EXERCISES ON PAXOS

DISTRIBUTED SYSTEMS

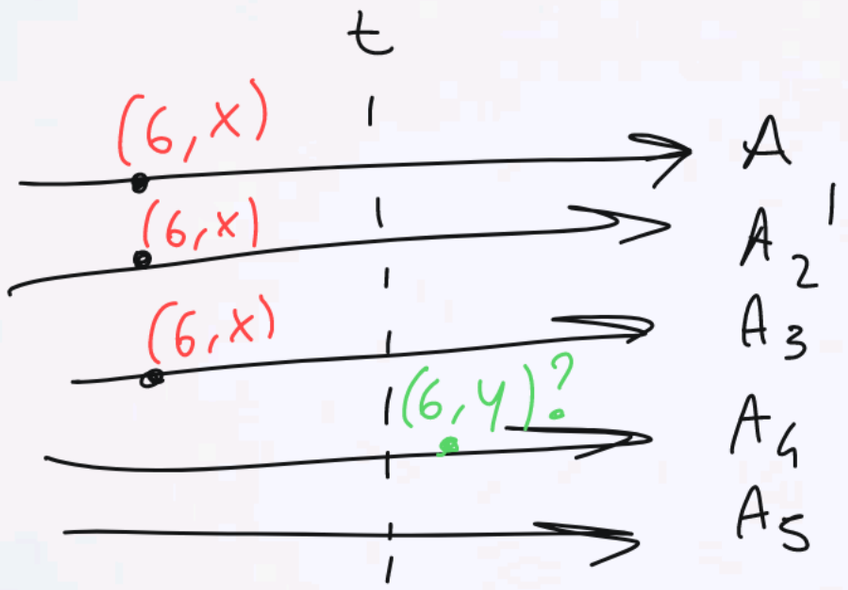
Master of Science in Cyber Security

SAPIENZA
UNIVERSITÀ DI ROMA

CIS SAPIENZA
CYBER INTELLIGENCE AND INFORMATION SECURITY

PAXOS

In Paxos, suppose that a cluster contains 5 servers and 3 of them have accepted proposal 6 with value X. Once this has happened, is it possible that any server in the cluster could accept a different value Y?
Explain your answer:
-If possible shows an execution;
-If not possible describe the rules of the algorithm that prevent such behaviour.



PAXOS

In Paxos, we have seen that each participant with id k (k in {1,2,…,n}), number its prepare requests using a number in the set m*n+k with m natural number.

That is participant 1 is allowed to prepare {1,n+1,2n+1,3n+1,…}.
Participant 2 is allowed to prepare {2,n+2,2n+2,…}.
….
Participant n prepares {n, 2n, 3n, 4n,..}.

Suppose we let each participant timestamps its prepare requests with the pair (ID, ts) (where ts is an increasing counter) (Example (1,0),(1,1,),(1,2)… for participant 1), and we use the following rule (ID',ts') > (ID, ts) if ID'>ID or ID'=ID ant ts'>ts.
Discuss if the properties of safety and liveness of Paxos are impacted.

PAXOS

Consider the modified implementation of paxos in the following slides. Discuss the Safety and Liveness properties of paxos. Specifically, if a property holds discuss why, if it does not, show a counterexample.

Algorithm 1 Paxos — Proposer p

1: **Constants:**
2: A, n, and f. {A is the set of acceptors. $n = |A|$ and $f = \lfloor (n-1)/2 \rfloor$.}
3: **Init:**
4: $crnd \leftarrow -1$ {Current round number}
5: **on** $\langle \text{PROPOSE}, val \rangle$
6: $crnd \leftarrow \text{pickNextRound}(crnd)$
7: $cval \leftarrow val$
8: $P \leftarrow \emptyset$
9: **send** $\langle \text{PREPARE}, crnd \rangle$ **to** A
10: **on** $\langle \text{PROMISE}, rnd, vrnd, vval \rangle$ with $rnd = crnd$ **from** acceptor a
11: $P \leftarrow P \cup (vrnd, vval)$
12: **on event** $|P| \geq n - f$
13: $j = \max\{vrnd : (vrnd, vval) \in P\}$
14: **if** $j \geq 0$ **then**
15: $V = \{vval : (j, vval) \in P\}$
16: $cval \leftarrow \text{pick}(V)$ {Pick proposed value $vval$ with largest $vrnd$}
17: **send** $\langle \text{ACCEPT}, crnd, cval \rangle$ **to** A

There is an implicit timeout that re-calls propose

Algorithm 2 Paxos — Acceptor a

1: **Constants:**
2: L {Set of learners}

3: **Init:**
4: $rnd \leftarrow -1$
5: $vrnd \leftarrow -1$
6: $vval \leftarrow -1$

7: **on** $\langle \text{PREPARE}, prnd \rangle$ with $prnd > rnd$ **from** proposer p
8: $rnd \leftarrow prnd$
9: **send** $\langle \text{PROMISE}, rnd, vrnd, vval \rangle$ **to** proposer p
10: **on** $\langle \text{ACCEPT}, i, v \rangle$ **from** proposer p
11: $rnd \leftarrow i$
12: $vrnd \leftarrow i$
13: $vval \leftarrow v$
14: **send** $\langle \text{LEARN}, i, v \rangle$ **to** L

This code does not sends nack. This is ok if there is a timeout on proposers.

PAXOS CODE

Algorithm 3 Paxos — Learner l

1: **Init:**
2: $V \leftarrow \emptyset$

3: **on** $\langle \text{LEARN}, (i, v) \rangle$ **from** acceptor a
4: $\quad V \leftarrow V \uplus (i, v)$

5: **on event** $\exists i, v : |\{(i, v) : (i, v) \in V\}| \geq n - f$
6: $\quad v$ **is chosen**

PAXOS

Consider the original paxos implementation (see slides on paxos). Assume that for a bug of the system two proposers may collide in a round number. That is they can generate requests with the exact same round number.

Discuss if the original algorithm still works, if it does would be possible to modify it to tolerate this kind of problems?

PAXOS

Consider a Paxos system composed by 4 acceptors {A1,A2,A3,A4} and two proposers {P1,P2}. On this system several instances of Paxos are run to solve sequential instances of the consensus problem. That is the proposers have to first decide on value one of a certain sequence, then on value two, then three and so on.

Q1: Discuss how would you solve this problem using as basic block the paxos implementation that you have seen during lecture.

PAXOS

Consider a Paxos system composed by 4 acceptors {A1,A2,A3,A4} and two proposers {P1,P2}. On this system several instances of Paxos are run to solve sequential instances of the consensus problem.

Consider now a setting where sending messages from proposer P1 is expensive.

Could you modify the basic instance of Paxos such that P1 on average sends less messages?

[HINT: Think carefully about the size of the quorums needed by P1, could you shrink it in some cases?]