# Approximate Model Counting: Is SAT Oracle More Powerful Than NP Oracle?

**Diptarka Chakraborty**
National University of Singapore, Singapore

**Sourav Chakraborty**
Indian Statistical Institute, Kolkata, India

**Gunjan Kumar**
National University of Singapore, Singapore

**Kuldeep S. Meel**
National University of Singapore, Singapore

## Abstract

Given a Boolean formula $\phi$ over $n$ variables, the problem of model counting is to compute the number of solutions of $\phi$. Model counting is a fundamental problem in computer science with wide-ranging applications in domains such as quantified information leakage, probabilistic reasoning, network reliability, neural network verification, and more. Owing to the #P-hardness of the problems, Stockmeyer initiated the study of the complexity of approximate counting. Stockmeyer showed that $\log n$ calls to an NP oracle are necessary and sufficient to achieve $(\varepsilon, \delta)$ guarantees. The hashing-based framework proposed by Stockmeyer has been very influential in designing practical counters over the past decade, wherein the SAT solver substitutes the NP oracle calls in practice. It is well known that an NP oracle does not fully capture the behavior of SAT solvers, as SAT solvers are also designed to provide satisfying assignments when a formula is satisfiable, without additional overhead. Accordingly, the notion of SAT oracle has been proposed to capture the behavior of SAT solver wherein given a Boolean formula, an SAT oracle returns a satisfying assignment if the formula is satisfiable or returns unsatisfiable otherwise. Since the practical state-of-the-art approximate counting techniques use SAT solvers, a natural question is whether an SAT oracle is more powerful than an NP oracle in the context of approximate model counting.

The primary contribution of this work is to study the relative power of the NP oracle and SAT oracle in the context of approximate model counting. The previous techniques proposed in the context of an NP oracle are weak to provide strong bounds in the context of SAT oracle since, in contrast to an NP oracle that provides only one bit of information, a SAT oracle can provide $n$ bits of information. We therefore develop a new methodology to achieve the main result: a SAT oracle is no more powerful than an NP oracle in the context of approximate model counting.

**2012 ACM Subject Classification** Theory of computation → Oracles and decision trees

**Keywords and phrases** Model counting, Approximation, Satisfiability, NP oracle, SAT oracle

**Digital Object Identifier** 10.4230/LIPIcs.ICALP.2023.123

**Category** Track B: Automata, Logic, Semantics, and Theory of Programming

**Funding** *Diptarka Chakraborty*: Supported in part by an MoE AcRF Tier 2 grant (MOE-T2EP20221-0009) and Google South & South-East Asia Research Award.
*Gunjan Kumar*: Supported in part by National Research Foundation Singapore under its NRF Fellowship Programme[NRF-NRFFAI1-2019-0004 ].
*Kuldeep S. Meel*: Supported in part by National Research Foundation Singapore under its NRF Fellowship Programme[NRF-NRFFAI1-2019-0004 ] and Campus for Research Excellence and Technological Enterprise (CREATE) programme, Ministry of Education Singapore Tier 2 grant MOE-T2EP20121-0011, and Ministry of Education Singapore Tier 1 Grant [R-252-000-B59-114].







## 1 Introduction

Let $\phi$ be a Boolean formula over $n$ propositional variables. An assignment $s \in \{T, F\}^n$ is called a *satisfying assignment* if it makes $\phi$ evaluate to true. Let $\mathsf{sol}(\phi)$ denote the set of all satisfying assignments. The model counting problem is to compute $|\mathsf{sol}(\phi)|$ for a given $\phi$. It is a fundamental problem in computer science and has numerous applications across different fields such as quantified information leakage, probabilistic reasoning, network reliability, neural network verification, and the like [12, 13, 17, 9, 8, 1]. The seminal work of Valiant [17] showed that the problem of model counting is #P-complete, and consequently, one is often interested in approximate variants of the problem. In this paper, we consider the following problem:

**Approximate Model Counting**

**Input** A formula $\phi$, tolerance parameter $\varepsilon > 0$, and confidence parameter $\delta \in (0, 1)$.
**Output** Compute an estimate $\mathsf{Est}$ such that

$$\Pr\left[\frac{|\mathsf{sol}(\phi)|}{1+\epsilon} \leq \mathsf{Est} \leq (1+\epsilon)|\mathsf{sol}(\phi)|\right] \geq 1-\delta.$$

Stockmeyer [16] initiated the study of the complexity of approximate model counting. Stockmeyer's seminal paper made two foundational contributions: the first contribution was to define the query model that could capture possible natural algorithms yet amenable enough to theoretical tools to allow non-trivial insight. To this end, Stockmeyer proposed the query model wherein one can construct an arbitrary set $S$ and query an NP oracle to determine if $|\mathsf{sol}(\phi) \cap S| \geq 1$. Stockmeyer showed that under the above-mentioned query model, $\log n$ calls to an NP oracle are necessary and sufficient (for a fixed $\varepsilon$ and $\delta$). Furthermore, Stockmeyer introduced a hashing-based algorithmic procedure to achieve the desired upper bound that makes $O(\log n)$ calls to NP-oracle. The lack of availability of powerful reasoning systems for problems in NP dissuaded the development of algorithmic frameworks based on Stockmeyer's hashing-based framework until the early 2000s [10].

Motivated by the availability of powerful SAT solvers, there has been a renaissance in the development of hashing-based algorithmic frameworks for model counting, wherein a call to an NP oracle is handled by an SAT solver in practice. The current state-of-the-art approximate model counter, ApproxMC [4], relies on the hashing-based framework and is able to routinely handle problems involving hundreds of thousands of variables. The past decade has witnessed a sustained interest in further enhancing the scalability of these approximate model counters. It is perhaps worth highlighting that Stockmeyer's query model captures queries by ApproxMC.

While the current state-of-the-art approximate model counters rely on the hashing-based framework, they differ significantly from Stockmeyer's algorithm for approximate model counting. The departures from Stockmeyer's algorithm have been deliberate and have often been crucial to attaining scalability. In particular, ApproxMC crucially exploits the underlying SAT solver's ability to return a satisfying assignment to attain scalability. In this context, it is worth highlighting that, unlike an NP oracle that only returns the answer Yes or No for a given Boolean formula, all the known SAT solvers are capable of returning a satisfying assignment if the formula is satisfiable without incurring any additional overhead. Observe that one would need $n$ calls to an NP oracle to determine a satisfying assignment. From this viewpoint, an NP oracle does not fully capture the behavior of an SAT solver, and one needs a different notion to model the behavior of SAT solver.

Delannoy and Meel [7] sought to bridge the gap between theory and practice by proposing the notion of a SAT oracle. Formally, a SAT oracle takes in a Boolean formula $\phi$ as input and returns a satisfying assignment $s \in \mathsf{sol}(\phi)$ if $\phi$ is satisfiable and $\perp$, otherwise. It is worth highlighting that we may need $n$ calls to an NP oracle to simulate a query to a SAT oracle, and therefore, it is conceivable for an algorithm to make $O(\log n)$ calls to a SAT oracle but $O(n \log n)$ calls to an NP oracle. Delannoy and Meel showcased precisely such behavior in the context of *almost-uniform generation*. Their proposed algorithm, UniSamp makes $O(\log n)$ calls to a SAT oracle and would require $O(n \log n)$ calls to an NP oracle if one were to replace a SAT oracle with an NP oracle. At the same time, it is not necessary that there would be a gap of $n$ calls for every algorithm: simply consider the problem of determining whether a formula is satisfiable or not. Only one call to an NP oracle (and similarly to a SAT oracle) suffices.

Furthermore, the notion of the SAT oracle has the potential to be a powerful tool to explain the behavior of algorithms, as highlighted by Delannoy and Meel. Given access to an NP oracle, the sampling algorithm due to Jerrum, Valiant, and Vazirani [11] (referred to as JVV algorithm) makes $O(n^2 \log n)$ calls to an NP oracle as well as a SAT oracle, i.e., there are no savings from the availability of a SAT oracle. On the other hand, the algorithm, UniSamp makes $O(\log n)$ and $O(n \log n)$ calls to SAT and an NP oracle respectively. Therefore, the NP oracle model would indicate that one should expect the performance gap between JVV and UniSamp to be linear, while the SAT oracle model indicates an exponential gap. The practical implementations of JVV and UniSamp indeed indicate the performance gap between them to be exponential rather than linear. Therefore, analyzing problems under the SAT oracle model has the promise to have wide-ranging consequences.

