CS425 Fall 2023 – Homework 3

(a.k.a. "2023: A Space Odyssey")

Out: Oct 10, 2023. Due: Oct 30, 2023 (2 pm US Central time.). It's a Monday!

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). Please do not attempt more than 8. If you attempt more, that creates more work for TAs, so we will take the **lowest** 8 scores among your questions. Choose 8 wisely!
2. Please hand in **solutions that are typed** (you may use your favorite word processor. We will not accept handwritten solutions. Figures (e.g., timeline questions) and equations (if any) may be drawn by hand (and scanned).
3. **All students (On-campus and Online/Coursera)** – Please submit PDF only! Please submit on Gradescope. [https://www.gradescope.com/]
4. Please **start each problem on a fresh page**, and **type your name at the top of each page**. **And on Gradescope please tag each page with the problem number!**
5. Homeworks will be **due at time and date noted above. No extensions. For DRES students only:** once the solutions are posted (typically a few hours after the HW is due), subsequent submissions will get a zero**. All non-DRES students must submit by the deadline time+date.**
6. Each problem has the same grade value as the others (10 points each).
7. Unless otherwise specified in the question, 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.
8. You can discuss lecture concepts and the questions on Piazza and with your friends, but you cannot discuss solutions or ideas on Piazza.

Prologue: It is the year 2049 A.D. Most of you are in your middle age. Cloud computing, as we know today (2023), 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 2049. 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 (1000x) than they were back in the 2020s, but more powerful – this means an entire AWS zone from 2020s can now be stored in one small room on a small spaceship!

Anyway, Moon and Mars have long since been colonized by humans. Humankind is next going to land on Saturn. A manned spacecraft New Horizons X is being launched to Saturn. Once on board, you meet the astronaut team led by Commander Amelia Brand, Pilot Rheya Cooper, and including you and ten other astronauts. The spacecraft carries its own powerful datacenter. You are the sole "Solar Computing Specialist." You must ensure that you troubleshoot and solve all problems that arise in the on-board distributed system (solving any 8 out of 10 problems would also suffice to save the mission; space can be forgiving that way).

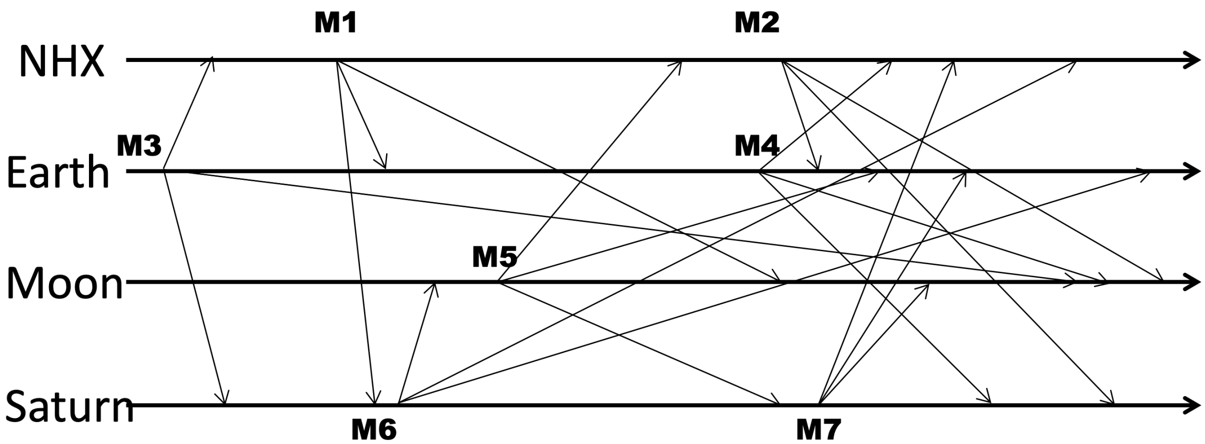
All characters and storylines are fictitious, and purely intended to keep the reading entertaining; these are not intended to be educational. Any resemblance to persons, places, animals, things, or events, living or dead, past, present, or future, is purely coincidental. No animals were harmed in the production of this homework.

