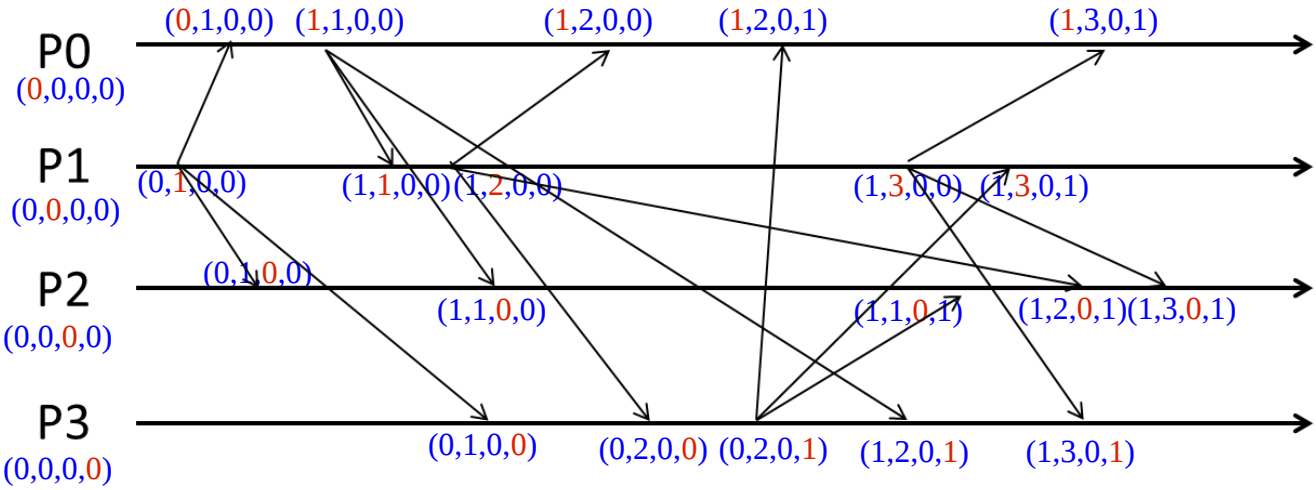
Name: Kevin Strasberg NetID: strasbe1 HW2

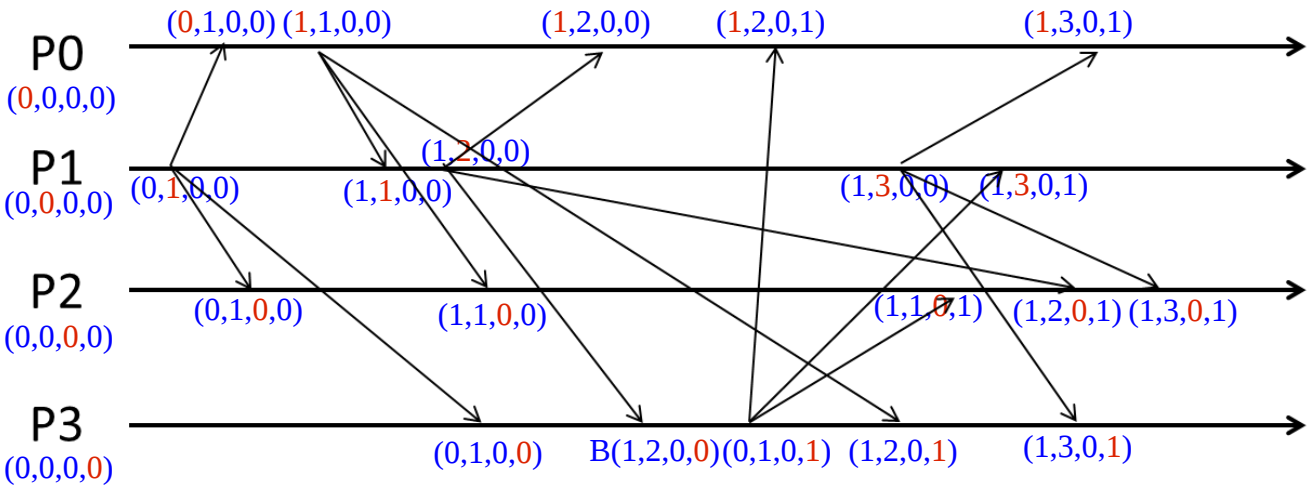
Problem 1:



Name: Kevin Strasberg NetID: strasbe1 HW2

Problem 2:

B = Buffer received message



Name: Kevin Strasberg NetID: strasbe1 HW2

Problem 3:

a. At most once
b. At least once
c. At least once
d. Any number of times because the server will not receive any requests
e. At least once