In this paper, we analyze the complexity of the problem of approximate model counting given access to a SAT oracle. Our study is motivated by two observations:

**O1** The modern state-of-the-art hashing-based techniques differ significantly from Stockmeyer's algorithmic procedure and, in particular, exploit the availability of SAT solvers. Yet, they make $O(\log n)$ calls to a SAT oracle, which coincides with the number of NP oracle calls in Stockmeyer's algorithmic procedure.

**O2** Stockmeyer provided a matching lower bound of $\Omega(\log n)$ on the number of NP calls, which follows from the simple observation that for a fixed $\varepsilon$, there are $\Theta(n)$ possible outputs that an algorithm can return. Since every NP call returns an answer, Yes or No, the trace of an algorithm can be viewed as a binary tree such that every leaf represents a possible output value. Therefore, the height of the tree (i.e., the number of NP calls) must be $\Omega(\log n)$. Since a SAT oracle returns a satisfying assignment (i.e., provides $n$ bits of information), the trace of the algorithm is no longer a binary tree, and therefore, Stockmeyer's analysis does not extend to the case of SAT oracles for approximate model counting.

To summarize, the best-known upper bound for SAT oracle calls for approximate model counting is $O(\log n)$, which matches the upper bound for NP oracle calls. However, the technique developed in the context of achieving a lower bound for NP oracle calls does not apply to the case of SAT oracle. Therefore, one wonders whether there exist algorithms with a lower number of SAT oracle calls. In other words, are SAT oracles more powerful than NP oracles for the problem of approximate model counting?

The primary contribution of this work is to resolve the above challenge. In contrast to the problem of uniform sampling, we reach a starkly different conclusion: SAT oracles are no more powerful than NP oracles in the context of approximate model counting. Formally, we prove the following theorem:

▶ **Theorem 1.1.** *For any $\epsilon, \delta \in (0,1)$, given a formula $\phi$, computation of $(\varepsilon, \delta)$-approximation of $|\mathsf{sol}(\phi)|$ requires $\tilde{\Omega}(\log n)$[1] queries to a SAT oracle.*

The establishment of the above theorem turned out to be highly challenging as the existing approaches in the context of NP oracles are not applicable to the SAT oracles. We provide an overview of our approach below.

## 1.1 Technical Overview

In order to provide the lower bound on the number of queries required by the SAT oracle, we work with a stronger SAT oracle model. In particular, an answer from a (standard) SAT oracle does not provide any extra guarantee/information other than that the returned assignment is a satisfying assignment of the queried formula. Our lower bound works even if we consider that the returned satisfying assignment is chosen randomly from the set of satisfying assignments. More specifically, we consider a stronger model, namely SAT-Sample *oracle*, which returns a uniformly chosen solution of a queried formula $\phi$ whenever the formula is satisfiable. It is worth remarking that while a SAT oracle can be simulated by only $n$ queries to an NP oracle, the best-known technique to simulate SAT-Sample makes $O(n^2 \log n)$ queries to an NP oracle [2, 7]. We prove the following theorem which implies Theorem 1.1.

▶ **Theorem 1.2.** *For any $\epsilon < 1/2$ and any $\delta \leq 1/6$, given a formula $\phi$, computation of $(\varepsilon, \delta)$-approximation of $|\mathsf{sol}(\phi)|$ requires $\tilde{\Omega}(\log n)$ queries to a* SAT-Sample *oracle.*

Although we consider $\epsilon < 1/2$ and $\delta \leq 1/6$ in the above theorem and provide the proof accordingly, our proof works even for any constant $\epsilon, \delta \in (0,1)$. Another thing to remark is that in our proof, we allow even exponential (in the size of the original formula) size formula to be queried in the SAT-Sample oracle, making our result stronger than what is claimed in the above theorem.

Let us assume that Alg is an algorithm that $(\epsilon, \delta)$-approximates $|\mathsf{sol}(\phi)|$ for any given input $\phi$ (on $n$ variables) by making $q$ queries to a SAT-Sample oracle. We will refer to such an algorithm as a SAT-Sample counter. We would like to prove a lower bound on $q$.

The main technical difficulty in proving our lower bound results comes from the enormous power of a SAT-Sample oracle compared to an NP oracle. An NP oracle can only provide a YES or NO answer, restricting the number of possible answers (from the NP oracle) to $2^q$ for a $q$-query counter with an NP oracle. On the other hand, since a SAT-Sample oracle returns a (random) satisfying assignment (if a satisfying assignment exists), the number of possible answers can be $2^{nq}$. Further, any counter can be adaptive – it can choose the next query adaptively based on the previous queries made and their corresponding answers. In general, proving a non-trivial (tight) lower bound for any adaptive algorithm turns out to be one of the notorious challenges, and the difficulty in proving such a lower bound arises in other domains like data structure lower bound, property testing, etc. One of the natural ways to prove any lower bound is to use the information-theoretic technique. However, one of the main challenges in applying such techniques in the adaptive setting is that conditional mutual information terms often involve complicated conditional distributions that are difficult to analyze.

To start with, we argue that we can assume that the SAT-Sample counter is "semi-oblivious" in nature. The number of satisfying assignments of a formula does not change by any permutation of the elements in $\{T, F\}^n$, and the SAT-Sample counter can only get

[1] The tilde hides a factor of $\log \log n$.

elements of $\mathsf{sol}(\phi)$ by querying the SAT-Sample oracle. So we argue that the only useful information of the $i^{th}$ query set (that is, the set of satisfying assignments of the formula that is given to the SAT-Sample oracle) is the size of its intersection with the previous $(i-1)$ query sets and their corresponding answers. We formalize it in Section 3.1.

We next use Yao's minimax principle to prove a lower bound on the number of queries to a SAT-Sample oracle made by a deterministic "Semi-oblivious counter" when the input formula $\phi$ is drawn from a "hard" distribution.

For the hard distribution, we construct $O(n^{3/4})$ formulas $\phi_\ell$ for each value of $\ell$ in the set $\{\lfloor n^{1/4} \rfloor, \lfloor n^{1/4} \rfloor + 1, \ldots, \lceil n^{3/4} \rceil\}$. The formulas $\phi_\ell$ are chosen in such a way that $|\mathsf{sol}(\phi_\ell)| \approx 2|\mathsf{sol}(\phi_{\ell+1})|$ thereby approximately counting the number of satisfying assignments (upto a multiplicative $(1+\epsilon)$-factor for small constant $\epsilon$) reduces to the problem of determining the value of $\ell$. The hard distribution is obtained by picking an $\ell$ uniformly at random from the set $\{\lfloor n^{1/4} \rfloor, \lfloor n^{1/4} \rfloor + 1, \ldots, \lceil n^{3/4} \rceil\}$ and using the corresponding formula $\phi_\ell$.

Finally, we show the lower bound using information theory. At a high level, we show that the information gained about $\ell$ by the knowledge of obtained samples is small unless we make $\tilde{\Omega}(\log n)$ oracle calls (Lemma 9). Then we turn to Fano's Inequality (Theorem 3) which links the error probability of a counter to the total information gain. Showing that the information gained by samples is small boils down to showing that the KL-divergence of the conditional distribution over the samples is small for all formulas $\phi_\ell$ (shown in the proof of the third part of Lemma 9). The difficulty in showing the above bound comes from the fact that the samples are adaptive and may not always be concentrated around the expectation. To overcome the above challenge, we first define an indicator random variable $Y_i$ to denote whether, at the $i^{th}$ query, the concentration holds (see the definition in Equation 10). Then we split it into cases: In the first case, we argue for the situation when concentration may not hold at some step of the algorithm (if $Y_i = 1$ for some $i \in [q]$). The second case is when concentration holds (if $Y_i = 0$ for all $i \in [q]$). We believe that the technique developed in this paper can be a general tool to show sampling lower bounds in a number of other settings.

