Chalmers Techniska Högskola
Data- och Informationsteknik

CTH TDA297
GU DIT290

Distribuerade system fk

Tentamen 2019-03-20

Dag, *Tid, Sal*: March 20th 2019, 08:30-12:30, SB-MU building

***Kursansvarig*:** Philippas Tsigas (Tel: 772 5409)

***Hjälpmedel*:** Inga

Totalt *Poängtal*: 62

***Betygsgränser*:**

CTH: 3:a 30 p, 4:a 38 p, 5:a 48 p

GU: Godkänd 30p, Väl godkänd 48 p

Instructions

- Please answer in English, if possible.
  If you have very big difficulty with that, though, you may answer in Swedish.
- **Do not forget to write your personal number and if you are a GU or CTH student and at which "linje".**
- Please start answering each assignment on a new page; number the pages and use only one side of each sheet of paper.
- Please write in a tidy manner and explain (briefly) your answers.
- Students must **not** write their personal number on the answer sheets since the exam is anonymous; they shall write that **only** on the name slip area that they will seal.

LYCKA TILL !!!!

1. (15 points) Moa is building a database that is replicated on N machines. To access the database, a client accesses any of the replicas. The communication between clients and the replicas and between the replicas themselves is reliable, point-to-point, and FIFO-ordered. However, the communication delays can vary significantly. Moa has designed the following variant of a totally-ordered multicast algorithm to be used on her system:

   > Each replica maintains Lamport's logical clock, and every inter-replica message is stamped with the unique id of the sender upon transmission. Whenever a replica receives a database update message from a client, it brodcasts the update in a message to all replicas (including itself). Whenever a replica receives an update message from another replica (or from itself) it puts the message into a local queue, and acknowledges the reception of the update by sending an acknowledgment message to all other replicas (including itself). The replica applies an update from its local queue to its local database if and only if:
   >
   > i) the update message has the lowest timestamp among the messages in the replica's local queue, and
   >
   > ii) the update message has been acknowledged by a quorum of N
   >
   > After a replica has applied the update to the local database, the update message and its acknowledgments are removed from the local queue. If later another acknowledgment arrives for the message, such a late acknowledgment is simply dropped from the queue.

   Prove that the above algorithm implements totally-ordered causal multicast (i.e. satisfies the following requirements: 1. Reliability: Integrity, Validity, Agreement 2. Ordering: Causal, Total-order) or produce a counter example.

2. (13 points) A connected bidirectional asynchronous network of n processes with identities has diameter D and may contain zero or more "evil" processes. Fortunately, the evil processes, if they exist, are not Byzantine, and will correctly execute any code we provide for them. Suppose that all processes wake up at time 0 and start whatever protocol we have given them. Suppose that each process initially knows whether it is evil, and knows the identities of all of its neighbors. However, the processes do not know the number of processes n or the diameter of the network D. Give a protocol that allows every process to correctly return the number of "evil" processes. Your protocol should only return a value once for each process (no converging to the correct answer after an initial wrong guess). Discuss and prove correctness and the me time complexity of your algorithm.

   Clarification: You get 10 points for a correct answer and 3 extra points if your algorithm terminates in $O(D)$ time.

3. (14 points) A distributed system is composed by n processes that have access to k resources ($k > 0$). Each process p_i tries to accesses exclusively h_i resources ($k \geq h_i > 0$) through a *REQUEST(h)* operation and, once it has acquired and accessed the resources, it can release them by executing a *RELEASE(h')* operation (the resources released may be a subset of those acquired). Describe in pseudo-code an algorithm implementing the *REQUEST(h)* and the *REPLY(h')* operations that guarantee no-starvation and that no two processes access the same resource at the same time. Processes communicate via point-to-point message passing communication. No failures are considered. Discuss and provide a correctness proof of your algorithm together with its time complexity.

   Clarification: You get 10 points for a correct answer and 4 extra points if the complexity of your algorithm is polynomial on the maximum number of conflicts per resource.

4. (10 points) Moa is building a "Grow-Only" Set that is replicated in the form of a sequential state machine on 2 servers, server S and server N. This set supports only add operations, that add an element on the set, and read operations, that return the state of the set without deleting any element, i.e. no deletes. To access the object, all client that are located in the south accesses the local copy at server S and all clients from the north access the local copy at server N. The two servers start from the same initial state. There is no communication and synchronization between the two servers until the end of the protocol when the two sets copies are meged. If there is no mobility, i.e. no client from the south can move to the north or vise versa, which is consistency that the scheme provides from the following three consistency models: 1)linearizability, 2) sequential consistency, 3) strong eventual consistency (CRDT "Grow-Only" Set)? Provide a proof sketch or counterexample. If there is mobility what is the consistency that this replication scheme provides?

5. (10 points) Each statement is either true or false. A correct answer gives 1 point, a wrong answer gives -1 point, no answer gives 0 points. Overall you cannot get less than 0 points for this question.

    1. There exists a symmetric leader election algorithm.
       A. True B. False
    2. There exist a symmetric solution for the resource allocation problem.
       A. True B. False
    3. A solution to the consensus problem can solve the mutual exclusion problem.
       A. True B. False
    4. Causal consistency implies Sequential Consistency.
       A. True B. False
    5. Linearizability implies Causal Consistency.
       A. True B. False
    6. Linearizability implies Sequential Consistency.
       A. True B. False
    7. In an asynchronous system, where only one processor might crash, consensus is solvable.
       A. True B. False
    8. The higher the quorum of an operation the higher the availability of this operation.
       A. True B. False
    9. The 3 Phase Commit protocol was introduced to improve the latency of the 2 Phase Commit protocol in executions where no faults take place.
       A. True B. False
    10. 3PC was introduced to handle undetected message losses.
       A. True B. False