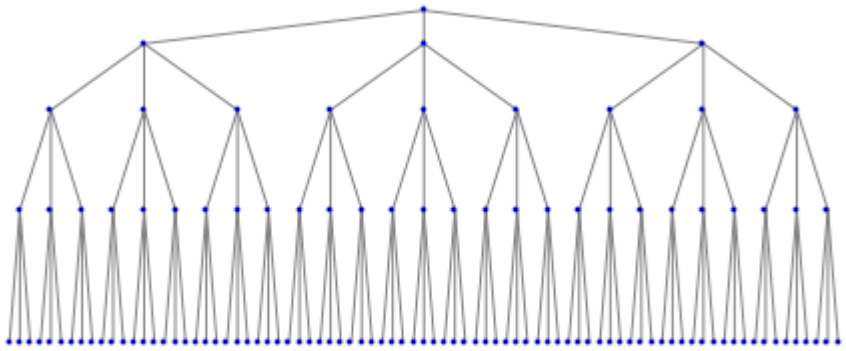
Name: Kevin Strasberg NetID: strasbe1 HW2

Problem 4:

a. 7 nodes since the highest level it can reach is 1 and the lowest level besides itself is it's parent's sibling

b. 48 nodes Since it contact all of its children and subchildren as well as the first layer of its siblings children

c. TTL = 8 since the message needs to be able to go from leaf->parent->parent->parent->root->differentChild->child->child->leaf



(4 levels)

Name: Kevin Strasberg NetID: strasbe1 HW2

Problem 5:

a. 45N Finger Table

i	45+2^i (mod 2^8)	ft[i]
0	45	132
1	46	132
2	47	132
3	53	132
4	61	132
5	77	132
6	109	132
7	173	234

b. Key 12 would be stored at peer N32 since 32 is the smallest id that is closes to 12. Then N45 would send query to the largest successor/finger entry <= k. Since none exists, it sends a query to N45's successor which is N32 which contains the key being looked for.

c. Only Node 199's finger table needs to be updated because that is the only node that can have any pointers to node 45.

Name: Kevin Strasberg NetID: strasbe1 HW2

Problem 6:

The bully algorithm needs a synchronous system because it relies on timeouts to detect process failures. The system also relies on synchronization to keep track of coordinators. For example, if there are two processes, a and b, and a sends out a coordinator message then b sends out a coordinator message. Since there is no synchronization none of the other nodes will know whether a or b is really supposed to be the coordinator at a given time. So if a takes longer to send than b, the receiving node will think that a is the new coordinator when really b is the new coordinator.

Name: Kevin Strasberg NetID: strasbe1 HW2

Problem 8:

a. A simple ring algorithm would use a token to control the write privileges. So if the process with the token does not want or need to write to a file, it will just send it to its adjacent neighbor who can choose to use it or not. Then, for 4 separate tokens that represent reading privileges that will be passed around in the same manor, if a token already has a reading token and is using it, it will forward its reading token on to its adjacent neighbor. In order to prevent more than one read with a simultaneous write, the process holding a writing token will need to broadcast a “prepare to write' message to all the other processes. This will happen by sending the message to its adjacent neighbor which will make the process stop reading before entering the next critical section and pause the circulation of the of the reading rings. The process will then forward the “prepare to write” message to the next neighbor that will do the same thing. The process wanting to write will not write until it has received the “prepare to write” message from its predecessor signifying all nodes know that the file is being written to. To allow one other process to simultaneously read while the file is being written to, there will be one higher privileged reading token, that does not need to block all next file reads from the “prepare to write” message, but will be passed around the ring in the same manor as the other 4 tokens. When the process that is finished writing is done, it will need to send a “finished writing” message to signal all the other processes to continue. Again, the writing process must wait for the “finished writing” message to pass the token along to the next process.

b. This guarantees safety since there are set numbers of tokens to represent the different scenerios required. The write token is safe since it does not write to the file until all other processes know the write is occurring and stop reading from the file. When the file is not being written to there are only 5 tokens that allow processes to read from a file meaning that only 5 will ever be able to read from the file at the same time.

c. The bandwidth for this algorithm is 1 for if processes are just reading files. If there is a write occurring, there are 2N messages since every node must forward a “prepare to write” and “finished writing” message to their neighbors. This means also that the client delay is 0 to 2N and the synchronization delay is between 1 and 2N messages

Name: Kevin Strasberg NetID: strasbe1 HW2

Problem 9:

If any subset of the system is in a deadlock, then the entire system will be in a deadlock since the system will not be able to make progress. With that, consider three voters with subsets of V1 = {1,2}, V2 = {2, 3}, and V3 = {3,1}. If the processes request access for the lock at the same time, then 1 will reply to itself which will causing 1 to not reply to the other requests since it has already voted. Then 2 will do the same and not reply to 1's request since 2 voted for itself. The same will have happened for three since the requests were done at the same time. This will make it so all three processes are waiting for replies from everyone in the system, but will never receive them from 1, 2, or 3 since they replied to themselves and the system will be dead-locked.