## 2 Notations and Preliminaries

For any integer $m$, let $[m]$ denote the set of integers $\{1, 2, \ldots, m\}$. For a formula $\phi$ over variable set $\mathsf{vars}(\phi) = \{v_1, \ldots, v_n\}$, we denote by $\mathsf{sol}(\phi)$ the set of satisfying assignments of $\phi$. If $\phi$ is not satisfiable then $\mathsf{sol}(\phi) = \emptyset$. We can interpret $\mathsf{sol}(\phi)$ as a subset of $\{T, F\}^n$. On the other hand, for any subset $A \subset \{T, F\}^n$ we denote by $\psi_A$ the formula whose set of satisfying assignments is exactly $A$; that is, $\mathsf{sol}(\psi_A) = A$.

### Oracles and query model

In our context of Boolean formulas, an *NP oracle* takes in a Boolean formula $\phi$ as input and returns Yes if $\phi$ is satisfiable (i.e., $\mathsf{sol}(\phi) \neq \emptyset$), and No, otherwise. Modern SAT solvers, besides determining whether a given formula is satisfiable or not, also return a satisfying assignment (arbitrarily) if the formula is satisfiable. This naturally motivates us to consider an oracle, namely SAT-Sample *oracle*, that takes in a Boolean formula $\phi$ as input and, if $\phi$ is satisfiable, returns a satisfying assignment uniformly at random from the set $\mathsf{sol}(\phi)$, and $\perp$, otherwise.

We rely on the query model introduced by Stockmeyer [16]: For a given $\phi$ whose model count we are interested in estimating, one can query the corresponding (NP/SAT) oracle with formulas of the form $\phi \wedge \psi_A$, where, as stated earlier, $\psi_A$ is an (arbitrary) formula whose set of solutions is $A$. We will use $\phi_A$ as a shorthand to represent $\phi \wedge \psi_A$. Throughout

this paper, we consider the above query model with query access to the SAT-Sample oracle. One call to the SAT-Sample oracle will be called a SAT-Sample query. By abuse of notation, we sometimes say "$A$ is queried" to refer to the formula $\phi_A$.

### $k$-wise independent hash functions

Let $n, m, k$ be positive integers and let $H(n, m, k)$ denote the family of *$k$-wise independent hash functions* from $\{T, F\}^n$ to $\{T, F\}^m$. For any $\alpha \in \{T, F\}^m$, and $h \in H(n, m, k)$, let $h^{-1}(\alpha)$ denote the set $\{s \in \{T, F\}^n \mid h(s) = \alpha\}$.

It is well-known (e.g., see [5]) that for any integer $n, m, k$, one can generate an explicit family of $k$-wise independent hash functions in time and space $poly(n, m, k)$. Moreover, for any $\alpha \in \{T, F\}^m$, $h^{-1}(\alpha)$ (where $h \in H(n, m, k)$) can be specified by a Boolean formula of size $poly(n, m, k)$.

### Concentration inequalities for limited independence

▶ **Lemma 1** ([15]). *If $X$ is a sum of $k$-wise independent random variables, each of which is confined to $[0, 1]$ with $\mu = \mathbb{E}[X]$ then*

1. *For any $\gamma \leq 1$ and $k \geq \gamma^2\mu$, $\Pr[|X - \mu| \geq \gamma\mu] \leq exp(-\gamma^2\mu/3)$.*
2. *For any $\gamma \geq 1$ and $k \geq \gamma\mu$, $\Pr[|X - \mu| \geq \gamma\mu] \leq exp(-\gamma\mu/3)$.*

### Basics of information theory

Let $X$ and $Y$ be two random variables over the space $\mathcal{X} \times \mathcal{Y}$. The mutual information $I(X;Y)$ between random variables $X$ and $Y$ is the reduction in the entropy of $X$ given $Y$ and hence

$$I(X;Y) = H(X) - H(X|Y) \leq H(X) \tag{1}$$

where $H(X) = -\sum_{x \in \mathcal{X}} \Pr[X = x] \log \Pr[X = x]$ is the Shannon entropy of $X$ and $H(X|Y)$ is the conditional entropy of $X$ given $Y$.

The *Kullback–Leibler divergence* or simply *KL divergence* (also called relative entropy) between two discrete probability distributions $P$ and $Q$ defined on same probability space $\mathcal{X}$ is given by :

$$KL(P||Q) := \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$$

where $p$ and $q$ are probability mass functions of $P$ and $Q$ respectively.

If the joint distribution of $X$ and $Y$ is $Q_{X,Y}$ and marginal distributions $Q_X$ and $Q_Y$ respectively, then the mutual information $I(X;Y)$ can also be equivalently defined as:

$$I(X;Y) := KL(Q_{X,Y}||Q_X \times Q_Y).$$

For three random variables $X, Y, Z$, the *conditional mutual information* $I(X;Y|Z)$ is defined as

$$I(X;Y|Z) := \mathbb{E}_Z[KL(Q_{(X,Y)|Z}||Q_{X|Z} \times Q_{Y|Z})].$$

For any three random variables $X, Y, Z$, the *chain rule* for mutual information says that

$$I(X;(Y,Z)) = I(X;Y) + I(X;Z|Y).$$

If $Z$ is a discrete random variable taking values in $\mathcal{Z}$ then we have

$$\mathbb{E}_Z[KL(Q_{(X,Y)|Z}||Q_{X|Z} \times Q_{Y|Z})] = \sum_{z\in\mathcal{Z}} Q_Z(z) \cdot KL(Q_{(X,Y)|Z=z}||Q_{X|Z=z} \times Q_{Y|Z=z})$$

$$= \sum_{z\in\mathcal{Z}} Q_Z(z) \cdot I(X;Y|Z=z).$$

▶ **Lemma 2** ([14]). *Let $P_X, P_Z, P_{Z|X}$ be the marginal distributions corresponding to a pair $(X,Z)$, where $X$ is discrete. For any auxiliary distribution $Q_Z$, we have*

$$I(X,Z) = \sum_x P_X(x) KL(P_{Z|X}(\cdot|x)||P_Z) \leq \max_x KL(P_{Z|X}(\cdot|x)||Q_Z).$$

▶ **Theorem 3** (Fano's inequality). *Consider discrete random variables $X$ and $\hat{X}$ both taking values in $\mathcal{V}$. Then*

$$\Pr[\hat{X} \neq X] \geq 1 - \frac{I(X;\hat{X}) + \log 2}{\log|\mathcal{V}|}.$$

Consider the random variables $X, Z, \hat{X}$. If the random variable $\hat{X}$ depends only on $Z$ and is conditionally independent on $X$, then we have

$$I(X;\hat{X}) \leq I(X;Z). \tag{2}$$

This inequality is known as the *data processing inequality*. For the further exposition, readers may refer to any standard textbook on information theory (e.g., [6]).

### MiniMax theorem

Yao's minimax principle [18] is a standard tool to show lower bounds on the worst-case performance of randomized algorithms. Roughly speaking, it says that to show a lower bound on the performance of a randomized algorithm $R$, it is sufficient to show a lower bound on any deterministic algorithm when the instance is randomly drawn from some distribution.

Consider a problem over a set of inputs $\mathcal{X}$. Let $\Gamma$ be some probability distribution over $\mathcal{X}$ and let $X \in \mathcal{X}$ be an input chosen as per $\Gamma$. Any randomized algorithm $R$ is essentially a probability distribution over the set of deterministic algorithms, say $\mathcal{T}$. By Yao's minimax principle,

$$\max_{X\in\mathcal{X}} \Pr[R \text{ gives wrong answer on } X] \geq \min_{T\in\mathcal{T}} \Pr_{X\sim\Gamma}[T \text{ gives wrong answer on } X].$$

## 3 Lower Bound on the number of queries to SAT-Sample oracle

In this section, we will prove Theorem 1.2, which implies Theorem 1.1. Let Alg be an adaptive randomized algorithm that given as input $\phi$ over $n$ variables $\mathsf{vars} = \{v_1, \ldots, v_n\}$ and output Est that is an $(\epsilon,\delta)$-approximation of $\mathsf{sol}(\phi)$. The only way Alg accesses the input $\phi$ is by making queries to the SAT-Sample oracle, that is, obtaining random satisfying assignments from $\mathsf{sol}(\phi_A)$, where $\phi_A = \phi \wedge \psi_A$. We will prove that Alg has to make at least $\tilde{\Omega}(\log n)$ such queries to the SAT-Sample oracle.

