Homework 1

1 Problem 1 (20 points)

Let \( x \in \{0, 1\}^\lambda \), and let \( H : \{0, 1\}^\lambda \to \{0, 1\} \) be a function such that \( H(r) = \langle x, r \rangle \) for at least a fraction \( p \) its inputs \( r \). Here, \( \langle x, r \rangle \) means the inner product mod 2 of \( x \) and \( r \): \( \langle x, r \rangle = \sum_{i=1}^{\lambda} x_i r_i \mod 2 \).

In class, we showed that if \( p \geq \frac{3}{4} + \epsilon \) for a non-negligible \( \epsilon \), then it is possible to determine \( x \) efficiently, given only polynomially-many queries to \( H \). Here, you will show that this is essentially tight.

(a) Construct two inputs \( x_0 \neq x_1 \) and a function \( H \) such that \( H(r) = \langle x_0, r \rangle \) for at least \( \frac{3}{4} \) of its inputs, and at the same time \( H(r) = \langle x_1, r \rangle \) for at least \( \frac{3}{4} \) of its inputs. Note that the two sets of inputs may be different.

This is why, when moving to the regime where \( p = \frac{1}{2} + \epsilon \), we could no longer give an algorithm that outputted a single \( x \). Instead, we had to output multiple \( x \) values, one of which was the right answer.

(b) Generalize the above construction to more inputs. For any integer \( n \), construct \( n \) distinct inputs \( x_0, \ldots, x_{n-1} \) and a function \( H \) such that \( H(r) = \langle x_i, r \rangle \) for at least \( p \) fraction of inputs simultaneously for all \( i \), where \( p = \frac{1}{2} + \frac{1}{2^n} \). Here, you may assume \( n \) is a power of 2.

2 Problem 2 (20 points)

In class, we tried to build a signature scheme from any one-way function. However, we ran into a roadblock, where we needed a one-time signature scheme whose message space was much larger than its public key. Here, we will use hashing to solve the problem.

Definition 1 A function \( H : \{0, 1\}^\lambda \to \{0, 1\}^{n(\lambda)} \), where \( n(\lambda) < \lambda \), is collision resistant if, for all PPT adversaries \( A \), there exists a negligible \( \epsilon \) such that

\[
\Pr[x_0 \neq x_1 \land H(x_0) = H(x_1) : (x_0, x_1) \leftarrow A(1^\lambda)] < \epsilon(\lambda)
\]
Note that if \( n(\lambda) < \lambda \), collisions \((x_0 \neq x_1 \text{ such that } H(x_0) = H(x_1))\) exist in abundance. Yet collision resistance means it is computationally infeasible to actually find such a collision.

Let \((\text{Gen}, \text{Sign}, \text{Ver})\) be a one-time signature scheme with public keys of length \( p(\lambda) \) and messages of length \( n(\lambda) \), where \( n(\lambda) \) may be smaller than \( p(\lambda) \). Let \( H : \{0, 1\}^{m(\lambda)} \rightarrow \{0, 1\}^{n(\lambda)} \) be a keyed hash function, where \( m(\lambda) \) is much larger than \( p(\lambda) \). Define \((\text{Gen}', \text{Sign}', \text{Ver}')\) as the following signature scheme for messages of length \( m(\lambda) \):

- \( \text{Gen}'(1^\lambda) = \text{Gen}(1^\lambda) \).
- \( \text{Sign}'(sk', M) = \text{Sign}(sk, H(M)) \). That is, first hash the message with \( H \), and then sign using \( \text{Sign} \).
- \( \text{Ver}'(pk', M, \sigma) = \text{Ver}(pk, H(M), \sigma) \).

(a) Show that, if \( H \) is collision resistant and \((\text{Gen}, \text{Ver}, \text{Sign})\) is one-time EUF-CMA secure, then so is \((\text{Gen}', \text{Ver}', \text{Sign}')\).

(b) Show that the collision resistance of \( H \) is also necessary for security. That is, if \( H \) is not collision resistant (but still compressing), then \((\text{Gen}', \text{Sign}', \text{Ver}')\) cannot possibly be a secure one-time signature scheme.

Collision resistant hash functions are widely believed to exist, and there are many constructions based on number theory. However, it is also widely believed that a generic one-way function is not sufficient to build a collision resistant hash function. Therefore, we are still short of our goal of constructing signatures from arbitrary one-way functions. Fortunately, a slightly weaker notion of collision resistant hashing functions, called universal one-way hash function (UOWHF), is possible from one-way functions, and is sufficient to build signature schemes, albeit with a slight tweak to the construction above.

3 Problem 3 (30 points)

Here, you will extend the Goldreich-Levin theorem to multiple hardcore bits.

Let \( F : \{0, 1\}^\lambda \rightarrow \{0, 1\}^{n(\lambda)} \) be a one-way function. Let \( F' : \{0, 1\}^{k\lambda + \lambda} \rightarrow \{0, 1\}^{k\lambda + n(\lambda)} \) be the function

\[
F'(r_1, \ldots, r_k, x) = (r_1, \ldots, r_k, F(x))
\]

Assume \( k \) is logarithmic in \( \lambda \). Consider the functions \( h_i(r_1, \ldots, r_k, x) = \langle r_i, x \rangle \). Show that \( h_1, \ldots, h_k \) are all simultaneously hardcore bits for \( F' \). This means that for any
PPT adversary $A$, there exists a negligible $\epsilon$ such that

$$\left| \Pr[1 \leftarrow A(F'(x'), h_1(x'), \ldots, h_k(x')) : x' \leftarrow \{0, 1\}^{k\lambda + \lambda}] - \Pr[1 \leftarrow A(F'(x'), b_1, \ldots, b_k) : x' \leftarrow \{0, 1\}^{k\lambda + \lambda}, b_1, \ldots, b_k \leftarrow \{0, 1\}^\lambda] \right| < \epsilon(\lambda)$$

To prove this, you can use the basic Goldreich-Levin theorem as a black box (but perhaps for a slightly modified one-way function); you do not need to reprove GL from scratch in this more general setting.