(If you read the first part of each question, you will see a story arc that can be made into a movie. There is a twist ending! So be sure to read the first parts in order 1, 2, 3… , 9, 10. Can you discover all old Sci-fi and cartoon references in this homework? No bonus points for finding these, alas, just fun.)

<u>**Problems**</u>:

1. 3…2…1… Liftoff! You're off to Saturn. During liftoff you're browsing code (what else?). You realize that one of the Earth programmers has written an algorithm for synchronous consensus (the same as that discussed in class), but your datacenter is asynchronous! 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. Point to where the synchronous consensus proof (discussed in class) breaks down when the system is asynchronous. Quick, you're about to exit the atmosphere!
2. You have detachment from the rocket! Suddenly you see your pet cat (who you had named "Doraemon" when you adopted her) in the corridor of the spaceship—you try to follow her but she disappears. You wonder if it's your imagination. Anyway, you figure you have bigger cats to catch… Along with Pilot Rheya Cooper, you switch the communications on. You see a chart of the multicast communications between your spacecraft New Horizons X (NHX), Earth station (Earth), Moon station (Moon), and unmanned Saturn (Saturn). If these stations use the *FIFO* Ordering algorithm, mark the timestamps at the point

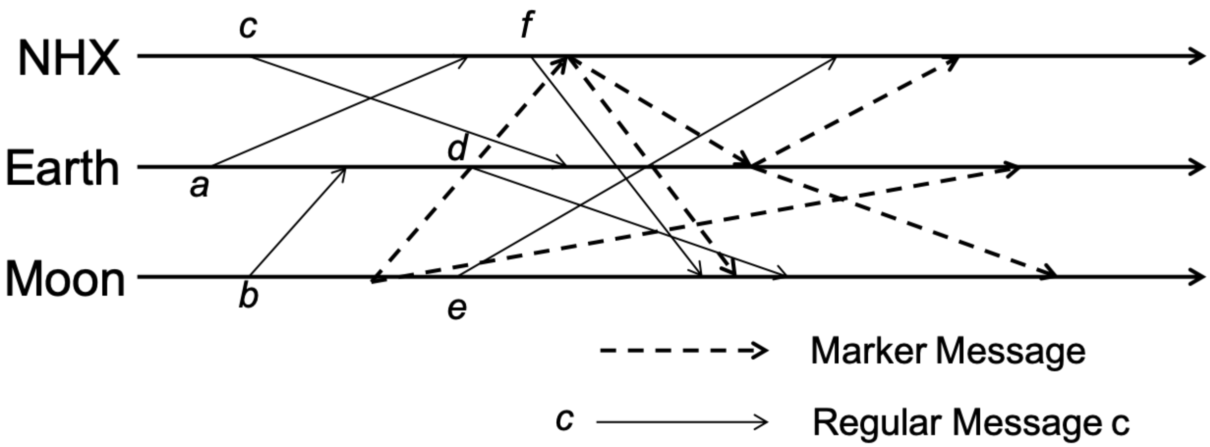
of each multicast send and each multicast receipt. Also mark multicast receipts that are buffered, along with the points at which they are delivered to the application.



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 (previous question). Commander Amelia Brand asks you to redo it quickly before your spacecraft crashes! Again, mention clearly all timestamps and all buffered messages.
4. Just this morning you also saw your pet dog (whom you had named "Einstein" when you adopted him on Earth) roaming inside the spaceship's corridor. You called out to him by name and he stared at you, but then ran away. You are perplexed, and you talk to the captain of your spaceship, Commander Amelia Brand. She asks you to take some rest. But there are miles to go before you sleep… Suddenly you wake up and see that you are looking at the FLP proof discussed in class. You are puzzled by a couple of things. Can you explain them briefly?
   i. You don't quite understand how, in Lemma 3, Case II, one can have p'=p, especially since the set C is obtained by *not* applying event e=(p,m)! Can you explain this discrepancy?
   ii. Why in the proof of Lemma 3, Case II does C *definitely* have a deciding run, i.e., schedule S? What if C never decides? Does the proof not hold then?

iii. It appears that Lemma 3's proof will hold no matter which event *e* is selected for the argument (*e* applicable on configuration C). Is this strue, or does *e* need to have some special characteristics?