We will start by arguing that we can assume that the adaptive algorithm Alg has some more structure. In particular, in Section 3.1 we will argue (in the same lines as in [3]) that we can assume Alg is a *semi-oblivious counter* (Definition 4).

We use Yao's Min-max technique to argue that obtaining a lower bound on a (randomized) semi-oblivious counter is the same as obtaining a lower bound on a (deterministic) semi-oblivious counter when the input is drawn from the worst possible distribution over the set of formulas on $n$ variables. In Section 3.2 we present the "hard" distribution that would help us prove the lower bound against any deterministic semi-oblivious counter. In Section 3.2.1 we present some properties of the hard instance that would be used for the final lower bound proof.

Finally in Section 3.3 we will use an information-theoretic argument to give a lower bound on the query complexity of any deterministic semi-oblivious counter and hence prove Theorem 1.2.

#### A note on the use of auxiliary variables in the queries to the SAT-Sample oracle

One thing we observe is that our lower bound proof does not assume that in the input formula $\phi$ all the variables are influential. In other words, we can assume that $\phi$ is on $n$ variables, the actual number of variables in $\phi$ may be significantly less. All we need for our lower bound proofs to go through is that the queries to the SAT-Sample oracle made by the algorithm are to $\phi \wedge \psi_A$ where the $\psi$ is a formula over $n$ variables. And the lower bound on the query complexity that we prove (Theorem 1.2) is $\tilde{O}(\log n)$. Hence, as long as the number of variables used in the queries to the SAT-Sample oracle is at most polynomial in the actual number of variables in the input formula $\phi$, our lower bound holds.

## 3.1 Semi-oblivious counter

Suppose given a formula $\phi$ over $n$ variables, a counter Alg makes $q$ calls to the SAT-Sample oracle with queried formulas $\phi_{A_1}, \cdots, \phi_{A_q}$ respectively, where each $A_i \subseteq \{T,F\}^n$. (Recall, $\phi_{A_i} = \phi \wedge \psi_{A_i}$, where $\psi_{A_i}$ denote the formula having $\mathsf{sol}(\psi_{A_i}) = A_i$.) Note, the $i$-th SAT-Sample oracle call by the counter Alg is specified by the set $A_i$. During the $i$-th call (for $1 \leq i \leq q$), suppose the counter Alg receives a sample $s_i \in A_i \cup \{\perp\}$. Note that the oracle calls made by Alg can be *adaptive*, i.e., the sets $A_1, \cdots, A_q$ are not fixed in advance – the counter Alg fixes $A_i$ only after seeing the samples $s_1, \cdots, s_{i-1}$ (outcomes of all the previous oracle calls).

We now define a special type of randomized SAT-Sample counter, referred to as *semi-oblivious counter*, which at any point of time queries the SAT-Sample oracle only by looking into the configuration of the previous step. We will later argue that to prove a query lower bound for general SAT-Sample counters, it suffices to consider semi-oblivious counters. In other words, semi-oblivious counters are as "powerful" as general SAT-Sample counters.

We first provide intuition for *semi-oblivious counter*. Note that permuting the variables of any formula $\phi$ permutes the set of satisfying assignments $\mathsf{sol}(\phi)$ but $|\mathsf{sol}(\phi)|$ is unchanged. Since a SAT-Sample counter needs to determine $|\mathsf{sol}(\phi)|$ only (not $\mathsf{sol}(\phi)$), the final output by the SAT-Sample counter, in some sense, should be based only on the relations between the samples and the query sets (not on their actual values). Before providing a formal definition, let us first introduce some terminology.

Given a family of sets $\mathcal{A} = \{A_1, \cdots, A_i\}$, (where $A_i \subseteq \{T,F\}^n$), the *atoms* generated by $\mathcal{A}$, denoted by $\mathsf{At}(\mathcal{A})$, are (at most) $2^i$ distinct sets of the form $\cap_{j=1}^{i} C_j$ where $C_j \in \{A_j, \{T,F\}^n \setminus A_j\}$. For example, if $i = 2$, then $\mathsf{At}(A_1, A_2) = \{A_1 \cap A_2, A_1 \setminus A_2, A_2 \setminus A_1, (A_1 \cup A_2)^c\}$.

▶ **Definition 4** (Semi-oblivious counter). *A semi-oblivious counter is a randomized algorithm $T$ that, given any formula $\phi$, at any step $i$, works in the following three phases:*

- ***Semi-oblivious choice:*** *Let $\mathcal{A}_{i-1} = \{A_1, \cdots, A_{i-1}\}$, $S_{i-1} = \{s_1, \cdots, s_{i-1}\}$, $C_{i-1} = \{c_1, \cdots, c_{i-1}\}$ be the set of first $i-1$ query sets, the set of first $i-1$ samples obtained, the set of first $i-1$ configurations, respectively. Only based on $C_{i-1}$ (without knowing the set $S_{i-1}$), $T$ does the following:*
  - *For each $A \in \mathsf{At}(\mathcal{A}_{i-1})$, it generates an integer $k_i^A$ between 0 and $|A \setminus S_{i-1}|$. ($k_i^A$ indicates how many unseen elements from the atom $A$ of the previous query sets are to be included in the next query set.)*
  - *It chooses a set of indices $K_i \subseteq \{1, \cdots, i-1\}$. ($K_i$ specifies the index set of previous samples that are to be included in the next query set.)*
- ***Query set generation:*** *In this phase, it decides the query set $A_i$ as follows:*
  - *Let us define the* candidate unseen set family *as*

$$\mathcal{U}_i := \{U \subseteq \{T,F\}^n \setminus S_{i-1} \mid \forall A \in \mathsf{At}(\mathcal{A}_{i-1}), |U_i \cap A| = k_i^A\}.$$

    *The algorithm $T$ chooses a set $U_i$ uniformly at random from the candidate unseen set family $\mathcal{U}_{i-1}$.*
  - *Let us denote $O_i := \{s_j \mid j \in K_i\}$. The algorithm $T$ decides the query set to be $A_i = U_i \cup O_i$.*
- ***Oracle call:*** *It places a query to the* SAT-Sample *oracle with the formula $\phi_{A_i}$. Let the $i$-th* configuration *$c_i$ specify whether $s_i = \perp$, or for which $j \in K_i$, $s_i = s_j$, or for which $A \in \mathsf{At}(\mathcal{A}_{i-1})$, $s_i \in A \cap U_i$.*

*In the end (after placing $q = q(n)$* SAT-Sample *oracle calls), depending on the set of all the configurations $C_q$, $T$ outputs an estimate on the $|\mathsf{sol}(\phi)|$.*

From now on, for brevity, we use $\mathsf{At}(U_i)$ to denote the set $\{U_i \cap A \mid A \in \mathsf{At}(\mathcal{A}_{i-1})\}$. Next, we show that if there exists a general SAT-Sample counter, then there also exists a semi-oblivious counter. The proof is inspired by the argument used in [3] and is given in Appendix A.

▶ **Lemma 5.** *If there is an algorithm that, given any input $\phi$ on $n$ variables, outputs an $(\epsilon, \delta)$-approximation of $|\mathsf{sol}(\phi)|$ while placing at most $q = q(n)$* SAT-Sample *oracle calls, then there also exists a (randomized) semi-oblivious counter that, given input $\phi$, outputs an $(\epsilon, \delta)$-approximation of $|\mathsf{sol}(\phi)|$ while also placing at most $q$* SAT-Sample *oracle calls.*

Suppose all the internal randomness of a semi-oblivious counter is fixed. (Since in the proof of Theorem 1.1, we will first apply Yao's minimax principle, it suffices to only consider deterministic decision trees.) Then, a semi-oblivious counter $T$ can be fully described by a decision tree $R$ where the path from the root to any node $v$ at depth $i$ (more precisely, the edges of this path) corresponds to the configuration of the first $i-1$ samples. Note that fixing the configurations of the samples till $i-1$ queries (and the internal randomness) fixes the size of an atom $A \in At(A_1, \cdots, A_i)$ (and hence of each $A_j$ for $j \leq i$). Formally,

**(i)** A path (from root) to any node $v$ at depth $i$ is associated with a sequence of query sets $\mathcal{A}_{i-1} = (A_1, \cdots, A_{i-1})$ such that the sizes of all atoms $A \in \mathsf{At}(\mathcal{A}_{i-1})$ are fixed.

