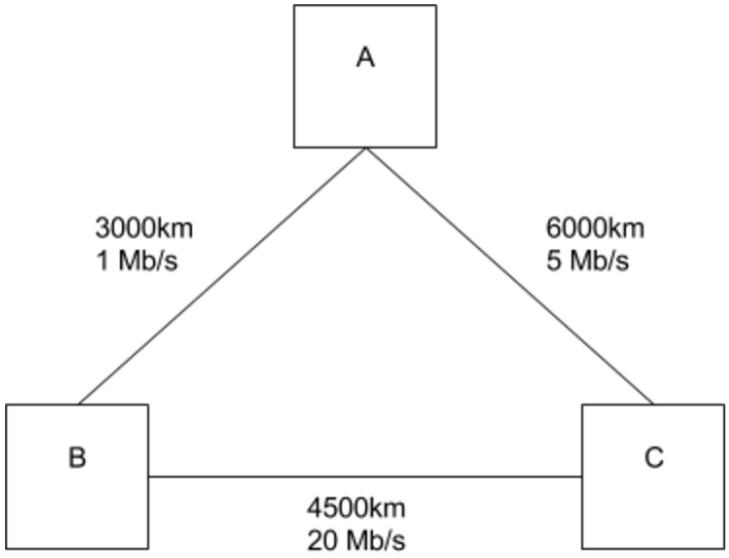
15-440/15-640: Homework 1

Due: September 21, 2017 10:30am

Name:
Andrew ID:

1 Network Communication

For this section, assume the speed of a signal in a wire is $2 * 10^8$m/s. Use the following diagram for questions 1, 2, and 3.



1. A, B, and C, are nodes attempting to accomplish some task in real time (e.g. playing a video game). The nodes elect one to be the 'host,' and all other nodes connect to the host to send data of an insignificant size. Which of the three nodes should be the host, and explain what factors you're considering. Show all work.

   The key insight to have here is that latency is all that matters, since the packet size is insignificant. Taking the speed of a signal to be $2x10^5 km/s$, we see that $A - B$ has 15ms of latency since time = distance divided by speed, and $\frac{3000km}{2x10^5 km/s} = 15ms$. By the same token, $A - C$ has 30ms, and $B - C$ has 22.5ms. Therefore B should be the host, as the average latency then is 18.75ms.

2. Now let's say A, B, and C are nodes involved in some very computationally expensive distributed computing work. They each computed some partial computation, but now need to combine their results in one node. Each node must send all their work to a single node, and you need all the data on hand in order to begin processing it all. A has 1 gigabyte of information, B has 5 gigabytes, and C has 3 gigabytes. Assume the network has no faults, which node would you pick to be the server in order to minimize the total time of combining all this information? Show all work.

The key insight here is that latency is completely overshadowed by the bandwidth limitations, given that the file sizes are so large. We want to figure out the difference in ratio of the speed of network transfers. We want to minimize = datasize / bandwidth + latency.

Given datasize D in Gigabytes, bandwidth M in Megabit/s, and length between nodes L in km, we want to find the equation: $\frac{D*8*1024*1024bits}{M*10^6 bits/sec} + \frac{L}{2x10^5 km/s}$ for each pair of nodes X, Y, and then take the minimum of the average for each node. Here we just plug the numbers into the equation.

(a) $B \to A$: $D = 5, M = 1, L = 3 \to$ 41.943s + 15ms = 41.958s

(b) $C \to A$: $D = 3, M = 5, L = 6 \to$ 5.033s + 30ms = 5.063s

(c) $A \to B$: $D = 1, M = 1, L = 3 \to$ 8.388s + 15ms = 8.403s

(d) $C \to B$: $D = 3, M = 20, L = 4.5 \to$ 1.258s + 22.5ms = 1.280s

(e) $A \to C$: $D = 1, M = 5, L = 6 \to$ 1.677s + 30ms = 1.707s

(f) $B \to C$: $D = 5, M = 20, L = 4.5 \to$ 2.097s + 22.5ms = 2.119s

$min(max(B \to A, C \to A), max(A \to B, C \to B), max(A \to C, B \to C))$
$\to min(max(41.9, 5.0), max(8.4, 1.2), max(1.7, 2.1))$
$\to min(41.9, 8.4, 2.1)$
$\to C$ is the fastest.

