CS425 Fall 2025 – Homework 3

(a.k.a. "2025: A Space Odyssey")

Out: Oct 13, 2025. Due: Nov 2, 2025 (2 pm US Central time.).

Note that the Deadline is on a Sunday at 2 PM US Central!

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

Instructions:

1. Please double-check that the top of this page mentions the correct semester you are taking the course in (otherwise, you are looking at an old version and we will not accept your submitted solutions).
2. **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!
3. 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).
4. **All students (On-campus and Online/Coursera)** – Please submit PDF only! Please submit on Gradescope. [https://www.gradescope.com/]
5. 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!**
6. 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.**
7. Each problem has the same grade value as the others (10 points each).
8. 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.
9. You can discuss lecture concepts and the questions on Piazza and with your friends, but you cannot discuss solutions or ideas on Piazza.
10. Like for any assignment, the instructions for this homework are a union (not an intersection) of the instructions included in this PDF and the instructions

(includes clarifications) on any pinned Piazza post related to this homework. Please follow instructions on Piazza carefully.

Prologue: It is the year 2049 A.D. Most of you are in your middle age. Cloud computing, as we know today (2024), 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. The newly elected President of the Earth, a braggadocio named L. Skum, has used his company to fund a manned spacecraft New Horizons X that will be launched towards 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.

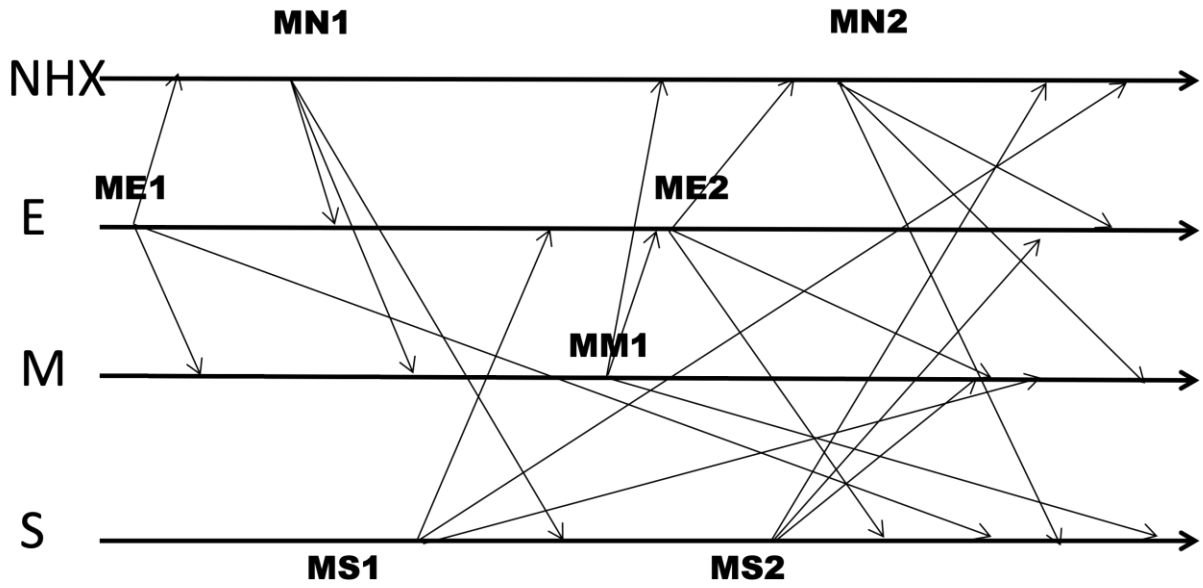
(If you read the first part of each question, you will see a story arc that can be made into a movie. There is a twist ending! So be sure to read the first parts in order 1, 2, 3… , 9, 10. Can you discover all old Sci-fi and cartoon references in this homework? No bonus points for finding these, alas,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

Problems:

1. 3…2…1… Liftoff! You're off to Saturn. During liftoff you're browsing code (what else?). Within the first minute after launch, you realize that one of the runs for the synchronous consensus (the same as that discussed in class) may have a bug. Concretely, see the synchronous consensus algorithm for a system of $N > 5$ machines assuming $f=4$ failures, and imagine it being run in an asynchronous system. All rounds are synchronous (i.e., deliver multicast messages correctly to

all as-yet non-faulty processes) *except* that in *each* of the 3rd *and* 4th rounds, one multicast message is completely dropped (i.e., in each of those rounds, one multicast message from some sender process is not received at any recipient process). Will all such runs of the algorithm still be correct? If yes, prove it. If no, show a counter-example. Quick, you're about to exit the Earth's 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 (E), Moon station (M), and unmanned Saturn (S). If these stations use the Causal 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 *total ordering* in the previous timeline (previous question). Total ordering uses a sequencer. Fortunately for you while this is an asynchronous system you know that (i) the above timeline shows the relative physical times of multicast send events, and (ii) because the sequencer uses quantum computing and entanglement) that the latency between any process Pi to and from the sequencer is zero (note that other process to process communications still have latency, as

shown above). Commander Amelia Brand asks you to show the list of all messages that will be buffered at each of the processes. Please show your full work, don't just give lists or counts of messages.