**(ii)** The node $v$ is labeled by a vector $\boldsymbol{k}_v = (k_i^A)_{A \in \mathsf{At}(\mathcal{A}_{i-1})}$ and a set $K_v \subseteq [i-1]$ which are used to determine the next query set $A_i = O_i \cup U_i$. (Again, $|U_i| = \sum_{A \in \mathsf{At}(\mathcal{A}_{i-1})} k_i^A$ and the set $U_i$ is fixed.) $A_i$ is used to place the next SAT-Sample oracle call.

**(iii)** For every possible value of the configuration at step $i$, there is a corresponding child of the node $v$, with the corresponding edge labeled by the value of the configuration.

For any node $v$, we use $A_v = O_v \cup U_v$ to denote the (random) query set (corresponding to the node $v$) determined by the $\boldsymbol{k}_v$ and $K_v$. Note that $|U_v| = \sum_{A \in \mathsf{At}(\mathcal{A}_{i-1})} k_i^A$. Further, we use $\mathcal{A}_v := (A_1, \cdots, A_v)$ for the sequence of query sets corresponding to a path to $v$ and node $v$. Observe the number of possible outcomes of the counter $T$ at any step $i$ is at most $i + 2^i + 1 \leq 2^{q+1}$ (since $i \leq q$). So the total number of nodes in the decision tree corresponding to the semi-oblivious counter $T$ is at most $2^{O(q^2)}$.

## 3.2 Hard instance

We will provide a set of inputs $\mathcal{X}$ (which, in our case, will be a set of formulas) and a distribution $\Gamma$ over $\mathcal{X}$. Then we will show that any deterministic semi-oblivious counter $D$ (note that $D$ knows $\mathcal{X}$ and $\Gamma$) which receives as input a formula $\phi \in \mathcal{X}$ randomly drawn as per distribution $\Gamma$ and returns an $(\epsilon, \delta)$-approximation of $\mathsf{sol}(\phi)$, must make $\tilde{\Omega}(\log n)$ queries to the $SAT$-oracle.

Let $k = (\log n)^9$. Let $\mathcal{X}$ be the set of all formulas (with $n$ variables). We now define the hard distribution $\Gamma$ over $\mathcal{X}$ as follows by describing the procedure of picking a formula in $\mathcal{X}$ according to $\Gamma$.

1. Pick $\ell \in \{\lfloor n^{1/4} \rfloor, \lfloor n^{1/4} \rfloor + 1, \ldots, \lceil n^{3/4} \rceil\}$ uniformly at random.
2. Draw a hash function $h_\ell \leftarrow H(n, \ell, k)$ uniformly at random.
3. Let $\phi_\ell$ be the formula whose set of satisfying assignments is $h_\ell^{-1}(F^\ell)$. (Recall, $h_\ell : \{T, F\}^n \to \{T, F\}^\ell$.)
4. The formula $\phi_\ell$ is the picked formula.

### 3.2.1 Properties of the hard instance

Let $f_\ell := \mathbb{E}[|\mathsf{sol}(\phi_\ell)|] = \mathbb{E}[|h_\ell^{-1}(F^\ell)|]$ for $\ell \in \{\lfloor n^{1/4} \rfloor, \lfloor n^{1/4} \rfloor + 1, \ldots, \lceil n^{3/4} \rceil\}$. Observe, it follows from the construction of $\phi_\ell$ and the properties of hash functions that $f_\ell = \frac{2^n}{2^\ell}$.

▶ **Lemma 6.** *With probability at least $1 - n2^{-n/20}$, we have*

$$\text{for all } \ell,\ ||\mathsf{sol}(\phi_\ell])| - f_\ell| \leq 2^{-n/10} f_\ell. \tag{3}$$

**Proof.** It is straightforward to see that the variance of $|\mathsf{sol}(\phi_\ell)|$ is $Var[|\mathsf{sol}(\phi_\ell)|] \leq f_\ell$. So by Chebyshev's inequality,

$$\Pr\left[||\mathsf{sol}(\phi_\ell)| - f_\ell| \geq 2^{-n/5} f_\ell\right] \leq \frac{2^{n/5}}{f_\ell} \leq \frac{2^{n/5} \cdot 2^\ell}{2^n} \leq 2^{-n/20}.$$

The lemma now follows from a union bound over all $\ell$. ◀

▶ **Definition 7.** *Once $\ell \in \{\lfloor n^{1/4} \rfloor, \lfloor n^{1/4} \rfloor + 1, \ldots, \lceil n^{3/4} \rceil\}$ has been picked in Step 1 of the construction of the hard instance (Section 3.2), let for any $S \subseteq \{T, F\}^n$*

$$\mathsf{N}_\ell(S) = \mathbb{E}\left[|\mathsf{sol}(\phi_\ell) \cap S|\right],$$

*where the expectation is over the choice of the hash function is Step 2 of the construction of the hard instance.*

Note that for any $S \subseteq \{T, F\}^n$ the value of $\mathsf{N}_\ell(S)$ is $|S|/2^\ell$.

▶ **Lemma 8.** *With probability at least $1 - \frac{2^{O(q^2)}}{n^{(\log n)^4}}$, the following holds:*
*For any node $v$ in the decision tree $R$ and any atom $A \in \mathsf{At}(U_v)$,*

1. *If $\mathsf{N}_\ell(U_v) < \frac{1}{n^{(\log n)^4}}$ then $|U_v \cap \mathsf{sol}(\phi_\ell)| = 0$. Similarly, if $\mathsf{N}_\ell(A) < \frac{1}{n^{(\log n)^4}}$ for any atom $A \in \mathsf{At}(U_v)$ then $|A \cap \mathsf{sol}(\phi_\ell)| = 0$*

2. *If* $\mathsf{N}_\ell(U_v) \geq (\log n)^5$ *then* $\frac{1}{2}\mathsf{N}_\ell(U_v) \leq |U_v \cap \mathsf{sol}(\phi_\ell)| \leq \frac{3}{2}\mathsf{N}_\ell(U_v)$. *Similarly, if* $\mathsf{N}_\ell(A) \geq (\log n)^5$ *then* $\frac{1}{2}\mathsf{N}_\ell(A) \leq |A \cap \mathsf{sol}(\phi_\ell)| \leq \frac{3}{2}\mathsf{N}_\ell(A)$
3. *If* $\mathsf{N}_\ell(U_v) \leq (\log n)^5$ *then* $|U_v \cap \mathsf{sol}(\phi_\ell)| \leq 2(\log n)^5$. *Similarly, if* $\mathsf{N}_\ell(A) \leq (\log n)^5$ *then* $|A \cap \mathsf{sol}(\phi_\ell)| \leq 2(\log n)^5$.

**Proof.** From Markov's inequality, we have

$$\Pr[|U_v \cap \mathsf{sol}(\phi_\ell)| \geq 1] \leq \Pr\left[|U_v \cap \mathsf{sol}(\phi_\ell)| \geq \left(\frac{1}{\mathsf{N}_\ell(U_v)} - 1\right)\mathsf{N}_\ell(U_v)\right] \leq 2\mathsf{N}_\ell(U_v)$$

Taking a union bound over all nodes $v$ with $\mathsf{N}_\ell(U_v) < \frac{1}{n^{(\log n)^4}}$ and all possible values of $\ell$ (which can take $O(n^{3/4})$ values), we get the first part.

From the first part of the Lemma 1, by setting $\gamma = 1/2$, we have

$$\Pr[|U_v \cap \mathsf{sol}(\phi_\ell)| \geq \mathsf{N}_\ell(U_v)] \leq \exp\left(-\frac{\mathsf{N}_\ell(U_v)}{12}\right)$$

for all nodes $v$ in $R$ such that $\mathsf{N}_\ell(U_v) \geq (\log n)^5$ (note that we have $k = (\log n)^9 > \gamma^2\mathsf{N}_\ell(U_v)$). Taking a union bound over all such nodes $v$ and all possible values of $\ell$, we get the second bound.

