as possible; time to last byte is essential;**

Miscellaneous 1

a) *Worm propagation.* Each instance of the code-pink worm infects 1 machine in one second. If we start from a single infected machine, how many new machines get infected from second 10 to second 11? How many were infected by second 11?

From second 0 to 1, 1 machine gets infected. From second 1 to 2, 2 new machines get infected. From second N to N+1, 2^N new machines get infected. From second 10 to 11 we have 2^10 which is 1024 new machines. The total number of machines infected by second 11 is 2048.

Note: We also gave full credit for solutions which assumed that time starts at 1, instead of 0.

b) *Public key cryptography*. Assume you know the public key of entity X, but X has no information about

you. Can you design a simple protocol to communicate *confidentially* with X in both directions (i.e., no one else knows what you and X are sending to each other)? If yes, specify the protocol, otherwise argue why this is not possible.

Yes. You send messages to X encrypted with X's public key and only X can decrypt them. Also, you send X in the first message your public key or a secret key with each X can encrypt the return communication. Note that this does not imply you are authenticated to X.

Miscellaneous 2

For each of the following statements, indicate whether the statement is True or False, and provide a very short explanation of your selection.

a. Time division multiplexing allows a connection to use unused slots of another connection. T F
Rationale:

False. Unused slots cannot be used by other connection.

b. In datagram switching networks two packets of the same flow always take the same route. T F
Rationale:

False. Each packet is forwarded independently.

c. Sliding window achieves higher throughput than Stop-and-Go. T F
Rationale:

True. Sliding window allows more in-the-fly packets during the same RTT.

d. Flow control slows down the sender when the network is congested. T F
Rationale:

False. Flow control slows down the sender when the receiver is slow.
(Congestion control slows down the sender when the network is congested.)

e. Multiplexing a large number of flows reduces burstiness. T F
Rationale:

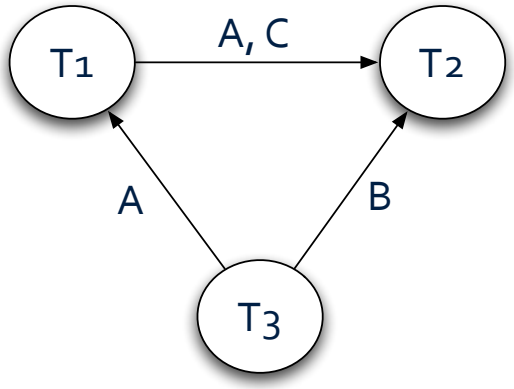
True. As number of flows increases, aggregate_peak/aggregate_avg decreases, as there less chance that the peak utilization of all flows coincide.

Concurrency Control

For parts (a-d), consider the following schedule of three transactions. Commit abbreviated "com"

Operation	1	2	3	4	5	6	7	8	9	10	11	12
T1:	R(C)		R(A)				W(A)			com		
T2:				W(C)				R(A)	W(B)		com	
T3:		R(A)			R(A)	W(B)						com

a) Draw the dependency graph for this schedule. Be sure to list the object(s) (A, B, or C) that is (are) the cause of each dependency on each edge.



b) Is this schedule conflict-serializable? If so, list a serial ordering of the transactions that would produce an equivalent schedule. If not, state why not.

Yes. T3 -> T1 -> T2.

c) This schedule of read and write operations could be generated by a system following the regular **2PL** (two phase locking) protocol. (Circle one)

TRUE **FALSE**

d) This schedule of read and write operations could be generated by a system following the **Strict 2PL** protocol. (Circle one)

TRUE **FALSE**

e) In general, is Strict 2PL is more likely to encounter deadlocks than regular 2PL? State **Why** or **Why Not**.

Yes. Locks are held longer in Strict 2PL, thus increasing the likelihood of deadlocks.