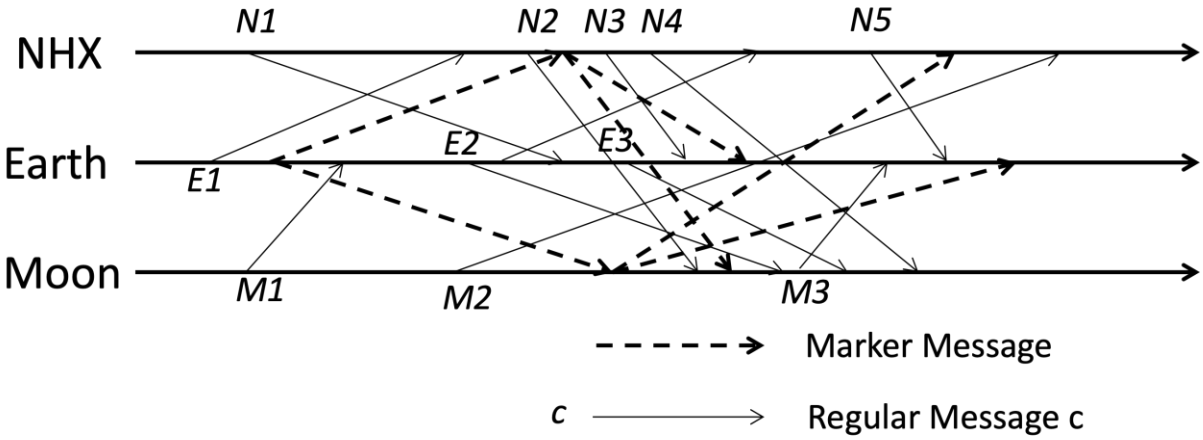
4. Just this morning you also saw your pet dog (whom you had named "Einstein" when you adopted him on Earth) roaming inside the spaceship's corridor. You called out to him by name and he stared at you, but then he ran away. You are perplexed, and you talk to the captain of your spaceship, Commander Amelia Brand. She says you are tired and asks you to take some rest. But there are miles to go before you sleep… To fix the consensus algorithm, Commander Amelia Brand and another of your 10 fellow astronauts, have together written a variant of the stock implementation of Paxos. You realize there is a bug. In a datacenter with N processes (N large enough), you realize that in some places in the implemented algorithm, instead of majority (for a quorum), it uses L=((4N/9)+1) processes. There are 3 variant algorithms:
   i. Both Election phase and Bill phase use L.
   ii. Election phase uses L instead of N/2+1, but Bill phase uses N/2+1.
   iii. Election phase uses N/2+1, but Bill phase uses L.
   iv. All phases use L instead of N/2+1.

   For each of the above **four** cases and fail-stop model, answer the following 3 questions:

   a. Is this new version live? Justify.
   b. Is this new version safe? Justify.
   c. Is this new version faster or slower than using the majority? 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 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, Ni, Ei, and Mi (for different values of i) are regular application messages. For a message A, you can use S(A) to denote its send event and R(A) to denote its receipt event. Markers shown as dotted lines. Unfortunately the onboard computer HAL (originally built at UIUC!) tells you that there are at least 3 causality violations in this snapshot. Can you (1) find at least 3 causality violations? (2) debug what caused those causality violations? (3) fix the causality violations by drawing a new

timeline where all conditions are obeyed (new timeline should only fix the root cause of those violations, but not change anything else). For process states, you can use the name of the latest event at that process, e.g., for Earth the local snapshot can be LocalSnapshot(Earth at Send(E1)). For initial state of any process, just say "Initial state". Quick, it's up to you to manually fix the snapshot!



6. 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. That's Toooootally 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 no leader election algorithm in there! To make things worse, the leader node elected has named itself "Hal", and this single leader seems to be acting up! Hal is threatening to take over the entire space ship and kill everyone on board! Jeez Louise, you feel like you've seen this movie somewhere! Quick, you need to design a new one!

   You decide to solve the *k-leader* election problem (for a given value of k, which is an integer), so that a single leader cannot monopolize everything. 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 *Ring election* Algorithm described in lecture to create a solution to the k-Leader Election problem. You may make the same assumptions as the original

algorithm. Briefly discuss why your algorithm satisfies the above Safety and Liveness, even when there are failures during the algorithm's execution.