Let $\gamma_v = \frac{(\log n)^5}{\mathsf{N}_\ell(U_v)}$. Since $k = (\log n)^9 > \gamma_v\mathsf{N}_\ell(U_v)$, from the second part of Lemma 1, we have

$$\Pr[|U_v \cap \mathsf{sol}(\phi_\ell)| \geq \gamma_v\mathsf{N}_\ell(U_v)] \leq \exp\left(-\gamma_v\frac{\mathsf{N}_\ell(U_v)}{3}\right) \leq O\left(\frac{1}{n^{(\log n)^4}}\right).$$

for all nodes $v$ such that $\mathsf{N}_\ell(U_v) \leq (\log n)^5$. Taking a union bound over all such nodes $v$ and all possible values of $\ell$, we get the third part. ◀

## 3.3 Proof of Theorem 1.2

**Proof of Theorem 1.2.** By Lemma 5 and Yao's minmax theorem we can assume that our SAT-Sample counter Alg is a (deterministic) semi-oblivious counter whose input is a randomly chosen formula $\phi \in \phi_n$, as per distribution $\Gamma$ and Alg returns Est which is an $(\epsilon, 2/3)$-approximation of $|\mathsf{sol}(\phi)|$. We will prove that Alg must make $q = \tilde{\Omega}(\log n)$ many $SAT$-oracle calls.

Recall the distribution $\Gamma$ (Section 3.2) over the set of all formulas. We can assume that the input to Alg is $\phi_\ell$, where $\ell$ is uniformly drawn from the set $\{\lfloor n^{1/4}\rfloor, \lfloor n^{1/4}\rfloor + 1, \ldots, \lceil n^{3/4}\rceil\}$.

Consider the path taken by the semi-oblivious counter Alg in the decision tree. Let the $i$th query made by Alg (that is at vertex $v_i$) be $A_i = U_i \cup O_i$ (as in Definition 4). Let $Z_i$ be the configuration (denoted as $c_i$ in Definition 4) of the sample from $A_i$. Note that the domain of $Z_i$ is $\Omega_i := O_i \cup \mathsf{At}(U_i) \cup \perp$.

Let Good be the event that the condition in Equation 3 (in Lemma 6) and the condition in Lemma 8 holds. Note that by Lemma 6 and Lemma 8 if $q \leq \log n$ then

$$\Pr[\mathsf{Good}] = 1 - o(1). \tag{4}$$

Let $X$ be the random variable that takes values in $\{\lfloor n^{1/4}\rfloor, \lfloor n^{1/4}\rfloor + 1, \ldots, \lceil n^{3/4}\rceil\}$ uniformly at random (in Step 1 of the construction of hard instance). Note that by the triangle inequality

$$|\mathsf{Est} - |\mathsf{sol}(\phi_\ell)|| \geq \left|\mathsf{Est} - \frac{2^n}{2^\ell}\right| - \left|\frac{2^n}{2^\ell} - |\mathsf{sol}(\phi_\ell)|\right|. \tag{5}$$

By Lemma 6 we know that with probability at least $(1 - 1/6)$, we have $|\frac{2^n}{2^\ell} - |\mathsf{sol}(\phi_\ell)|| \leq \frac{1}{2^{n/10}} \cdot \frac{2^n}{2^\ell}$. On the other hand, since $\mathsf{Alg}$ outputs an $(\epsilon, \delta)$-approximation of $|\mathsf{sol}(\phi)|$ (with $\epsilon < 1/2$ and $\delta < 1/6$), Equation 5 implies that with probability at least $(1 - \frac{1}{6} - \delta) \geq \frac{2}{3}$ we have

$$\left|\mathsf{Est} - \frac{2^n}{2^\ell}\right| \leq \left(\epsilon + \frac{1}{2^{n/10}}\right)\frac{2^n}{2^\ell} \leq \frac{1}{2} \cdot \frac{2^n}{2^\ell}, \tag{6}$$

where the last inequality follows from the fact that $\epsilon \leq 1/3$. Since $|\frac{2^n}{2^\ell} - \frac{2^n}{2^{\ell'}}| > \frac{1}{2} \cdot \frac{2^n}{2^\ell}$ for any integer $\ell' \neq \ell$, so Equation 6 is satisfied only when $\hat{X}$ is same as the picked $\ell$ (that is $\hat{X} = X$) where,

$$\hat{X} = \underset{\ell \in \{\lfloor n^{1/4} \rfloor, \lfloor n^{1/4} \rfloor + 1, \ldots, \lceil n^{3/4} \rceil\}}{\arg\min} \left|\frac{2^n}{2^\ell} - \mathsf{Est}\right|.$$

Hence, assuming $\mathsf{Good}$

$$\frac{1}{3} \geq \Pr[\hat{X} \neq X]. \tag{7}$$

By Fano's Inequality (Theorem 3)

$$\Pr[\hat{X} \neq X] \geq 1 - \frac{I(X; \hat{X})}{O(\log n)} \tag{8}$$

Since the final outcome of the algorithm is determined by the outcome at each step, i.e., $\boldsymbol{Z} = (Z_1, \ldots, Z_q)$, so by the data processing inequality (Equation 2), we have

$$I(X; \hat{X}) \leq I(X; Z_1, \ldots, z_q). \tag{9}$$

Let $Y_i$ be the random variable that defined as

$$Y_i = \begin{cases} 1 & \text{if } \frac{1}{n^{(\log n)^4}} \leq \mathsf{N}_\ell(U_i) \leq n^{(\log n)^4} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

Again by the data-processing inequality (Equation 2), we have

$$I(X; Z_1, \ldots, Z_q) \leq I(X; Y_1, Z_1, \ldots, Y_q, Z_q). \tag{11}$$

By the chain rule of mutual information, we have

$$I(X; Y_1, Z_1, \ldots, Y_q, Z_q) = \sum_{i \in [q]} I(X; Y_i, Z_i | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}) \tag{12}$$

Finally, we will show, in the following lemma, that conditioned on the fact $\mathsf{Good}$ happens we can upper bound $I(X; Y_1, Z_1, \ldots, Y_q, Z_q)$ by $O(\log \log n)$.

▶ **Lemma 9.** $I(X; (Y_1, Z_1, \ldots, Y_q, Z_q)) \leq q(O(\log \log n) + O(\log q) + \frac{2^{2q} poly(\log n)}{n^{(\log n)^3}})$.

We defer the proof of Lemma 9 and complete the proof of Theorem 1.2 assuming Lemma 9.

From the Equations 7, 8, 9, 11 and Lemma 9, we have that assuming Good happens

$$\begin{aligned}
\frac{1}{3} \geq & \Pr[\hat{X} \neq X] && \text{[From Equation 7]}\\
\geq & 1 - \frac{I(X;\hat{X})}{O(\log n)} && \text{[From Equation 8]}\\
\geq & 1 - \frac{I(X;Z_1,\ldots,z_q)}{O(\log n)} && \text{[From Equation 9]}\\
\geq & 1 - \frac{I(X;Y_1,Z_1,\ldots,Y_q,Z_q)}{O(\log n)} && \text{[From Equation 11]}\\
\geq & 1 - \frac{I(X;Y_1,Z_1,\ldots,Y_q,Z_q)}{O(\log n)} && \text{[From Equation 12]}\\
\geq & 1 - \frac{q\log\log n}{\log n} && \text{[From Lemma 9]}
\end{aligned}$$

Thus, from Equation 4, if $q \leq \log n$

$$1 - \frac{q\log\log n}{\log n} \leq \frac{1}{3} + \Pr[\mathsf{Good}] \leq \frac{1}{3} + O(1)$$

which implies

$$q = \Omega\left(\frac{\log n}{\log\log n}\right).$$ ◀

### 3.3.1 Proof of Lemma 9

▶ **Lemma 10.** *The following holds:*

1. *Conditioned on event that $Y_j = 1$ for some $j \leq i$,*
$$I(X; Z_i | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}, Y_i) \leq O(\log\log n),$$
2. $I(X, Y_i | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}) \leq 1$,
3. *Conditioned on the event that $Y_1 = 0, \ldots, Y_{i-1} = 0$,*
$$I(X, Z_i | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}, Y_i) \leq O(\log q) + \frac{2^{2q} poly(\log n)}{n^{(\log n)^3}}.$$

**Proof.** We will prove Part 1, 2, and 3 one by one.

**Proof of Part 1.** We will prove that conditioned on event that $Y_j = 1$ for some $j \leq i$,

$$I(X; Z_i | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}, Y_i) \leq O(\log\log n).$$

From (1), we have

$$I(X, Z_i | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}, Y_i) \leq H(X | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}, Y_i).$$

Note that if $Y_j = 1$ then by definition of $Y_j$ we have $\frac{1}{n^{(\log n)^4}} \leq \frac{|U_j|}{2^\ell} \leq n^{(\log n)^4}$, that is,

$$\frac{|U_j|}{n^{(\log n)^4}} \leq 2^\ell \leq |U_j| n^{(\log n)^4}.$$

Note that by definition of the semi-oblivious counter the sets $|U_1|, \ldots, |U_i|$ are deterministically determined by $Z_1, \ldots, Z_i$ . Thus, there are $O(\log(n^{(\log n)^4})) = O((\log n)^5)$ possible values of $\ell$ and hence

$$H(X | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}) \leq O(\log\log n).$$

This proves the first part.

