1. CNFs and DNFs. Recall that a Boolean formula is said to be in 3-CNF form if it is the conjunction of clauses, with each clause being the disjunction of at most 3 literals. A Boolean formula is said to be in 3-DNF form if it is the disjunction of terms, with each term being the conjunction of at most 3 literals.

Describe a polynomial-time computable function that is given as input a fan-in two (\(\land, \lor, \neg\))-circuit \(C(x_1, x_2, \ldots, x_n)\), and produces a 3-CNF Boolean formula \(\phi\) on variables \(x_1, x_2, \ldots, x_n\) and additional variables \(z_1, z_2, \ldots, z_m\) for which for every setting of the \(x\) variables

\[
\exists z_1, z_2, \ldots, z_m \ \phi(x_1, x_2, \ldots, x_n, z_1, z_2, \ldots, z_m) = 1 \iff C(x_1, x_2, \ldots, x_n) = 1.
\]

Also, describe a polynomial-time computable function that is given as input a fan-in two (\(\land, \lor, \neg\))-circuit \(C(x_1, x_2, \ldots, x_n)\), and produces a 3-DNF Boolean formula \(\phi\) on variables \(x_1, x_2, \ldots, x_n\) and additional variables \(z_1, z_2, \ldots, z_m\) for which for every setting of the \(x\) variables

\[
\forall z_1, z_2, \ldots, z_m \ \phi(x_1, x_2, \ldots, x_n, z_1, z_2, \ldots, z_m) = 1 \iff C(x_1, x_2, \ldots, x_n) = 1.
\]

Hint: identify the \(z\) variables with the gates of \(C\).

2. Toda’s Theorem (Part I). This problem concerns the class \(\oplus P\), which you will show to be quite powerful. A uniform way to define the classes \(NP, BPP\) and the new class \(\oplus P\) is in terms of polynomial-time nondeterministic Turing Machines that are standardized so that they make the same number of binary nondeterministic choices on each computation path. Specifically, a language \(L\) is in:

- **NP** iff there is such a polynomial-time nondeterministic Turing Machine \(M\) for which \(x \in L\) implies that at least one of the computation paths accepts, and \(x \notin L\) implies that no computation paths accept.

- **BPP** iff there is such a polynomial-time nondeterministic Turing Machine \(M\) for which \(x \in L\) implies that at least \(2/3\) of the computation paths accept, and \(x \notin L\) implies that at most \(1/3\) of the computation paths accept.
• $\oplus P$ iff there is such a polynomial-time nondeterministic Turing Machine $M$ for which $x \in L$ implies that an odd number of the computation paths accept, and $x \notin L$ implies that an even number of the computation paths accept.

Below we will also be discussing the classes $\text{NP}^A, \text{BPP}^A$ and $(\oplus P)^A$. These are the classes obtained by replacing the polynomial-time nondeterministic Turing Machine $M$ in the definitions above with a polynomial-time nondeterministic oracle Turing Machine $M$ that is equipped with language $A$ as its oracle. As usual, if we write $C$ instead of $A$ in the exponent, for some complexity class $C$, we mean that any language $A \in C$ is permitted as the oracle. If $C$ has a complete language (as $\text{NP}^A$ and $(\oplus P)^A$ do, for any oracle $A$), then by using that language as the oracle we can solve any instance of a problem in $C$ with a single call to this specific oracle.

(a) The following is a more general restatement of the Valiant-Vazirani Theorem from Lecture 8.

**Lemma 5.1** There is a randomized procedure that receives as input an integer $n$, runs in $\text{poly}(n)$ time, and outputs a $\text{poly}(n)$-size circuit $C$ with the following property: for each subset $T \subseteq \{0,1\}^n$, if $|T| > 0$, then with probability at least $1/(8n)$ over the randomness of the procedure,

$$|\{x : x \in T \text{ and } C(x) = 1\}| = 1.$$ 

Using this lemma, prove that for every oracle $A$, $\text{NP}^A \subseteq \text{BPP}^{(\oplus P)^A}$. It may be helpful to think about the non-relativized statement first.

(b) Prove that for every oracle $A$,

$$\text{NP}^A \subseteq \text{BPP}^A \Rightarrow \text{NP}(\text{NP}(\text{NP}(\ldots (\text{NP}^A)\ldots)))) \subseteq \text{BPP}^A$$

in which the tower of $\text{NP}$ classes has height $i$, for $i = 1, 2, 3, \ldots$.