3. Assume you can pick which node generates the 1, 3, and 5 gigabytes of information, and you can pick which node becomes the server. What combination will you pick and why? Show all work.

Again, distance between servers doesn't really matter here. Since C has the fastest connections, it makes sense to have C be the server. It then follows that C should contain the 5GB ahead of time, making the latency for receiving that information 0. Finally, B should have the 3GB since $B \to C$ has a better bandwidth than $A \to C$. This leads to the ratio: $max(3/20, 1/5) = 0.2$. This solution is then roughly 20% faster than in part B.

4. (For the following question, ignore the previous diagram.) Consider two nodes A and B in a network, where the bandwidth of network path from A to B is 5KB/s and propagation time is 120ms. On the reverse path i.e. network path from B to A, the bandwidth is 10Kb/s and propagation time is 80ms. Data packets are of size 500 bytes and acknowledgment packets are of size 100 bytes. Based on this information answer the following questions. Please note that each of the following questions are independent of each other.

   (a) What is the throughput that can be achieved when A is transmitting to B using Stop-and-Wait protocol?

   The transmission time of a data packet from A to B is (500 B)/(5 KB/s) = 100msec. The transmission time of an acknowledgment from B to A is (100 B)/(10 KB/s) = 10 msec. The total propagation time is 80+120 = 200 msec. Thus an RTT for sending a data packet and receiving an acknowledgment for it is 200+100+10 = 310 msec. With Stop-and-Wait, A sends one data packet per RTT, so the throughput it can achieve is (500 B)/(310 msec) = 500/0.310 B/s = 1613 B/s.

   (b) Suppose now A is using a sliding window for transmitting packets. Determine the size of the sliding window in terms of packets that A should use to transfer data as fast as possible.

   To go as fast as possible, A must use a window that is at least as large as the bandwidth-delay product, which is (5 KB/s)(310 msec) = 1,550 bytes. The question asks for how many data packets, which is given by: Ceiling[1550 B / (500 B/pkt)] = 4 pkts

2 Consistency + Classic Synchronization

1. For reference:

   `https://en.wikipedia.org/wiki/Dining_philosophers_problem`

   The dining philosophers have found a potential solution to avoid deadlock. Before eating, each philosopher would flip a coin to decide whether he should pick up the left fork or the right fork first. If the second fork is taken, the philosopher would put the first fork down, and flip the coin again. Do you think that this solution is deadlock-free? Does it guarantee that the philosopher's won't starve? Explain your answers.

   This solution is deadlock-free, however it does not guarantee freedom from starvation. The philosophers are never stuck waiting and unable to do "useful work." However, a philosopher might not ever get a chance to eat because at least one of his/her forks is always busy.

2. What are the differences between Mutexes and Semaphores? Explain scenarios where you would prefer using one over the other. One scenario for each situation should suffice.

   Mutexes: used to protect the shared resources from concurrent access. For example, Readers Writers problem.

   Semaphores: used to indicate occurrence of an event to other processes. For example, producer consumer problem.

3 Remote Procedure Calls

1. What would be some of the benefits of having RPC calls where both the server and client stubs are running on the same hardware?

   Modularity - you can move either the server or client to different machines but this will not result in any significant changes to the code

   Security - Because of the segmented address spaces, corrupt code or input could not do as much damage

2. There are some drawbacks to using RPC on the same hardware - describe one scenario where it would not be preferable.

   In low-latency, high-throughput scenarios, since there is some overhead with marshalling and unmarshalling data

3. In an RPC system, programmers may need to pass complex objects or structures to the remote procedures. These complex objects may contain pointer based structures such as reference to a tree node or a linked list. Discuss the implications and approaches of passing this data by value and by reference.