**Proof of Part 2.** Since $Y_i$ can take only binary values, we have

$$I(X, Y_i | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}) \leq 1.$$

This proves Part 2.

**Proof of Part 3.** We will now prove the upper bound on $I(X; (Y_i, Z_i) | Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1})$ for each $i \in [q]$, conditioned on $Y_j = 0$ for all $j \in [i]$.

Note that $Z_1, \ldots, Z_{i-1}$ fixes the size of $O_i$ and each atoms in $\mathsf{At}(U_i)$. Note that the domain of $Z_i$, i.e., $\Omega_i$ is $\perp \cup O_i \cup \mathsf{At}(U_i)$. Let $r = |O_i| + 2 \leq q + 2$.

We define an auxiliary distribution $Q_{(Y_i,Z_i)}$ as follows:

$$Q_{(Y_i,Z_i)}(y_i, z_i) := Q_{Y_i}(y_i) Q_{Z_i|Y_i}(z_i | y_i)$$

where, $Q_{Y_i}(0) = Q_{Y_i}(1) = 1/2$ and

$$Q_{Z_i|Y_i}(z_i|y_i) = \begin{cases} \frac{1}{r}, & z_i \in O_i \cup \perp \\ \frac{1}{r} \cdot \frac{|z_i|}{|U_i|}, & z_i \in \mathsf{At}(U_i) \end{cases}$$

Let $P_X, P_Z, P_{Z|X}$ be the marginal distributions corresponding to a pair $(X, Z)$. Conditioned on $Y_j = 0$ for all $j \in [i]$ and $Z_j = z_j$ for all $j \in [i-1]$ for any $(z_1, \ldots, z_{i-1}) \in \Omega_1 \times \cdots \times \Omega_{i-1}$, we have for any $\ell \in \mathcal{X}$ (note that, for brevity, we have ignored the conditioning on $Y_1, Z_1, \ldots, Y_{i-1}, Z_{i-1}$, in the expression below)

$$KL(P_{Z_i|X}(\cdot|X = \ell) || Q_{Z_i}) = \sum_{z_i \in \Omega_i} P_{Z_i|X}(z_i|X = \ell) \log \frac{P_{Z|X}(z_i|X = \ell)}{Q_{Z_i}(z_i)} \tag{13}$$

Note that if $z_i \in \perp \cup O_i$ then $Q_{Z_i}(z_i) = \frac{1}{r} \geq \frac{1}{q+2}$. Hence,

$$\frac{P_{Z_i|X}(z_i|X = \ell)}{Q_{Z_i}(z_i)} \leq q + 2 \leq 2q.$$

Now we consider the case when $z_i \in \mathsf{At}(U_i)$.

If $\mathsf{N}_\ell(z_i) \geq (\log n)^5$ then from Lemma 8 we have

$$P_{Z_i|X}(z_i|X = \ell) = \frac{|z_i \cap \mathsf{sol}(\phi_\ell)|}{|U_i \cap \mathsf{sol}(\phi_\ell)|} \leq 3\mathsf{N}_\ell(z_i)/\mathsf{N}_\ell(U_i).$$

Note that

$$Q_{Z_i}(z_i) = \frac{1}{r} \cdot \frac{|z_i|}{|U_i|} \geq 2q\mathsf{N}_\ell(z_i)/\mathsf{N}_\ell(U_i).$$

Therefore, we have

$$\frac{P_{Z_i|X}(z_i|X = \ell)}{Q_{Z_i}(z_i)} \leq O(q).$$

For the case when $\mathsf{N}_\ell(z_i) < \frac{1}{n^{(\log n)^4}}$, we have $|z_i \cap \mathsf{sol}(\phi_\ell)| = 0$. Hence the sum

$$\sum_{z_i} P_{Z_i|X}(z_i|X = \ell) \log \frac{P_{Z|X}(z_i|X = \ell)}{Q_{Z_i}(z_i)}$$

when, $z_i \in \perp \cup O_i$ or $z_i \in \mathsf{At}(U_i)$ such that $\mathsf{N}_\ell(z_i) \geq (\log n)^5$ or $\mathsf{N}_\ell(z_i) < \frac{1}{n^{(\log n)^4}}$, is at most $O(\log q)$.

Now we bound the sum

$$\sum_{z_i} P_{Z_i|X}(z_i|X=\ell)\log\frac{P_{Z|X}(z_i|X=\ell)}{Q_{Z_i}(z_i)}$$

when $z_i \in \mathsf{At}(U_i)$ such that

$$\frac{1}{n^{(\log n)^4}} < \mathsf{N}_\ell(z_i) < (\log n)^5.$$

If $\mathsf{N}_\ell(z_i) \le (\log n)^5$ then we have

$$|z_i \cap \mathsf{sol}(\phi_\ell)| \le 2(\log n)^5$$

and thus

$$P_{Z_i|X}(z_i|X=\ell) \le \frac{4(\log n)^5}{\mathsf{N}_\ell(U_i)}.$$

Note that

$$Q_{Z_i}(z_i) = \frac{1}{r}\cdot\frac{|z_i|}{|U_i|} \ge \frac{1}{2q}\mathsf{N}_\ell(z_i)/\mathsf{N}_\ell(U_i).$$

Hence,

$$\frac{P_{Z|X}(z_i|X=\ell)}{Q_{Z_i}(z_i)} \le O(q(\log n)^5/\mathsf{N}_\ell(z_i)).$$

Therefore, we have

$$\begin{aligned}
&\sum_{z_i:\frac{1}{n^{(\log n)^4}}<\mathsf{N}_\ell(z_i)\le(\log n)^5} P_{Z_i|X}(z_i|X=\ell)\log\frac{P_{Z|X}(z_i|X=\ell)}{Q_{Z_i}(z_i)}\\
&< \sum_{z_i:\frac{1}{n^{(\log n)^4}}<\mathsf{N}_\ell(z_i)\le(\log n)^5} \frac{4(\log n)^5}{\mathsf{N}_\ell(U_i)}\log(2q(\log n)^5/\mathsf{N}_\ell(z_i))\\
&\le 2^q\frac{8(\log n)^5}{n^{(\log n)^4}}\log(2q(\log n)^5 n^{(\log n)^4})\\
&\le \frac{2^{2q}poly(\log n)}{n^{(\log n)^4}}.
\end{aligned}$$

The second last inequality follows because there are at most $2^q$ possible values of such $z_i$, $\mathsf{N}_\ell(U_i) \ge n^{(\log n)^3}/2$ and $\mathsf{N}_\ell(z_i) \ge \frac{1}{n^{(\log n)^3}}$.

