COMP 4541 – Final Project

A. K. Goharshady

Release Date: April 23, 2024
Deadline: May 18, 2024 (23:59 HKT)

This project accounts for 30% of your total grade. You should submit your solution as two files, one `pdf` and one `sol`. As usual, handwritten and scanned solutions will not be accepted since the TAs might be unable to read your handwriting. However, you can draw your figures, if any, by hand. Your submission must be entirely your own work. All submissions will go through a plagiarism check. If you submit the same solutions as another student, you will both get a grade of F for the whole course. The deadline is firm and no extensions will be granted. You will *not* be allowed to resubmit the project. If you cannot come up with a protocol that satisfies all the requirements, try to achieve as many of them as you can. We will give you partial credit.

Exercise 1

This exercise accounts for 30% of your total grade in this course.

Inspired by proof-of-stake, Amir would like to create an Ethereum smart contract in which people can lock up their funds (in ether) and then vote on a very important matter. The matter can be resolved in one of k ways, thus each person's vote is an integer $\{1, 2, \dots, k\}$. Amir is asking for your help. Specifically, he would like to have a protocol (explained in English in a pdf file) and a smart contract (implemented in solidity) that provides as many of the following functionality/desired properties as possible:

a. Anyone on the Ethereum network can sign up in the protocol by depositing 1 ether. One may sign up several times and thus pay a larger overall deposit.

b. At the end of the protocol, everyone can get their deposits back. The returned value may be a bit smaller than the original deposit, e.g. 0.9 ether, if you need to pay for something in your protocol.

c. Each deposit of 1 ether entitles the depositor to one vote.

d. After a particular deadline t_1, the contract stops accepting deposits and instead only accepts votes by those who have already deposited money.

e. No one can vote twice for the same deposit.

f. After a later deadline t_2, the contract stops accepting votes.

g. After an even later deadline t_3, everyone with access to the blockchain should be able to figure out which option from the set $\{1, 2, \dots, k\}$ received the maximum number of votes. If there is a tie, you can break it arbitrarily as you wish.

h. No one should be able to know another person's vote.

i. If Alice has voted for $i \in \{1, 2, \dots, k\}$ and she wants to prove this to Bob, she should be able to do so. You can assume that Bob has access to the blockchain.

j. Your contract should not have any of the security vulnerabilities discussed in the lectures.

k. Your contract should use as little gas as possible and be affordable. You should find out exactly how much gas each one of your functions uses in the worst case and report it in your pdf file.

Finally, you are allowed to have participants in the contract who take roles other than voting, but any such participation should be well-incentivized, i.e. you should argue why they can make money by helping you run your contract if your protocol depends on their participation.