5. Now your spaceship is passing by the Dark Side of the Moon. It's a glorious view! To ensure things are working properly Commander Amelia Brand and Pilot Rheya Cooper ask you 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. In the figure, a, b, c, … are regular application messages. You can use S(a) to denote the send event of a and R(a) to denote its receipt event. Markers shown as dotted lines. Can you help the intern find the snapshot recorded by this run? Don't forget to include both process states and channel states. For process states, you can use the name of the latest event at that process (For initial state, just say "Initial state". Note that Markers don't count as events). Quick, it's up to you to manually calculate the snapshot!



6. Your spacecraft needs to perform a slingshot (gravity assist) in order to land on Saturn. 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 *Quorum-based election* Algorithm described in lecture (used inside Chubby, and inside Paxos) to create a solution to the k-Leader Election problem. You may make the same assumptions as the Chubby election Algorithm. Briefly discuss why your algorithm satisfies the above Safety and Liveness, even when there are failures during the algorithm's execution.

7. You're still doing well physically and emotionally in this long trip, mostly because you were trained well at Illinois. You're about halfway through the trip to Saturn. As you're retiring to your room to sleep, you see both your cat Doraemon and dog Einstein walking together in the spaceship corridor. You call out to them, but they run away again. Trippy, Dude! Before you can chase them, you notice the spacecraft wobbling quite a bit, and you need to fix this. You trace the wobbling problem to the on-board storage system, and the fact that there is something wrong with the leader election algorithm in there! Quick! You start looking through the code and realize that someone has implemented the Bully algorithm. But the system on board is an asynchronous system! Quick, create an example to convince your captain that the Bully algorithm will not work without synchrony assumptions!
8. Bam! Your New Horizons X spacecraft has just suffered a massive strike from an asteroid! Alarms are going off all around you. And a dog can be heard yelping and a cat can be heard whelping in agony, somewhere inside the spacecraft. You talk to Commander Brand and your colleague on board Pilot Rheya Cooper about this, and they counsel you that the animals are your imagination. Anyway, back to work… You quickly figure out that the alarms are because of the mutual exclusion algorithm implemented in the system – if you can fix it, the spacecraft will return to normal operations.

   You see that someone has implemented the Maekawa algorithm, but to "make it more efficient" (a dangerous phrase, if there was one!) they have "optimized" the voting set members similar to how Ricart-Agrawala works. In other words, when a voting set member *pi* receives a Release message from a process *pj* (which is exiting the critical section), the voting set member process *pi* now immediately multicasts a Reply message to *all* waiting processes in pi's queue, and empties the queue. Give a concrete scenario where this algorithm violates mutual exclusion.
9. Whew! You are almost there! Now that the spacecraft has been repaired (after the asteroid strike) and the partition has healed, you realize you're almost at Saturn! You're no longer seeing your dog Einstein and cat Doraemon (though you kinda hear them sometimes, which makes you question your own sanity). You notice that there are fewer astronauts up in the command center of the spacecraft—you say to yourself they're all probably resting up for the landing.

Suddenly you notice an issue with misordered deliveries of multicasts aboard the system. You quickly narrow it down to how virtual synchrony was implemented in the on-board distributed system. You try very hard to remember the basics of Vsync (virtual synchrony) that you learnt in CS425. For each of the following questions about VSync, say whether they are true or not (i.e., they satisfy VSync or not) – additionally, please justify WHY they are true/false, and if you select false, please also say how Vsync actually should behave. Quick, before your spaceship crashes!

i. A set of N nodes are in a view. The next view delivered at each of those nodes contains only that node itself.
ii. A set of 2 nodes P1, P2, are in a view V1. The very next view V2 delivered at P1 contains only {P1, P2}. The next view V2 delivered at P2 contains only {P2}.
iii. A node joins the system (at a view), does not send any multicasts, and even though it did not fail, it leaves immediately in the subsequent view.
iv. A failed node's multicasts do not need to be considered, i.e., it is ok if some members which survive to the next view receive this failed node's multicasts and some of these surviving members don't.
v. A node that joins in the middle of a view gets included in the current view but it does not need to receive any of that view's multicasts, and can wait until the next view.

