Homework 4

1 Problem 1 (40 points)

A point function is a function $I_x(y) = \begin{cases} 1 & \text{if } x = y \\ & \text{otherwise} \end{cases}$. Consider an obfuscator $\mathcal{O}$, which is only guaranteed to operate on the circuits I_x for $x \in \{0,1\}^n$.

(a) Show that indistinguishability obfuscation is meaningless for point functions.

As a result, we need to consider stronger notions. One possibility is VBB obfuscation. Note that the VBB impossibility required a rather complex functionality, and as such it remains possible that $\mathcal{O}$ to be a VBB obfuscator.

(b) For a function $\ell = \ell(n)$, let NP_ℓ be the set of languages that are decidable in polynomial time with $\ell(n)$ bits of non-determinism, where n is the input length. More precisely, L is in NP_ℓ if, for any input length n, there is a circuit $R : \{0,1\}^n \times \{0,1\}^{\ell(n)} \to \{0,1\}$ such that $x \in L$ if and only if there exists a $w \in \{0,1\}^{\ell(n)}$ such that $R(x,w) = 1$.

Assuming that $\mathcal{O}$ is a VBB obfuscator and is efficient, prove the following: for any super-logarithmic function $\ell \in \omega(\log n)$, there exists a language L in $NP_\ell \setminus P$. In other words, there is a problem that is efficiently decidable with ℓ bits of non-determinism but is undecidable with out non-determinism.

Observe that $NP_{\text{log}} = P$, since with logarithmic non-determinism you can do a brute force search for all possible witnesses. The result in part (a) basically shows that the *only* attack on L is a brute force search. Thus, while VBB obfuscating point functions may be possible, it has a very strong complexity-theoretic consequence.

(c) Another possibility for a security definition for point functions is something called input hiding: given $\mathcal{O}(I_x)$ for a uniformly random point x, it is computationally infeasible for find x. Show how to achieve input hiding obfuscation for point functions, assuming only injective one-way functions.

(d) More generally, an obfuscator $\mathcal{O}$ for a set of programs $\mathcal{F}$ is input hiding if, given $\mathcal{O}(f)$ for a random choice of $f \leftarrow \mathcal{F}$, it is computationally infeasible to find f.

Fix a circuit "topology" C of size s, which is a circuit of gates with 2-bit inputs and 1-bit outputs, such that the wires connected gates are specified, but the gates themselves are unspecified. Let $\mathcal{F}$ be the set of circuits obtained from C by filling in each gate with arbitrary functions from 2 bits to 1 bit.

Devise an obfuscator for $\mathcal{F}$ with *information-theoretic security*. That is, for your scheme, prove that even a computationally unbounded attacker has a negligible (in s) probability of recovering f, given an obfuscation of f sampled randomly from $\mathcal{F}$.

This suggests that circuit hiding is probably not a meaningful notion of security for general circuits.

2 Problem 2 (10 points)

Recall the security definition for a PRF. When we switch to considering quantum adversaries, typically the only thing that would change is that we allow the adversary to have a quantum computer. However, the queries the adversary makes are still classical. Call a PRF secure in this way a post-quantum PRF. It turns out that our construction and analysis of PRFs from one-way functions from the beginning of the course works also for building post-quantum PRFs. Thus we can construct post-quantum PRFs from one-way functions, provided the one-way functions are secure against quantum computers.

A stronger notion of security, however, considers an adversary that can query the PRF on a quantum superposition of inputs. That is, the adversary submits a state $\sum_{x,y} \alpha_{x,y}|x, z\rangle$, and in response gets the state $\sum_{x,y} \alpha_{x,y}|x, y \oplus H(x)\rangle$, where $H(x)$ is either the PRF or a random function. The adversary's task is still to distinguish the two cases. Call such PRFs fully-quantum PRFs. (Note that the PRF itself is still classical, just that it is being evaluated on superpositions of inputs).

Explain why the proof of security for constructing PRFs from PRGs that we saw in class breaks down when trying to prove that the construction is a fully-quantum PRF.

3 Problem 3 (25 points)

Let $f : [N] \to \{0, 1\}$ be a function. Recall that Grover's algorithm lets you find a random solution x to $f(x) = 1$, making only $O(\sqrt{N/r})$ calls to f, where r is the number of solutions. The number of solutions r need not be known.

Suppose now your goal is to find *all* possible solutions. The naive approach is to run Grover's algorithm as above roughly until you hit every solution. This becomes an instance of the coupon collectors problem. This approach will require running Grover as above approximately $r \log r$ times, giving a total time of $O(\sqrt{(Nr)} \log r)$ time.

Show how to remove the extra $\log r$ factor, obtaining an algorithm that runs in time $O(\sqrt{Nr})$, and still finds all solutions. You can assume r is known.

Hint: You will need to run Grover's algorithm on functions other than f.

4 Problem 4 (25 points)

Recall the quantum money scheme shown in class. Consider the case of a single banknote. The secret serial number for the note consists of two bit strings $b, c \in \{0,1\}^\lambda$. The banknote consists of λ qubits, where the ith qubit is in the state $|\psi_{b_i,c_i}\rangle$, where:

- $|\psi_{0,c}\rangle = |c\rangle$
- $|\psi_{1,c}\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}(-1)^c|1\rangle$

Suppose the bank also offers a verification oracle for the banknote. That is, the bank, on input a quantum money state $|\phi_1\rangle \dots |\phi_\lambda\rangle$, measures $|\phi_i\rangle$ in the basis B_{b_i}, where $B_b = \{|\psi_{b,0}\rangle, |\psi_{b,1}\rangle\}$. It obtains the output c'_i. If c', the bitstring consisting of all c_i, is identical to c, then the bank accepts. Otherwise, it rejects. Additionally, in either case, it returns whatever is left of the banknote after the measurement.

(a) Show that given such an oracle, and a single valid banknote, it is possible in polynomial time to forge new banknotes in polynomial time (with high probability).

(b) Suggest a fix to the attack in part (a). You do not need to prove the security of your fix, but must provide an informal argument why it blocks the attack from (a)