7. 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… Commander Amelia Brand, Pilot Rheya Cooper, and 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 the datacenter uses the Ricart-Agrawala algorithm for mutual exclusion but instead of using the usual and boring (Lamport timestamp, process id) lexicographic pair, the algorithm instead has a bug which uses (process id, Lamport timestamp) lexicographic pair. The rest of the Ricart-Agrawala algorithm remains unchanged (i.e., is not buggy). Your fellow astronaut says this algorithm, even without failures: a) violates safety, b) violates liveness, and c) does not satisfy causal ordering. Are they right on any of these counts (which ones)? Give a proof or counter-example for each.

8. 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 (VSync) 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 correctly. Quick, before your spaceship crashes!

   i. A set of N nodes are in a view, and deliver that view containing N nodes. All nodes fail before sending or receiving any messages or further views.

ii. A set of N nodes are in a view, and deliver that view containing N nodes. The next view delivered at each of those nodes contains only that node itself, and no other nodes.
iii. A set of 3 nodes P1, P2, P3, are in a view V1. The very next view V2 delivered at P1 and P2 contains {P1, P2}. The next view V2 delivered at P3 contains {P1, P2, P3}.
iv. A set of 3 nodes P1, P2, P3, are in a view V1. The very next view V2 delivered at P1 contains {P1, P2}. The next view V2 delivered at P2 contains {P1, P2}. The new view V2 delivered at P3 contains {P3}.
v. A node Pi joins the system (at a view) and all views (including its own) now contain it (and all other nodes). Then no node sends any multicasts in that view. Finally, even though Pi did not fail, Pi leaves immediately in the subsequent view, and all views of non-Pi processes exclude only Pi, and Pi's view includes only Pi.
vi. A node Ni joins the system (at a view), sends a multicast, and then Ni fails immediately. No other nodes fail, and the next view excludes (only) Ni but contains all other nodes. Ni's single multicast is delivered to all the surviving processes in the new view (which excludes Ni).
vii. A node Ni joins in the middle of a view and gets included in the current view but it does not need to deliver any of that view's multicasts, and can wait until the next view.

9. You're almost there, but the doors won't open! Doraemon is going crazy in the background and Einstein is roaring like a tiger. Someone on board just started playing a Ukulele and started singing "You are my sunshine" but with some strange lyrics that talk about Lamport timestamps. You're absolutely losing your mind! You breathe gently, remember your meditation lessons, and that helps calm you down. You figure out that it's because an infamous ukulele-playing professor implemented Maekawa's algorithm, and this algorithm has now deadlocked. You get a flash of a brilliant idea to modify Maekawa's algorithm to make it deadlock free. The idea is this: at a process Pi that is inside enter(), after setting its state to Wanted, instead of multicasting the request to all its voting set (Vi) members, Pi *sorts* the members in Vi lexicographically (lowest id to highest id), and sends them requests sequentially, waiting after each request (to a Vi member) to receive a Reply, before moving on to the next member from Vi. Answer two questions:
    i. Is this algorithm safe? That is, does it ensure mutual exclusion?
    ii. Is this algorithm deadlock-free?

iii. Does this algorithm avoid starvation? Starvation occurs when there is no deadlock, but still some process stays stuck inside enter() for infinite time.

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 a snapshot algorithm that you implemented to coordinate all the spaceship doors. Here is the snapshot algorithm for process Pi:

→ If **P*i* is the Initiator, Pi process creates special messages called "Marker" messages, and then follows the rest of the algorithm below.**

→ **For any process Pi that receives a marker on incoming channel** C_{ki} (note that *k* does not exist for the Initiator)

if (this is the first Marker P*i* is seeing) **// always true at Initiator process at the initiation point**

P*i* records its own state first

Mark the state of channel C_{ki} as "empty"

for *j=1 to N* except *i, k*

P*i* sends out a Marker message on outgoing channel C_{ij}

Start recording the incoming messages on each of the incoming channels at P*i*: C_{ji} (for *j=1 to N* except *i, k*)

else **// already seen at least one Marker message**

Stop recording C_{ki} and mark that channel's state as the sequence of messages that have been received on all channels C_{ji} *(where j=1 to N* except *i)* since Pi received its first marker

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

i. Is this algorithm correct? If yes, prove so. If no, give a counterexample (draw a timeline).

ii. How would you fix this algorithm? Try to have the minimal number of fixes. Quick, your oxygen is running out!
iii. (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 on the Moon, but try to make yours epic!).

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

Epilogue/After Credits Scene with the Twist Ending (Read only after until you've read the Prologue and stories in all questions above): 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?..."

Finally, in a shocking second after-credits twist, President of the Earth L. Skum suddenly comes online and says, "That was a good simulation. Just like the rest of the universe!"

--- The End ---

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

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