Distributed Systems

End Sem Exam 2023

PART A

Instructions:

1. For each of the questions; tick ALL the correct options. Note: Multiple answers may be correct. For certain questions, none of the options may be correct.
2. All questions are to be attempted.
3. If the question is not clear; clearly state your assumption in the space beside/after the question and attempt the question.
4. Do not provide explanations for the options you tick mark.
5. If ALL the options provided are incorrect then mention so beside the question. If nothing is mentioned nor ticked, then, the question will be treated as un-attempted.
5. Rough Work is not be done on the left margin

Marking Scheme:

For every correct answer marked correctly: 5 marks
For every correct answer that is not marked: -2 marks
For every wrong answer marked as correct: -3 marks

1. Topic: Traversal Algorithms
   a. Awerbuch is faster than classical DFS because of the parallelisation of the information passed.
   b. Awerbuck will run faster than classical DFS if the topology of network is a tree when both are run on the same machine
   c. Cidon's will perform similar in terms of time complexity to classical DFS if the topology of network is a tree.
   d. If the topology of a network is a clique then the Cidon algorithm will work exponentially better than the classical DFS in terms of running time on the same machine.

   Ans: a,c,d

2. Topic: GFS
   a. GFS stores data as files and metadata as key-value pairs in a distributed file system.
   b. GFS employs a centralized master node to coordinate file operations, manage metadata, and handle failures.
   c. GFS uses a write-once-read-many (WORM) model for file updates, where files can be appended but not overwritten.
   d. GFS allows clients to modify file blocks in parallel and uses a lock-based protocol to ensure consistency and mutual exclusion.
   e. GFS uses a distributed hash table (DHT) to map file blocks to chunkservers, and each chunkserver stores one or more replicas of each block.

   Ans: a,b,e

3. Topic: Mutual Exclusion
   a. Suzuki-Kasami is a non-token based algorithm.

b. Ricart-Agarwala's algorithm needs 2N − 2 messages per CS execution.
c. In a distributed system, mutual exclusion always requires the use of a global clock to coordinate access to shared resources.
d. In the Lamport's algorithm, a process can enter the critical section only when it receives a message from all other processes in the system.
e. System Throughput is the rate at which the system executes requests for the CS.

Ans: b,e

4 Topic: Agreement Protocols

a. If we know the solution to Byzantime Agreement problem, then, we can create a solution such that all non-faulty processes agree on a common vector (v1,v2,…,vn) where if the kth process is non-faulty and its initial value is vk, then the kth value in the vector agreed upon by non-faulty processes must be vk. If the kth process is faulty then the kth value agreed upon by the non-faulty process can be any value.
b. For Lamport-Shostak-Pease Algorithm for B(N,T) of N processes and atmost T faulty processes; if there is only one faulty process and that process is not the general, then, the vector of values for all the other non faulty processes at B(N-1, T-1) would be exactly the same.
c. In the Consensus Algorithm for crash failure; suppose a process that crashed in between but then revived had the minimum initial value k among all processes. Then the value agreed upon by all processes is always k.
d. The Consensus Algorithm for crash failures; gives correct results always.
e. No nodes will crash in at least one round in the Consensus Algorithm for crash failures if the number of crashes is less than the parameter f.

Ans: a, e

5. Topic: MapReduce

Lets write a mapper function for a MapReduce which is given a string as input:

```
def mapper(text_string):
        key =  text_string[0]
        value = 1
        return (key,value)
```

Consider the following input to MapReduce:

Apple
Banana
Baseball
Badminton
Brother
Austria
Apricot
Poverty
Piano
Tiger
Timber
Telephone

Assuming there are 4 reducers, and shuffle & sort phase is applied to the above input, how many key-value pairs can each reducer possibly receive?

a. 2, 4, 5, 1
b. 3, 3, 3, 3
c. 4, 2, 3, 3
d. 7, 0, 2, 3

Ans: c,d

6. Topic: Deadlocks

a. In the Mitchell-Merritt algorithm, if a cycle of N nodes forms and persists long enough, the lowest priority process in the cycle will execute the Detect step after at most N*(N − 1)/2 consecutive Transmit steps.
b. In the Chandy Misra Haas Edge Chasing algorithms, the space complexity is O(N).
c. The four necessary conditions for a deadlock situation to occur are mutual exclusion, hold and wait, pre-emption and circular wait.
d. For the single-resource model, the out-degree of a node in a WFG should be 1.

Ans: all false

7. Topic: Agreement Protocol

a. In the two phase commit protocol, during coordination failure, because of the blocking problem, all active sites have to wait for the coordination site to recover irrespective of their log file content.
b. There is no difference in handling of site failures by the 2 phase commit protocol and the 3 phase commit protocol.
c. In two and three phase commit protocol; deadlocks will not occur.
d. The real use of the Two-phase commit protocol is atomicity (all-or-nothing commits at all sites)

Ans: b,d

8. Topic: GHS Algorithm for minimal spanning tree

a. If all nodes are woken up simultaneously, and each node starts running the GHS algorithm sending out connect messages over its minimum outgoing edge, then, the overall result will be incorrect
b. The GHS algorithm works for all directed graphs
c. The GHS algorithm provided by the authors and taught in class works for all undirected graphs
d. The largest weight edge in the MST will be the final edge in the GHS algorithm to join two fragments.

