Homework 5

CS425/ECE428 Spring 2026

Due: Friday, May 1, at 11:59 p.m.

1. Two-phase commit (2PC) . . . . . 3 points

The figure below shows a system of three servers (Server 1, Server 2, and Server 3) processing a distributed transaction. The transaction executes operations on (non-replicated) objects that are distributed across these three servers. The system uses the two-phase commit protocol to ensure atomicity for distributed transactions. Server 1 also acts as the coordinator for the transaction, and interacts with the client that issues transaction operations. The one-way network delay between the client and the coordinator, and among the three servers is indicated in the figure.

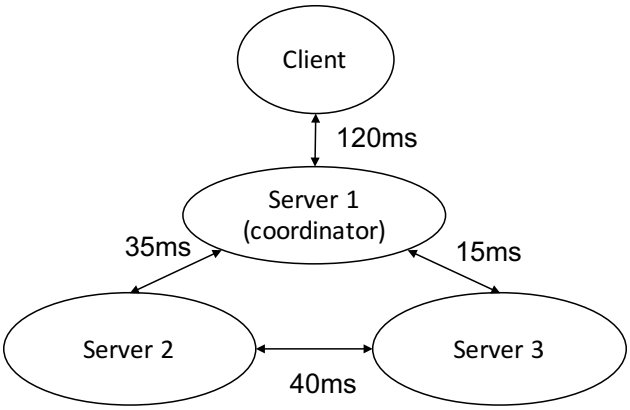

Figure 1: Figure for question 1

Any local processing at a server or self-messages take negligible time. The client issues a COMMIT request for its transaction at time 't'. Assume no messages are lost, no server crashes, and no server wishes to abort the transaction.

Answer the following questions in terms of time duration elapsed (in ms) since time 't':

(a) (1 point) When will Server 1 commit the transaction?

(b) (1 point) When will Server 2 commit the transaction?

(c) (1 point) When will Server 3 commit the transaction?

2. Two-phase commit (2PC) and Paxos ..... 8 points
Now consider a modification of the above question, where each individual server is replaced by a set of three servers that act as replicated state machines, as shown in the figure below.

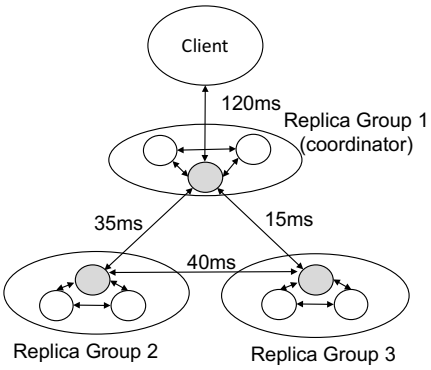

Figure 2: Figure for question 2

The three servers within each replica group use Paxos to achieve consensus within the group. The shaded server within each replica group acts as the leader (i.e. it acts as both the proposer and the distinguished learner for Paxos), while the remaining two servers act as acceptors. During two-phase commit (2PC), the leader of the coordinator group communicates with the leaders of the participant groups (one-way network delays for communication among the group leaders are shown in the figure). The client communicates with the leader of the coordinator group (one-way network delay for this is also shown in the figure). The one-way network delay for communication between the servers within each replica group is **5ms** (not shown in the figure).

During the execution of 2PC in such a system, there are three points at which a consensus must be achieved within the nodes in a replica group for a transaction to be committed: (i) at each participant group to prepare for a commit, (ii) at the coordinator to decide on a commit after receiving a vote from each participant, and (iii) at each participant again to log the final commit. (Note that Replica Group 1 performs the special tasks of the coordinator as well as acts as a participant replica group, storing objects for which the final commit needs to be logged. Assume that logging this final commit at Replica Group 1 needs a separate round of consensus as per (iii), and is not clubbed with the consensus executed by the coordinator for deciding on a commit as per (ii)).

Any local processing at a server or self-messages take negligible time. The client issues a COMMIT request for its transaction at time 't'. Assume that there are no failures or lost messages. Further assume that the leader of each replica group has already been elected / pre-configured. No node in any participant group wishes to abort the transaction.

Answer the following questions in terms of time duration elapsed (in ms) since time 't':

(a) (2 points) When will the leader in Replica Group 1 commit the transaction? When will the other servers in Replica Group 1 commit the transaction?

(b) (2 points) When will the leader in Replica Group 2 commit the transaction? When will the other servers in Replica Group 2 commit the transaction?

(c) (2 points) When will the leader in Replica Group 3 commit the transaction? When will the other servers in Replica Group 3 commit the transaction?

(d) (2 points) What is the earliest time at which the leader of Replica Group 1 (coordinator) can safely send a message to the client stating that the transaction will be successfully committed?

