Final Project/Problem Set

DONGNING GUO

Due on Thursday December 9, 2021 by 11 PM.

We are giving everyone the same assignment instead of letting you choose between different topics. You can work individually or in groups of two, but no more than two. No student is allowed to work for multiple groups. If you work as a group of two, only one of you should upload your work, which should clearly specify both names. It is fine to ask and answer technical questions on Piazza. Teams (including teams of one) are not allowed to share code or wholesale solution/ideas either on or off Piazza. Violations of academic integrity requirements will be taken seriously.

Throughout this assignment we use "the notes" to refer to the notes posted on Canvas for weeks 4, 5, and 6 of the quarter on the security, latency, and throughput properties of the proof-of-work longest-chain-win protocol (called the Nakamoto consensus in the notes). Unless explicitly defined here in this assignment, the formal model and definitions follow the notes.

Recall that we introduced three key parameters:

(1) the aggregate honest mining rate h, in blocks per second;
(2) the aggregate adversarial mining rate a, in blocks per second;
(3) the block propagation delay upper bound Δ, in seconds.

Recall that blocks are numbered (Definition 1 in the notes). We also assume that honest mining is highly distributed so that no honest miner will mine two honest blocks in a row.

A sample code is posted along with this assignment. The code succinctly generates all block mining times, which can also be useful for both Problems 1 and 2.

PROBLEM 1 PRE-MINING LEAD.

Definition 1 (Pre-mining lead.). *The pre-mining lead (of the adversary) at time t is the height of the highest block mined by time t minus the height of the highest honest block mined by time t. We use L_t to denote the pre-mining lead at time t.*

By definition, $L_t \geq 0$ for all $t > 0$. In particular, by time t, if the adversary mines any block that is higher than the highest honest block, then the adversary has a strictly positive lead; otherwise the lead is equal to zero. Over time the lead's fluctuation depends on the random mining times and the adversarial mining strategy.

If the adversary's goal is to attack a specific honest block b (defined in some ways unimportant in this problem), the adversary can begin the attack in advance by trying to develop a pre-mining lead. If the adversary has a positive lead when the target block b is mined, the adversary is more likely to win the race than in the case of zero lead.

A specific adversarial strategy we introduced and often discussed is the private-mining attack. In such an attack the adversary mines its own blockchain in private and let the honest miners extend a public blockchain. To minimize the growth of the public blockchain, all honest blocks are delayed by the maximum amount. A formal definition of private mining is given here:

Definition 2 (Private mining). *Every honest block is maximally delayed before reaching other honest miners. Beginning from time 0, the adversary mines a private chain as long as it is no shorter than any other chain; otherwise, it mines after a highest honest block for a new longer private chain.*

Answer the following questions:

(1) [4 points] Let $h = 0.12$, $a = 0.08$, and $\Delta = 0$. Assume the adversary mines in private from the origin of time $t = 0$. Simulate one realization of L_t, $t \in [0, 1000]$, i.e., observe the lead over the first 1,000 seconds. Plot L_t as a function of time t from 0 to 1000 (note that time is continuous). During this period, L_t takes some integer values. Describe the distribution of L_t using a histogram. Explain your findings. [Hint: The lead can only change when a block is mined. In other words, let t_n denote the mining time of block n; for every $n = 1, 2, \dots,$ the lead remains constant during $[t_{n-1}, t_n)$. For fixed very large t, L_t has a simple, well-known distribution in the special case of $\Delta = 0$. You can develop a deeper understanding by finding this distribution in an analytic form, but this is not required for receiving full credit.]

(2) [4 points] Assume $\Delta > 0$. Is the following conjecture true?

Conjecture 1. *Private mining maximizes the lead at all times.*

If the conjecture is not true, describe a counter example. If it is true, provide a strong justification; ideally provide a concise, rigorous proof. [Hint: The question is whether a different adversarial strategy could ever achieve a larger lead at any point in time for any realization of the mining times. How might the lead change from $L_{t_{n-1}}$ to L_{t_n} upon the mining of an adversarial block? How might the lead change upon the mining of an honest block? Answer these two questions under an arbitrary strategy; then answer these questions under private mining. What can you conclude?]

PROBLEM 2 SIMULATING THE PRIVATE-MINING ATTACK

We consider an attack that targets a certain block number b. (For instance, one can target block number $b = 1234$, which is the 1234-th block, whose height is in general smaller than 1234.) Because the adversary here is only interested attacking an honest block, so we assume that block b is honest. I repeat some definitions found in the notes.

Definition 3 (Confirmation by depth of k). *If some t-credible blockchain of height $h_b + k$ or higher includes block b, then block b is said to be confirmed by depth of k at time t.*

Definition 4 (Safety violation (by depth)). *If block b and a different block at the same height are both confirmed by depth of k (possibly at different times), we say the safety of block b is violated, so is the safety of its height h_b.*

Definition 5 (Private-mining attack on honest block b). *Every honest block is delayed by the maximum amount before reaching other honest miners. Until block b is mined, the adversary mines on a private chain as long as it is no shorter than any other chain; otherwise, it mines off a highest honest block for a new longer private chain (the purpose of this pre-mining is to try to gain a lead). If an honest node mines block b, the adversary then mines on the longest chain that does not contain block b.*

If the target is block b, the adversary can begin the attack well before block b is mined by trying to develop a pre-mining lead. If the adversary has a positive lead when block b is mined, the adversary is more likely to win the race than in the case of zero lead.

Assume confirmation by depth of $k = 5$. Simulate the private-mining attack:

(1) [4 points] Assume $\Delta = 0$. Assume $b = 1$. In this case, block b is the first block mined after the genesis block, there is no pre-mining lead. Let $h = 0.7\lambda$ and $a = 0.3\lambda$. For $\lambda = 0.01$, 0.2, and 1.0, respectively, simulate the private-mining attack to estimate the probability of safety violation to within 0.01 of the true probability. Report your estimate and also submit your code. [Hint: For the desired accuracy, you may need to simulate a few thousand trials. See the sample code posted along with this assignment written in Julia. The code's running time is under one second. Feel free to use your favorite programming language.]

(2) [4 points] With all else the same as in part 1, estimate the probability of safety violation in steady state, by which we mean the bock number b is so large that the probability no longer depends on b. In this case the adversary would have a (random) pre-mining lead. [Hint: Simulate the attack with pre-mining from the origin of time $t = 0$. Set b to be a reasonablly large number so that your simulation does not take long. Is $b = 100$ sufficient?]

(3) [4 points] With all else the same as in part 1, estimate the probability of block 1's safety violation in the case of $\Delta = 1$. In this case the honest blocks are subject to delays of one second, so that the growth of the honest blockchain is slower. Note that $b = 1$ in this case, so there is no pre-mining.

Submit your work as two files, including one PDF file which includes your answer and all graphs, and a second file that includes all your code (make it a .zip file if your code consists of multiple files).