Hint: first, figure out how to use error-reduction to argue that $\text{NP}^{\text{BPP}} \subseteq \text{BPP}^\text{NP}$ and $\text{BPP}^{\text{NP}} \subseteq \text{BPP}$. If you are stuck, you can take the relativized versions of these statements as given, for partial credit.

(c) Prove that $\text{co-(\oplus P)} \subseteq \oplus P$.

(d) Prove that $(\oplus P)^{\oplus P} \subseteq \oplus P$.

Hint: Use the fact that odd $\times$ odd = odd. For each language $L$ in $(\oplus P)^{\oplus P}$ with associated nondeterministic oracle TM $M$, you will be designing a nondeterministic TM $M'$. A helpful strategy is for $M'$ to begin by nondeterministically guessing a transcript of $M$ on input $x$, which contains a sequence of nondeterministic choices made by $M$, together with a sequence of queries made to the oracle, and a sequence of yes/no answers. Many of these transcripts will be inaccurate in the sense that they don’t agree with the functioning of $M$ on the specified computation path, with the actual oracle in $\oplus P$ answering queries. Nevertheless, some nondeterministic guesses produce “correct” transcripts...

(e) Prove that $\text{PH} \subseteq \text{BPP}^{\oplus P}$.
3. Approximate counting and sampling with an \textbf{NP} oracle. For every \( n, k \) (positive integers, with \( k \leq n \)), there is a multiset \( \mathcal{H}_{n,k} \) of functions \( h : \{0,1\}^n \to \{0,1\}^k \), called an “\( n \)-wise independent hash family”. This multiset comes equipped with a probabilistic procedure that runs in time \( \text{poly}(n) \) and outputs a uniformly chosen \( h \) from \( \mathcal{H}_{n,k} \), in the form of a circuit for \( h \) of size \( \text{poly}(n) \). These functions behave like random functions from \( n \) bits to \( k \) bits in the following sense:

\textbf{Lemma 5.2} For every set \( S \subseteq \{0,1\}^n \) and every \( y \in \{0,1\}^k \):

\[
\Pr_{h \in \mathcal{H}_{n,k}} \left[ \left| \{x : x \in S \land h(x) = y\} \right| > 2 \cdot \frac{|S|}{2^k} \right] \leq 2^{-2n},
\]

provided that \( 2^k \leq 4|S|/n^4 \).

Note that for a random function \( h \) from \( n \) bits to \( k \) bits, the expected size of

\[
\{x : x \in S \land h(x) = y\}
\]

is \( |S|/2^k \); the lemma says that with high probability, the same set with respect to a function \( h \) drawn uniformly from \( \mathcal{H}_{n,k} \) does not exceed this expected size by more than a factor of two.

In the problems below, the input is a set \( S \subseteq \{0,1\}^n \) given \textit{implicitly} by a circuit \( C : \{0,1\}^n \to \{0,1\} \) for which \( C(x) = 1 \) iff \( x \in S \). You can think of \( C \) as an instance of \textsc{circuit sat}, and then the questions below concern the problems of estimating the number of satisfying assignment, and sampling from them, respectively.

(a) Describe a probabilistic polynomial-time procedure, with access to an \textbf{NP} oracle, that with probability at least 7/8 outputs an integer \( k \) for which \( 2^k < \frac{|S|}{n^2} \leq 2^{k+2} \). Hint: argue that deciding whether an implicitly given set has size at least \( s \), for polynomially-large \( s \), is in \textbf{NP}, and then perform an experiment for each \( k = 1, 2, 3, \ldots \).

(b) Describe a probabilistic polynomial-time procedure, with access to an \textbf{NP} oracle, that outputs “fail” with probability at most 7/8 and otherwise outputs an exactly uniformly distributed element of \( S \). Hint: suppose a notebook has \( L \) lines on every page, with an enumeration of the elements of a set \( S \) are written on a subset of the lines in the notebook. Consider selecting a random page and a random line on that page, and outputting the element written on that line, or “fail” if the line is empty. What is the probability of outputting a given element of \( S \)? What is the probability of outputting “fail”?