1. Given a message-passing asynchronous network $N$ and a distributed algorithm $A$, recall the three definitions for time:

- $f(A)$ is the synchronous time of $A$. That is, the time it takes the algorithm to terminate assuming all processors start simultaneously, and working synchronously (so that all messages sent at time $t$ are received before time $t + 1$).
- $g(A)$ is the longest-chain time of $A$. This is the longest chain of messages, assuming any non-empty subset of processors as starters.
- $h(A)$ is the bounded-delay time. This is the longest time until termination, assuming any non-empty subset of processors as starters, and assuming that each message is delivered within less than 1 time unit.

Consider the 5 possible cases: all three measures are equal, only two of them are equal and differ from the other (three cases), and all three measures are distinct. Give examples of networks and algorithms, for as many of the five as you can.

2. The question relates to the PIF algorithm of Segall.

(a) Proof the correctness and message complexity of the PI algorithm (slide 40 in the introduction). Consider the case of either one initiator or multiple initiators.

(b) Write a pseudo code for the PIF algorithm for the case of a single initiator, and briefly explain its correctness.

3. The question relates to Franklin’s $2n \log n$ algorithm for election bidirectional in rings (or Peterson’s 1st algorithm for election in unidirectional rings).

(a) Assume a set of $n$ identities, that all $n!$ permutations are equally probable, and that all processors start the algorithm as candidates. What is the expected number of processors that will remain candidates after the first phase? Give a direct combinatorial proof.

(b) In a variation of Franklin’s algorithm, a process stays to the next phase only if it is greater than the $k$ processes of each of its sides (maximum of $2k + 1$ processes). What is the message complexity of the algorithm? What is the best $k$?

4. Prove that Peterson’s 1st algorithm (to find a leader in a unidirectional ring) always determines exactly one processor as a leader.