CS425 Fall 2016 – Homework 3

(a.k.a. “The Martian”)

Out: Oct 13, 2016. Due: Nov 1, 2016 (Start of Lecture. 2 pm US Central time.)

Topics: Snapshots, Multicast, Consensus, Paxos, Leader Election, Mutual Exclusion (Lectures 13-18)

Instructions:

1. Attempt any 8 out of the 10 problems in this homework (regardless of how many credits you’re taking the course for). If you attempt more, we will grade only the first 8 solutions that appear in your homework (and ignore the rest). Choose wisely!
2. Please hand in hardcopy solutions that are typed (you may use your favorite word processor. We will not accept handwritten solutions. Figures and equations (if any) may be drawn by hand. Online students can email solutions, and MCS-DS students must upload on Coursera.
3. Please start each problem on a fresh sheet (not just page), and type your name at the top of each sheet. Staple all your sheets together.
4. Homeworks will be due at the beginning of class on the day of the deadline. No extensions.
5. Each problem has the same grade value as the others (10 points each).
6. Unless otherwise specified, the only resources you can avail of in your HWs are the provided course materials (slides, textbooks, etc.), and communication with instructor/TA via discussion forum and e-mail.
7. You can discuss lecture concepts and the questions on Piazza and with your friends, but you cannot discuss solutions or ideas. All work must be your own.

Prologue: It is the year 2030 A.D. Most of you are in middle age. Cloud computing, as we know today, does not exist – it’s now called “Solar Computing”. Sure, there are a few quantum computers here and there, but transistor-based computers still rule the roost in the 2030s. Datacenters are still around, and all the distributed computing concepts you’re learning today in CS425 still apply. The only catch is that datacenters are much smaller (100x) than they were back in 2015 A.D. – this means an entire AWS zone from 2015 can now be stored in one rack!

Anyway, Moon has been colonized by humans. Man has been to Mars. The next step is Mars colonization. In order to kickstart the Mars colonization process, a manned
spacecraft New Horizons X is being launched to Mars with ten astronauts on board. The spacecraft carries its own powerful datacenter. You are one of the astronauts on board. You are the sole “Solar Computing Specialist.” You must ensure that you troubleshoot and solve all problems that arise in the onboard distributed system (solving any 8 out of 11 problems would also suffice to save the mission).

Any resemblance to persons, places, things, or events, living or dead, past, present, or future, is purely coincidental.

**Problems:**

1. 3...2...1... Liftoff! You’re off to Mars. During liftoff you’re browsing code (what else?). Within the first minute after launch, you realize that one of the Earth programmers has written an algorithm for synchronous consensus (the same as that discussed in class) for a rack of N=10 machines, however they have only configured the consensus to run for 2 rounds before terminating. Your fellow astronauts believe this is not a problem, and you know they are wrong. You know that once you exit the Earth’s atmosphere, cosmic rays will increase in frequency and they can knock out an arbitrary number of machines simultaneously. Now all you have to do is to show a counter-example to convince your fellow astronauts that with an arbitrary number of failures this programmed synchronous consensus will not work. Quick, you’re about to exit the atmosphere!

2. You have detachment from the rocket! You switch communications on. You see a chart of the multicast communications between your spacecraft New Horizons X (NHX), Earth station, Moon station, and Mars (unmanned). If these stations use the FIFO Ordering algorithm, mark the timestamps at the point of each multicast send and each multicast receipt. Also mark multicast receipts that are buffered, along with the points at which they are delivered to the application.
3. As New Horizons X is passing through the Van Allen belts, the spacecraft’s reactor and engines suddenly shut down. Oops, you realize that you should have used causal ordering in the previous timeline (Question #2). Can you redo it quickly before your spacecraft crashes? Again, mention clearly all timestamps and all buffered messages.

4. To fix the consensus algorithm, one of your fellow astronauts has written a variant of the stock implementation of Paxos. In a datacenter with 1000 processes, you know already (via previous experiments) that you will never have more than 20 failures (anything more, and the spacecraft is doomed anyway, so this is a safe assumption to make). While perusing the code you realize that instead of majority (for a quorum), it uses 21 processes (everywhere in the protocol wherever a quorum was previously needed). You need to quickly figure out the answer to three questions:
   a. Is this new version live?
   b. Is this new version safe?
   c. Is this new version faster or slower than the majority (just greater than 50%) version of the protocol? Why?

5. Now your spaceship is passing by the Dark Side of the Moon. It’s a glorious view! To ensure things are working properly you decide to run the Chandy-Lamport snapshot algorithm on the ongoing communications between your spacecraft, and the manned Earth station, and manned Moon station. But due to a crash at the different stations, the algorithm only outputs the following timeline. Quick, it’s up to you to manually calculate the snapshot!
6. You’re about halfway through the trip to Mars. Unfortunately, the spacecraft has been on a wobble over the last 6 hours, and this is affecting all the astronauts adversely – many are throwing up, and two are unconscious. You’re still doing well (physically and emotionally speaking), mostly because you were trained well at Illinois. You trace the wobbling problem to the on board storage system, and the fact that there is no leader election algorithm in there! Quick, you need to design one!

The datacenter on board (with hundreds of machines) uses a ring-based DHT (among the machines) with a Chord-like routing algorithm with each peer maintaining 3 ring successors and 3 ring predecessors. This system needs to elect a leader that has the highest DHT Id in the system.

a. Design a leader election protocol that is efficient in that it uses very few messages (O(1) per participant). The only messages you can use are the DHT routing messages.

b. Argue briefly why your algorithm satisfies safety and liveness when finer tables are all correct and there are no failures during execution (proof not needed).

c. What is the completion time and number of messages in your leader election protocol?

d. Discuss briefly what might happen if failures occur during the election run, and finger tables are inconsistent.