Now by Lemma 2 conditioned on the event that $Y_j = 0$ for all $j \le i$ we have

$$\begin{aligned}
I(X;Z_i|Y_1,Z_1,\ldots,Y_{i-1},Z_{i-1},Y_i) &\le KL(P_{Z_i|X}(\cdot|X=\ell)||Q_{Z_i})\\
&\le O(\log q) + \frac{2^{2q}poly(\log n)}{n^{(\log n)^3}}.
\end{aligned}$$

◀

**Proof of Lemma 9.** We will first prove that for any $i$

$$I(X;(Y_i,Z_i)|Y_1,Z_1,\ldots,Y_{i-1},Z_{i-1}) \le O(\log\log n) + O(\log q) + \frac{2^{2q}poly(\log n)}{n^{(\log n)^3}}.$$

By the chain rule of mutual information,

$$\begin{aligned}
&I(X;(Y_i,Z_i)|Y_1,Z_1,\ldots,Y_{i-1},Z_{i-1})\\
=&I(X;Y_i|Y_1,Z_1,\ldots,Y_{i-1},Z_{i-1})+I(X;Z_i|Y_1,Z_1,\ldots,Y_{i-1},Z_{i-1},Y_i)\\
\leq&O(\log\log n)+O(\log q)+\frac{2^{2q}poly(\log n)}{n^{(\log n)^3}},
\end{aligned}$$

where the last inequality follows from Lemma 10.

Again by the chain rule of mutual information, we have

$$\begin{aligned}
&I(X;(Y_1,Z_1,\ldots,Y_q,Z_q))\\
=&\sum_{i=1}^{q} I(X;(Y_i,Z_i)|Y_1,Z_1,\ldots,Y_{i-1},Z_{i-1})\\
\leq& q(O(\log\log n)+O(\log q)+\frac{2^{2q}poly(\log n)}{n^{(\log n)^3}}).
\end{aligned}$$

◀

## 4 Conclusion

In this paper, we study the power of SAT oracles in the context of approximate model counting and show a lower bound of $\tilde{\Omega}(\log n)$ on the number of oracle calls. This is in contrast to other settings where a SAT oracle is provably more powerful than an NP oracle. In fact, we prove that even with a much more powerful oracle (namely SAT-Sample oracle), the number of queries needed to approximately count the number of satisfying assignments of a Boolean formula is $\tilde{\Omega}(\log n)$.

## References


1. Teodora Baluta, Shiqi Shen, Shweta Shinde, Kuldeep S Meel, and Prateek Saxena. Quantitative verification of neural networks and its security applications. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pages 1249–1264, 2019.
2. Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of NP-witnesses using an NP-oracle. *Information and Computation*, 163(2):510–526, 2000.
3. Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power of conditional samples in distribution testing. *SIAM J. Comput.*, 45(4):1261–1296, 2016. doi:10.1137/140964199.
4. Supratik Chakraborty, Kuldeep S Meel, and Moshe Y Vardi. Algorithmic improvements in approximate counting for probabilistic inference: From linear to logarithmic sat calls. Technical report, Rice University, 2016.
5. Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. The MIT Press and McGraw-Hill Book Company, 1989.
6. Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006.
7. Remi Delannoy and Kuldeep S Meel. On almost-uniform generation of SAT solutions: The power of 3-wise independent hashing. In *Proceedings of the 37th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 1–10, 2022.
8. Leonardo Duenas-Osorio, Kuldeep Meel, Roger Paredes, and Moshe Vardi. Counting-based reliability estimation for power-transmission grids. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31(1), 2017.
9. Matthew Fredrikson and Somesh Jha. Satisfiability modulo counting: A new approach for analyzing privacy properties. In *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–10, 2014.

10 Carla P Gomes, Ashish Sabharwal, and Bart Selman. Model counting: A new strategy for obtaining good bounds. In *AAAI*, volume 10, pages 1597538–1597548, 2006.

11 Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical computer science*, 43:169–188, 1986.

12 Dan Roth. On the hardness of approximate reasoning. *Artificial Intelligence*, 82(1-2):273–302, 1996.

13 Tian Sang, Paul Beame, and Henry A Kautz. Performing bayesian inference by weighted model counting. In *AAAI*, volume 5, pages 475–481, 2005.

14 Jonathan Scarlett and Volkan Cevher. An introductory guide to Fano's inequality with applications in statistical estimation. *arXiv preprint*, 2019. `arXiv:1901.00555`.

15 Jeanette P Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff–Hoeffding bounds for applications with limited independence. *SIAM Journal on Discrete Mathematics*, 8(2):223–250, 1995.

16 Larry Stockmeyer. The complexity of approximate counting. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 118–126, 1983.

17 Leslie G Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

18 Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *18th Annual Symposium on Foundations of Computer Science (SFCS 1977)*, pages 222–227. IEEE Computer Society, 1977.


## A Proof of Lemma 5

Consider any general SAT-Sample counter, $T$. We will show that there exists a semi-oblivious counter that performs similarly. Given a sequence of query-sample pairs $\{(A_1, s_1), \ldots, (A_{i-1}, s_{i-1})\}$, we say the query $A_i$ is a good strategy by $T$ (given $\{(A_1, s_1), \ldots, (A_{i-1}, s_{i-1})\}$) if the counter $T$ can return the correct output by fixing the next query to $A_i$. It suffices to show that, given a sequence of query-sample pairs $\{(A_1, s_1), \ldots, (A_{i-1}, s_{i-1})\}$, if $A_i$ is a good strategy then any $A'_i$ is also a good strategy if $A'_i \cap \{s_1, \ldots, s_{i-1}\} = A_i \cap \{s_1, \ldots, s_{i-1}\}$ and $|A'_i \cap A| = |A_i \cap A|$ for atoms $A \in At(A_1, \ldots, A_{i-1})$. This means that to fix the next query, all it requires to fix the intersection size with each atom $A \in At(A_1, \ldots, A_i)$ and a subset of $\{s_1, \ldots, s_{i-1}\}$ (to be included in next query). We prove it in the following claim.

▷ Claim 11. Suppose $A_i$ is a good strategy for $\{(A_1, s_1), \ldots, (A_{i-1}, s_{i-1})\}$. Consider $A'_i$ such that $A'_i \cap \{s_1, \ldots, s_{i-1}\} = A_i \cap \{s_1, \ldots, s_{i-1}\}$ and $|A'_i \cap A| = |A_i \cap A|$ for atoms $A \in At(A_1, \ldots, A_{i-1})$. Then $A'_i$ is also a good strategy for $\{(A_1, s_1), \ldots, (A_{i-1}, s_{i-1})\}$.

Proof. We denote by $\mathcal{S}_N$ the symmetric group acting on a set of size $N$. Any $\sigma \in \mathcal{S}_N$ can be thought of acting on any set of size $N$ (by thinking the elements of the set as numbered $1, \ldots, N$ and $\sigma$ acting on the set $[N]$). For any element $x$ in the set, we will denote by $\sigma(x)$ the element after the action of $\sigma$. For any $\sigma \in \mathcal{S}_N$ and set $A$ (with $|A| = N$) we denote by $\sigma(A)$ the following set $\sigma(A) := \{\sigma(x) \mid x \in A\}$.

Let $\sigma \in \mathcal{S}_{2^n}$ be a permutation acting on the set $\{T, F\}^n$. For any $\phi$ observe that $|\mathsf{sol}(\phi)| = |\sigma(\mathsf{sol}(\phi))|$. Since any counter estimates $|\mathsf{sol}(\phi)|$ only, we observe that if $A_i$ is a good strategy for $\{(A_1, s_1), \ldots, (A_{i-1}, s_{i-1})\}$ then $\sigma(A_i)$ is also a good strategy for $\{(\sigma(A_1), \sigma(s_1)), \ldots, (\sigma(A_{i-1}), \sigma(s_{i-1})\}$ for any $\sigma : \{T, F\}^n \to \{T, F\}^n$ that preserves the atoms $\mathsf{At}(A_1, \ldots, A_{i-1})$ and the elements $\{s_1, \ldots, s_{i-1}\}$.

Since $|A'_i \cap A| = |A_i \cap A|$ for atoms $A \in At(A_1, \ldots, A_{i-1})$ and $A'_i \cap \{s_1, \ldots, s_{i-1}\} = A_i \cap \{s_1, \ldots, s_{i-1}\}$, there exists a $\sigma$ such that $\sigma(A_j) = A_j$, $\sigma(s_j) = s_j$ for all $j \leq i-1$ and also $\sigma(A_i) = A'_i$. By our earlier observation, $A'_i$ is also a good strategy for $\{(A_1, s_1), \ldots, (A_{i-1}, s_{i-1})\}$. ◁