   Passing by value is simple: just copy the value into the network message. Passing by reference is hard. It makes no sense to pass an address to a remote machine since that memory location likely points to something completely different on the remote system. If you want to support passing by reference, you will have to send a copy of the arguments over, place them in memory on the remote system, pass a pointer to them to the server function, and then send the object back to the client, copying it over the reference.

4 Distributed File Systems

1. In a few sentences or less, describe the CAP theorem and what its implications are for distributed file system design.

   The CAP theorem states that no distributed system can have more than two of three of consistency, availability, and partition tolerance. This means that every distributed data store must make trade-offs between the three.

2. In this spectrum of three qualities with trade-offs, where does AFS lie?

   If I open a file to write, and make some changes, and before I commit my changes, someone else opens the same file, they will not see my changes. So AFS lacks Consistency. Two users cannot simultaneously edit the same file (vs. Dropbox or Google Docs for instance).

3. If you were to design a DFS for a global bank with many branches across the world to store critical financial records, which DFS qualities would you try to optimize for and why?

   Answers can vary based on which aspect of a bank the students target. If the issue at hand is a client checking their balance, for instance, then ideally the user would always see the most recent balance no matter what, meaning consistency would be very key. If the issue is a global bank with many distributed servers and global clients, then partition tolerance and availability would be very important.

5 Time & Synchronization

1. Fig. 4 Processes running events numbered a, b, c... to send and receive messages.
   There are 4 processes running in a distributed system, as shown in Fig 1. Labels a, b, c... indicate the events running on these processes while receiving and sending images. Using this diagram as reference answer the following questions:

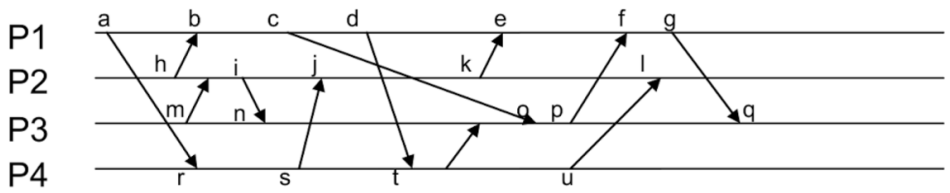
   (a) A clock at the client side reads 8:20:00. The server's clock reads 8:10:00. What would be the time at the client, if they use the following techniques for synchronization:

   Christian's Algorithm.
   Berkeley's Algorithm.

   Assume message delays are negligible. Show all your work.

   Christian's algorithm : In this algorithm it is assumed that server has an accurate clock. The clock obtains the time from server, and adds half of round trip delay to it. In this case the delay is 0, so the client sets his clock to 8:10:00.

   Berkeley's algorithm: In this case the client sets the clock to average of both the times, which is 8:15:00 in this case.



(b) For each labelled event list the Lamport timestamps associated with each of them. Assume that the initial logical clock is set to 0 in each process, and the timestamp is maintained as a single integer value. Repeat part 1 with Vector clock timestamps.

Event	Lamport TS	Vector TS

Table:

Event	Lamport TS	Vector TS
a	1	(1,0,0,0)
b	2	(2,1,0,0)
c	3	(3,1,0,0)
d	4	(4,1,0,0)
e	6	(5,5,1,2)
f	10	(6,5,5,4)
g	11	(7,5,5,4)
h	1	(0,1,0,0)
i	3	(0,3,1,0)
j	4	(1,4,1,2)
k	5	(1,5,1,2)
l	8	(4,6,1,5)
m	1	(0,0,1,0)
n	4	(0,3,2,0)
o	8	(4,3,4,4)
p	9	(4,3,5,4)
q	12	(7,5,6,4)
r	2	(1,0,0,1)
s	3	(1,0,0,2)
t	5	(4,1,0,3)
u	7	(4,1,0,5)