Ans: all false

PART B

Do any TWO [2X15 marks]

Question 1

A. Define 3 properties needed of algorithms for mutual exclusion in the distributed system (1)
Ans: ME, Progress, Bounded waiting,

B. Mutual exclusion in a distributed system can also be achieved by having every process waiting to enter the critical section send a msg to a process holding the resource(the host). The host will collect the requests and arbitrate entry to the critical section. Would the above mechanism work with respect to the properties you mentioned above? What are some situations where the above mechanism wont work. Explain (5)
Ans: Cant ensure bounded waiting, also no progress also possible as everyone might be waiting as the process to which permission is given may not have other resources required to proceed.

C. Will consistent hashing work if my business needs more than 360 servers ? Explain. (2)
Ans: yes. The servers can be distributed on a ring of any size. We can take a circle of any size much greater than the number of servers. Circle need not be 360

D. (i)In the Google File System; why do we need lease times for primary chunkservers? (1)
Ans: (i) if a primary can not be reached or fails the master can wait till expiry time and make someone else as primary. If lease was not there and the master did this then there could have been two primary servers functioning.
(ii) primary also understands that if expiry time when it recovered then master would have found another primary and it does not have to continue .
(ii) Are all replicas of a node always consistent? Explain what forms of issues can be seen. (6)
Ans:

No all are not consistent. Four cases were discussed in class
(i) primary wrote but secondaries couldn't
(ii) primary wrote and secondary wrote but other secondary did not
(iii) Concurrent writes – each gets a start index concurrently but the streaming data though written serially from concurrent writes may overwrite data making it consistent but undefined
(iv) if might have duplicate entry of same thing in two places for primary but only one entry in secondary making the total picture of those different. This happens when on retrying also it failed and so the whole process is repeated . so primary still has an old copy for which index the secondary hasn't written.
(check few papers. I have given an example in class of possible of issues)

Question 2: Consider below Tarrys algorithm for undirected graph traversal.

A. What is the running time of Tarry's algorithm? (2)
O(V+E) . Show
B. Prove each vertex v is visited at most deg(v) times, except the starting vertex s, which is visited at most deg(s) + 1 times. (3)
Some explanation needed
C. Does the algorithm always terminate on start node s ? (3) Yes. prove
D. Would Tarry's algorithm be able to visit every vertex of G? (3) Yes. prove
E. How would you compare this algorithm with the best traversal algorithm you have in mind? (4) Comparison of running and msg time with Cidons can be one way

```
TARRY(G):
    unmark all vertices of G
    color all edges of G white
    s ← any vertex in G
    RECTARRY(s)
```

```
RECTARRY(v):
    mark v                                  ⟨⟨"visit v"⟩⟩
    if there is a white arc v→w
        if w is unmarked
            color w→v green
        color v→w red                       }
        RECTARRY(w)                         } ⟨⟨"traverse v→w"⟩⟩
    else if there is a green arc v→w
        color v→w red                       }
        RECTARRY(w)                         } ⟨⟨"traverse v→w"⟩⟩
```

Question 3

A. [2X4=8 marks] Consider a modified two-phase commit protocol (M2PC), where participants do not write transactions to their logs until after "Commit" is received. In all other ways the modified protocol is identical to standard 2PC. In all parts of this question, consider the M2PC protocol. Assume we have a system where the only failures involve hosts halting with

their disks and logs intact, followed by reboots, with no network message loss (if a server sends a message and the target crashes before responding, the server will resend the message after a reboot). Suppose there is a coordinator C and two participants P1 and P2.

a. Assume the following sequence of events happens:
   C sends <Prepare T> to P1, P2
   P1 sends <Ready T> to C.
   P2 sends <Ready T> to C
   What message will be sent after this sequence in M2PC?
b. What happens if P1 crashes and recovers. The sequence of events looks like this
   C sends <Prepare T> to P1, P2
   P1 sends <Ready T> to C.
   P2 sends <Ready T> to C
   P1 crashes
   P1 recovers
   What message will be sent after this sequence in M2PC?
c. Assume that after part b, the remainder of the M2PC protocol executes for transaction T without further incident. Does T commit successfully on P1? On P2? Why?
d. What would happen if standard 2PC protocol was followed, does T commit successfully on P1? On P2? Why?
   Ans: https://web.stanford.edu/class/cs245/exams/final_win20_sol.pdf

B a. Consider the Consensus Algorithm for Crash Failures. We assumed that if a process crashes, that process will not participate in further activities forever. However, let us modify the algorithm a bit such that a faulty process can crash once, then restart once again and participate in further activities and then can crash once again but it cannot revive after crashing for second time. The consensus will be reached for all the non-faulty processes in 2(f+1) + 1 rounds of the algorithm. (4)

Ans: False, let us say there are three processes A,B and C with values 1,2, and 3. Lets say process A is faulty and others are fine. We need to reach a consensus in 5 rounds(2*(1+1)+1). Process A fails initially only then process B and C proceed:
Round 1:
B and C both have 2 as their values.
Round 2:
Again B and C have 2 as their values.
Round 3:
Again B and C have 2 as their values.
Round 4:
Again B and C have 2 as their values.
Round 5:
Process A comes back to life sends process B value 1 and dies.
After round 5,
Process B has value 1, while process C has value 2, so consensus is not reached.

b. Suppose in the orginal algorithm there exists a round k where k < f+1, such that in round k no process sends any value to any other process, then, we can terminate the algorithm after round k.(3)
   Ans: True. Explanation to be given

   On this last question, some thought they need to prove it. Check what they have done. Also some had questions about what value will the process take when it recovers. They can assume whatever.