7. Your spacecraft needs to perform a slingshot (gravity assist) in order to land on Mars. However, this means going through the dreaded Asteroid belt between Mars and Jupiter! Before the slingshot, you realize the above leader election algorithm will not work, and that for fault-tolerance you will need multiple leaders. Solve the k-leader election problem (for a given value of k). It has to satisfy the following two conditions:
• Safety: For each non-faulty process p, p’s elected = of a set of k processes with the lowest ids, OR = NULL.

• Liveness: For all runs of election, the run terminates AND for each non-faulty process p, p’s elected is not NULL.

Modify the Bully Algorithm described in lecture to create a solution to the k-Leader Election problem. You may make the same assumptions as the Bully Algorithm, e.g., synchronous network. Briefly discuss why your algorithm satisfies the above Safety and Liveness, even when there are failures during the algorithm’s execution.

8. Bam! Your New Horizons X spacecraft has just suffered a massive strike from an asteroid! Alarms are going off all around you. You quickly figure out that the fault lies with the mutual exclusion algorithm implemented in the system – if you can fix it, the spacecraft will return to normal operations.

You see that the datacenter uses the Ricart-Agrawala algorithm for mutual exclusion but instead of using the usual and boring (Lamport timestamp, process id) pair, the algorithm instead uses (FIFO local sequence number, process id) pair. The rest of the Ricart-Agrawala algorithm remains unchanged. Your fellow astronaut says this algorithm, even without failures: a) violates safety, b) violates liveness, and c) does not satisfy causal ordering. Is he right on any of these counts (which ones)? Give a proof or counter-example.

9. Whew! Now that the spacecraft has been repaired (after the asteroid strike) and the partition has healed, you realize you’re almost at Mars! To make sure nothing goes wrong during landing, it’s time to make sure the virtual synchrony implementation in the datacenter is correct. You see the following instances in the log. For each of the following executions, say whether it is a) correct (and why), or b) if it is incorrect (and what change in the timeline would have made it correct).

a. p1, p2, p3 each deliver a view V11=[p1,p2,p3]. Then p1 multicasts message M32, however then p3 fails, and p1 and p2 have deliver the next view V12=[p1,p2], and only then do p1 and p2 deliver M32.

b. p1, p2, p3 each deliver a view V11=[p1,p2,p3]. Then p1 multicasts message M32, however it is not delivered at p1, p2 or p3. Then p3 fails and p1 and p2 deliver the next view V12=[p1,p2].

c. p1, p2, p3 each deliver a view V11=[p1,p2,p3]. Then p1 multicasts message M32, and p1 delivers it immediately. However then p3 fails and p1 and p2 deliver the next view V12=[p1,p2]. Only then does p2 deliver M32.

d. p1, p2, p3 each deliver a view V11=[p1,p2,p3]. Then p1 multicasts message M32 and concurrently p2 multicasts message M45. Both p1 and p2 deliver each others’ messages, but they never deliver their own multicasts. But p3
f. p1 and p2 deliver the next view V12={p1,p2}.

e. p1, p2, p3 each deliver a view V11={p1,p2,p3}. Then p1 multicasts message M32, and delivers it immediately, and then p1 fails. p2 and p3 each respectively deliver the views {p2} and {p3}. M32 is never delivered at p2 or p3.

f. p1, p2, p3 each deliver a view V11={p1,p2,p3}. Then p1 multicasts message M32. A fourth process p4 joins, and all processes p1-p4 deliver the next view V12={p1,p2,p3,p4}. M32 is delivered then at processes p1-p4.

10. W00t! Your spacecraft has landed on Mars! As a sign of respect for your firefighting skills as the “Solar Computing Specialist” and for rescuing the mission multiple times, your fellow astronauts have unanimously decided to give you the honor of being the first person to land on Mars! But the spacecraft doors won’t open! You’re stuck in the exit hatch. Thankfully you have access to a computer, and you quickly figure out the problem may lie with the snapshot algorithm that you re-implemented. Here it is:

First, Initiator Pi records its own state

Initiator process creates special messages called “Marker” messages

for j=1 to N except i

Pi sends out a Marker message on outgoing channel C_{ij}

Starts recording the incoming messages on each of the incoming channels at Pi: C_{ji} (for j=1 to N except i)

Whenever a process Pi receives a Marker message on an incoming channel C_{ji}

if (this is the first Marker Pi is seeing)

Pi records its own state first

Marks the state of channel C_{ji} as “empty”

for j=1 to N except i

Pi sends out a Marker message on outgoing channel C_{ij}

Starts recording the incoming messages on each of the incoming channels at Pi: C_{ji} (for j=1 to N except i)

else // already seen at least one Marker message
– if this is the (N-1)th (last) marker being received at $P_i$,
  
  for $j=1$ to $N$ except $i$

  Mark the state of channel $C_{ji}$ as all the messages that have arrived on it (until now) since recording was turned on for $C_{ji}$

  else do nothing

Terminate when all processes have received (N-1) markers each

a. Is this algorithm correct? If yes, prove so. If no, give a counterexample (draw a timeline).
b. How would you fix this algorithm? Quick, your oxygen is running out!
c. (Optional, no points for this part, answer only if you want to) When you set your foot on Mars, as the first human to do so, what will be your first words to the world? (Neil Armstrong had great words, but try to make yours epic!).