4 Problem 4 (30 points)

A random self reduction is a way to re-randomize an instance of a problem. Here, you will explore some applications of such random self-reductions.

Let $G : \{0, 1\}^\lambda \rightarrow \{0, 1\}^{2\lambda}$ be a length-doubling PRG. Let $D_0$ be the distribution $G(x)$ for a random $x$. Let $D_1$ be the uniform distribution over $\{0, 1\}^{2\lambda}$.

We will say that $G$ has a perfect random self reduction is there is a PPT $\text{ReRand} : \{0, 1\}^{2\lambda} \rightarrow \{0, 1\}^{2\lambda}$ such that the following is true:

- For any fixed $y \in \{0, 1\}^{2\lambda}$ in the image of $G$, $\text{ReRand}(y)$ samples from $D_0$.
- For any fixed $y \in \{0, 1\}^{2\lambda}$ not in the image of $G$, $\text{ReRand}(y)$ samples from $D_1$.

A random self reduction means that a random instance is as hard as the hardest instance. Indeed, given any supposedly hard instance $y$, we can apply the random self reduction to get a random instance, and solving the random instance lets us solve $y$. Note that such $\text{ReRand}$ may exist without being able to tell whether $y$ is in the image of $G$ or not (which would violate PRG security). We will assume we have a $G$ that is both a secure PRG and admits a perfect random self reduction. We now consider a couple applications.

(a) Suppose a PPT adversary $A$ can run in time $T$ and break $G$ with advantage $\epsilon$. Construct an adversary $B$ running in time $\text{poly}(T, 1/\epsilon)$ which can break $G$ with advantage $99/100$. In other words, a random self reduction lets you boost the probability of distinguishing.

(b) In class, our construction of a PRF from a PRG incurred a “loss” of $nq$, where $n$ is the number of input bits and $q$ is the number of queries. In other words, a PRF adversary with advantage $\epsilon$ is turned into a PRG adversary with advantage $\epsilon/nq$.

In practice, this “loss” is important. If a different construction had a loss $n$ or even 1, then the PRG only needs to be secure against attacks with higher
success probability $\epsilon/n$ or even $\epsilon$, meaning the security parameter can be set lower. This in turn improves the efficiency of the protocol.

If $G$ has a perfect random self reduction, show how the loss in the reduction for the PRF we saw in class can be improved to just $n$. That is, starting with a PRF adversary with advantage $\epsilon$, derive an adversary for $G$ with advantage at least $\epsilon/n$.

(c) Unfortunately, random self reductions seem unlikely to exist for general PRGs. As evidence, we will show that breaking $G$ with a random self reduction is very close to lying in the complexity class $NP \cap coNP$. Thus, the existence of a re-randomizeable PRG requires hardness in $NP \cap coNP$. It is believed that one-way functions can exist without requiring such hardness (though hard problems in $NP \cap coNP$ are widely believed to exist).

To make our lives easier, assume that it is possible, for any security parameter $\lambda$, to deterministically compute some $y$ that is not in the range of $G$, in time polynomial in $\lambda$. Call this “Assumption 1”. Note that it is possible to sample $y$ not in the image of $G$ by simply sampling a random string in $\{0,1\}^{2\lambda}$; Assumption 1 requires that it is possible to deterministically generate such a $y$.

Then, assuming $G$ has a perfect random self reduction, show the following:

(i) There is a polynomial $p_1(\lambda)$ and a polynomial-time deterministic algorithm $V_1(y, w)$ that takes $y \in \{0,1\}^{2\lambda}$ and $w \in \{0,1\}^{p_1(\lambda)}$ and outputs a single bit, such that $y$ is in the image of $G$ if and only if there exists a $w$ such that $V_1(y, w) = 1$. This shows that breaking $G$ is in $NP$.

(ii) There is another polynomial $p_2(\lambda)$ and polynomial-time deterministic algorithm $V_2(y, w)$ that takes $y \in \{0,1\}^{2\lambda}$ and $w \in \{0,1\}^{p_2(\lambda)}$ and outputs a single bit, such that $y$ is in the image of $G$ if and only if there does not exist a $w$ such that $V_2(y, w) = 1$. This shows that breaking $G$ is in $coNP$.

For a hint, note that $\text{ReRand}$ can be made deterministic by explicitly feeding in the random coins: $\text{ReRand}(y) = \text{ReRand}(y; r)$ for random coins $R$ from some set $\{0,1\}^{p(\lambda)}$. You will use the deterministic version of $\text{ReRand}$ in your constructions of $V_1, V_2$.

Thus a re-randomizeable PRF satisfying Assumption 1 requires there to be hard problems in $NP \cap coNP$. We can eliminate Assumption 1 by relaxing $NP$ and $coNP$ to randomized variants called $AM$ and $coAM$. 