10. W00t! Your spacecraft has landed on Saturn! As a sign of respect for your firefighting skills as the "Solar Computing Specialist" and for rescuing the mission multiple times, all your fellow astronauts, and Commander Amelia Brand and Pilot Cooper, have unanimously voted to give you the honor of being the first human to land on Saturn! But the spacecraft doors won't open! You're stuck in the exit hatch. Thankfully you have access to a terminal, and you quickly figure out the problem *may* lie with the snapshot algorithm that you implemented to coordinate all the spaceship doors. Consider a file F that is present in a distributed system of N processes (N large). There are no failures or message losses in the system. The mutual exclusion required on this file has the following safety and liveness conditions (different from those discussed in lecture):

Safety: At most *k/2* processes (*k* is even) may obtain write access to the file simultaneously. At most *k* processes may obtain read access to the file simultaneously. If any process has write access to F, no other process should be able to read it. If any process has read access to F, no other process should be able to write it.
Liveness: Requests to access and release the resource eventually succeed.

Answer these three parts:

a. Briefly describe a token ring-based distributed algorithm for the above problem (pseudocode would be a good idea). Your algorithm must not have more than *k* token messages in the system at any point of time (simultaneously). Hint: Token message can contain writable fields.
b. Argue briefly that your algorithm guarantees all the Safety clauses. (a formal proof is not required, however you are free to write one).
c. Can your algorithm livelock, i.e., violate Liveness? Suggest an idea to address this issue. Argue that your idea reduces the frequency of livelocks (you don't need to prove Liveness).
d. Given one process that is currently writing, and *k* processes waiting to read, what is the synchronization delay (i.e., time for *all* the *k* processes to start reading, i.e., the last one)? Calculate both best case and worst case. Since this calculation may be hard, you can assume for simplicity that: i) reads take quite long (i.e., tokens are not released by readers until everyone has started reading), ii) tokens *cannot* be combined into one message, and iii) in one time unit, only one message can be transmitted, anywhere in the system (the latter means that it suffices to calculate the total number of token transfers for the synchronization delay). Also N >> *k*. If you need to make other simplifying assumptions, be reasonable and specify them clearly. Show all your calculations.
e. (Optional, no points for this part, answer only if you want to) When you set your foot on Saturn, 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!).

====== (UN-OFFICIAL) END OF HOMEWORK 3 =====

<u>Epilogue with the Twist Ending (Read only after until you've read the Prologue and stories in all questions above)</u>: As you take humankind's first steps on Saturn, you look back at the Horizons X lander spacecraft. You see your dog Einstein and cat Doraemon together peering down at you through the porthole window. You remember they are indeed real, and that you did indeed bring them along with you from Earth! The long trip and cryogenic sleep made you woozy and forgetful! You realize that Einstein and Doraemon were just too disoriented by the space travel experience, and that's why they kept running away from you throughout the trip. It all makes sense now!

The Earth station, from millions of miles away, speaks in your earpiece, "Congratulations! You just completed the first solo human mission to Saturn! Woohoo!" You're happy, but then you stop and ask Earth station, "What do you mean – "Solo mission"?! What about the other ten astronauts? What about Commander Amelia Brand and Pilot Rheya Cooper who were with me?" There is a pause. Earth station responds, “Ten astronauts? Brand and Cooper…? What kind of names are those…? Do you feel alright?...”

--- The End ---

(PS: Did you catch all the sci-fi references in this homework?)

====== (OFFICIAL) END OF HOMEWORK 3 =====