3. DHT ..... 15 points

Consider a Chord DHT with a 16-bit address space and the following 100 nodes (hexadecimal values in parentheses).

```
    167  (a7),     753  (2f1),    1622  (656),   5002  (138a),
   5291  (14ab),   5877  (16f5),   5944  (1738),   6426  (191a),
   6727  (1a47),   7258  (1c5a),   7632  (1dd0),   7807  (1e7f),
   9234  (2412),  13778  (35d2),  14634  (392a),  15725  (3d6d),
  16484  (4064),  18825  (4989),  18849  (49a1),  19145  (4ac9),
  20251  (4f1b),  20594  (5072),  20938  (51ca),  22309  (5725),
  22785  (5901),  23373  (5b4d),  23826  (5d12),  24173  (5e6d),
  24665  (6059),  26033  (65b1),  26094  (65ee),  26135  (6617),
  27566  (6bae),  27671  (6c17),  27882  (6cea),  28908  (70ec),
  29469  (731d),  29980  (751c),  30217  (7609),  32132  (7d84),
  32172  (7dac),  32181  (7db5),  32667  (7f9b),  32933  (80a5),
  33185  (81a1),  33543  (8307),  33659  (837b),  34384  (8650),
  35263  (89bf),  35902  (8c3e),  36337  (8df1),  36475  (8e7b),
  36534  (8eb6),  37524  (9294),  37843  (93d3),  38386  (95f2),
  38439  (9627),  39333  (99a5),  39390  (99de),  40253  (9d3d),
  40730  (9f1a),  41794  (a342),  42258  (a512),  42415  (a5af),
  44525  (aded),  44904  (af68),  46136  (b438),  46588  (b5fc),
  46951  (b767),  47390  (b91e),  47877  (bb05),  48888  (bef8),
  49162  (c00a),  49220  (c044),  50340  (c4a4),  50665  (c5e9),
  50894  (c6ce),  51340  (c88c),  52596  (cd74),  52766  (ce1e),
  53397  (d095),  53528  (d118),  55208  (d7a8),  55562  (d90a),
  56361  (dc29),  56804  (dde4),  56835  (de03),  57373  (e01d),
  57441  (e061),  57942  (e256),  58391  (e417),  58393  (e419),
  58643  (e513),  60081  (eab1),  60904  (ede8),  62348  (f38c),
  62453  (f3f5),  64220  (fadc),  65465  (ffb9),  65514  (ffea),
```

For programmatic computations, these numbers have also been made available at:
`https://courses.grainger.illinois.edu/ece428/sp2026/assets/hw/hw5-ids.txt`

(a) (5 points) List all 16 finger table entries of node 27671.

(b) (10 points) List the nodes that would be encountered on the lookup of the following keys by node 27671:

(i) 37227

(ii) 17084

4. MapReduce ..... 6 points

(a) (6 points) Given two 2D matrices M_a (with dimensions 5×10000) and M_b (with dimensions 10000×20). Use a map-reduce chain to multiply them, resulting in a matrix M_c (with dimensions 5×20). The input to the map-reduce chain is in the following key-value format: (k, v), with $k = (i, j, x)$, where $x \in a, b$ denotes the matrix identity, i and j indicate the matrix indices, and v is the corresponding value $M_x[i, j]$. The output of your map-reduce chain must of the form (k, v) with $k = (i, j)$ indicating the indices of the resulting matrix M_c, and v being the corresponding value at those indices (i.e. $M_c[i, j]$).

Assume there are 1000 nodes (or servers) in your cluster. Your map-reduce chain must support proper load-balancing across these nodes. Ensure that a single node is not required to handle more than ≈2000 values at any stage. You can assume that, if allowed by your map-reduce semantics, the underlying framework perfectly load-balances how tasks are assigned to nodes, and how keys are assigned to tasks.

Also mention, for each stage in your map-reduce chain, how many tasks each node runs, and at most how many values each such task handles.

5. Dominant Resource Fair Scheduling ..... 8 points

(a) (8 points) Consider two cloud jobs, Job 1 and Job 2. Each task of Job 1 requires 20 units of CPU and 20MB RAM. Each task of Job 2 requires 5 units of CPU and 50MB RAM. You need to schedule these jobs on a system with 200 units of CPU and 1GB (i.e. 1000MB) RAM using the dominant resource fairness criteria. How many tasks for each job will you schedule to maximize resource utilization while achieving dominant resource fairness (to the best extent possible)?

(Note that you cannot schedule fractional tasks. The unfairness introduced due to rounding the number of tasks to integer values can be ignored. Moreover, unfairness can also arise when scheduling more tasks for one type of job is not possible due to resource capacity constraints, but another job can still be scheduled.)

There are two possible solutions for the above problem. You need to find both solutions, and mention how much total cluster resources (i.e. how many units of CPU and how much RAM) are utilized by each solution.