The Theory Behind Blockchains (Spring 19)
Problem Set 4

Submission: By 15:00 on May 29th, 2019. To be graded by Nathan Geier.

Instructions: You can collaborate with each other and consult external resources. However, you should write the solution on your own, and for each question, list all collaborators and external resources. Otherwise, write explicitly "None". Do not discuss solutions in the course forums; you are welcome to ask for clarifications if needed.

1 Proofs of Space and Depth Robust Graphs (50 points)

In this question let G be a directed acyclic graph (DAG) over n vertices and assume that G has an efficient representation — for any node i you can compute in time polylog(n) all of its predecessor nodes

$$\{j : \text{there's an edge from } j \text{ to } i\} \ .$$

We say that the graph is (d, r)-depth-robust, if after removing any r nodes, there is still a (directed) path in the graph of length d.

For a function H, let $L_H : [n] \to \{0, 1\}^{256}$ be a labeling of G computed as follows:

- If i has indegree 0, define $L_H(i) = H(i)$. Otherwise,
- let $j_1 < j_2 < \cdots < j_m$ be its predecessor nodes, and define $L_H(i) = H(i, L_H(j_1), \ldots, , L_H(j_m))$.

Consider the following candidate (P, V) for a proofs of space. In an initialization phase, both the prover P and verifier V compute the labeling L_H of G. In the proof phase, the verifier chooses a random node $i \leftarrow [n]$, and asks the prover for the label of i. Upon receiving the answer, the verifier makes sure it is consistent with the label $L_H(i)$.

a. **(15 points)** For any d, r, describe a (d, r)-depth-robust graph G on $n = \Theta(dr)$ nodes with diameter n (i.e., the maximal shortest path's length between any two nodes is n), where the prover can always successfully answer the verifier queries using at most $O(r \log(n))$ space and d oracle calls to H.

b. **(15 points)** Assume that G is $(n/3, n/3)$-depth-robust. Let P^* be a prover that stores the labels $L_H(i)$ for $n/4$ nodes and makes at most $n/6$ queries to H. Show that when H is modeled as a random oracle, P^* fails to answer the verifier's query with probability at least 0.16.

c. **(20 points)** In the suggested candidate the verifier has to compute the right label $L_H(i)$ every times it verifiers a query, thus paying a lot of time (or space). Suggest for the verifier to only pay time $\tilde{O}(n)$ and polylog(n) space once and for all during the initialization phase, and polylog(n) space and time for every subsequent proof.

2 Proofs of Sequential Work (50 points)

Recall the PoSW from recitation 8 (Pietrzak). We described an interactive public-coin protocol (P, V) and mentioned a way to transform it into a non-interactive proof π via the Fiat-Shamir paradigm, based on a hash function H (modeled as a random oracle). In the following, assume that the time parameter T is a power of 2 (namely, $T = 2^t$).

a. **(20 points)** Given a tuple (N, x, T, y) which represents the statement $y = x^{2^T} \bmod N$, describe the structure of a valid proof π (i.e., after applying the Fiat-Shamir transform over Pietrzak's interactive PoSW).

b. **(15 points)** Describe the verification process given a statement (N, x, T, y) and a proof π.

c. **(15 points)** Compute the running time of your verification process in terms of the number of calls to H and the number of multiplications over $\mathbb{Z}_N^*$.