

1 Approximate Model Counting: Is SAT Oracle More 2 Powerful than NP Oracle?

3 **Diptarka Chakraborty** ✉

4 National University of Singapore, Singapore

5 **Sourav Chakraborty** ✉

6 Indian Statistical Institute, Kolkata

7 **Gunjan Kumar** ✉

8 National University of Singapore, Singapore

9 **Kuldeep S. Meel** ✉

10 National University of Singapore, Singapore

11 — Abstract —

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

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

33 **2012 ACM Subject Classification** Theory of computation \rightarrow Oracles and decision trees

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

35 **Digital Object Identifier** 10.4230/LIPIcs.ICALP.2023.129

36 **Funding** *Diptarka Chakraborty*: Supported in part by an MoE AcRF Tier 2 grant (MOE-T2EP20221-
37 0009) and Google South & South-East Asia Research Award.

38 *Gunjan Kumar*: Supported in part by National Research Foundation Singapore under its NRF
39 Fellowship Programme[NRF-NRFFAI1-2019-0004]

40 *Kuldeep S. Meel*: Supported in part by National Research Foundation Singapore under its NRF Fellow-
41 ship Programme[NRF-NRFFAI1-2019-0004] and Campus for Research Excellence and Technological
42 Enterprise (CREATE) programme, Ministry of Education Singapore Tier 2 grant MOE-T2EP20121-
43 0011, and Ministry of Education Singapore Tier 1 Grant [R-252-000-B59-114]



© Diptarka Chakraborty, Sourav Chakraborty, Gunjan Kumar, and Kuldeep S. Meel;
licensed under Creative Commons License CC-BY 4.0

50th International Colloquium on Automata, Languages, and Programming (ICALP 2023).

Editors: Kousha Etessami, Uriel Feige, and Gabriele Puppis; Article No. 129; pp. 129:1–129:18

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

44 **1 Introduction**

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

54 **Approximate Model Counting**

56 **Input** A formula ϕ , tolerance parameter $\epsilon > 0$, and confidence parameter $\delta \in (0, 1)$.

57 **Output** Compute an estimate **Est** such that

$$58 \Pr \left[\frac{|\text{sol}(\phi)|}{1 + \epsilon} \leq \text{Est} \leq (1 + \epsilon)|\text{sol}(\phi)| \right] \geq 1 - \delta.$$

60 Stockmeyer [16] initiated the study of the complexity of approximate model counting.
 61 Stockmeyer’s seminal paper made two foundational contributions: the first contribution was
 62 to define the query model that could capture possible natural algorithms yet amenable enough
 63 to theoretical tools to allow non-trivial insight. To this end, Stockmeyer proposed the query
 64 model wherein one can construct an arbitrary set S and query an NP oracle to determine if
 65 $|\text{sol}(\phi) \cap S| \geq 1$. Stockmeyer showed that under the above-mentioned query model, $\log n$ calls
 66 to an NP oracle are necessary and sufficient (for a fixed ϵ and δ). Furthermore, Stockmeyer
 67 introduced a hashing-based algorithmic procedure to achieve the desired upper bound that
 68 makes $O(\log n)$ calls to NP-oracle. The lack of availability of powerful reasoning systems for
 69 problems in NP dissuaded the development of algorithmic frameworks based on Stockmeyer’s
 70 hashing-based framework until the early 2000s [10].

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

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

90 Delannoy and Meel [7] sought to bridge the gap between theory and practice by proposing
 91 the notion of a SAT oracle. Formally, a SAT oracle takes in a Boolean formula ϕ as input
 92 and returns a satisfying assignment $s \in \text{sol}(\phi)$ if ϕ is satisfiable and \perp , otherwise. It is worth
 93 highlighting that we may need n calls to an NP oracle to simulate a query to a SAT oracle,
 94 and therefore, it is conceivable for an algorithm to make $O(\log n)$ calls to a SAT oracle but
 95 $O(n \log n)$ calls to an NP oracle. Delannoy and Meel showcased precisely such behavior
 96 in the context of *almost-uniform generation*. Their proposed algorithm, **UniSamp** makes
 97 $O(\log n)$ calls to a SAT oracle and would require $O(n \log n)$ calls to an NP oracle if one were
 98 to replace a SAT oracle with an NP oracle. At the same time, it is not necessary that there
 99 would be a gap of n calls for every algorithm: simply consider the problem of determining
 100 whether a formula is satisfiable or not. Only one call to an NP oracle (and similarly to a
 101 SAT oracle) suffices.

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

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

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

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

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

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

138 ► **Theorem 1.1.** *For any $\epsilon, \delta \in (0, 1)$, given a formula ϕ , computation of (ϵ, δ) -approximation
139 of $|\text{sol}(\phi)|$ requires $\tilde{\Omega}(\log n)^1$ queries to a SAT oracle.*

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

143 1.1 Technical Overview

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

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

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

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

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

178 To start with, we argue that we can assume that the **SAT-Sample** counter is "semi-
179 oblivious" in nature. The number of satisfying assignments of a formula does not change

¹ The tilde hides a factor of $\log \log n$

180 by any permutation of the elements in $\{T, F\}^n$, and the SAT-Sample counter can only get
 181 elements of $\text{sol}(\phi)$ by querying the SAT-Sample oracle. So we argue that the only useful
 182 information of the i^{th} query set (that is, the set of satisfying assignments of the formula
 183 that is given to the SAT-Sample oracle) is the size of its intersection with the previous $(i - 1)$
 184 query sets and their corresponding answers. We formalize it in Section 3.1.

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

188 For the hard distribution, we construct $O(n^{3/4})$ formulas ϕ_ℓ for each value of ℓ in
 189 the set $\{\lfloor n^{1/4} \rfloor, \lfloor n^{1/4} \rfloor + 1, \dots, \lceil n^{3/4} \rceil\}$. The formulas ϕ_ℓ are chosen in such a way that
 190 $|\text{sol}(\phi_\ell)| \approx 2|\text{sol}(\phi_{\ell+1})|$ thereby approximately counting the number of satisfying assignments
 191 (upto a multiplicative $(1 + \epsilon)$ -factor for small constant ϵ) reduces to the problem of determining
 192 the value of ℓ . The hard distribution is obtained by picking an ℓ uniformly at random from
 193 the set $\{\lfloor n^{1/4} \rfloor, \lfloor n^{1/4} \rfloor + 1, \dots, \lceil n^{3/4} \rceil\}$ and using the corresponding formula ϕ_ℓ .

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

208 **2 Notations and Preliminaries**

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

214 **Oracles and query model**

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

222 We rely on the query model introduced by Stockmeyer [16]: For a given ϕ whose model
 223 count we are interested in estimating, one can query the corresponding (NP/SAT) oracle
 224 with formulas of the form $\phi \wedge \psi_A$, where, as stated earlier, ψ_A is an (arbitrary) formula

129:6 Approximate Model Counting: Is SAT Oracle More Powerful than NP Oracle?

225 whose set of solutions is A . We will use ϕ_A as a shorthand to represent $\phi \wedge \psi_A$. Throughout
 226 this paper, we consider the above query model with query access to the SAT-Sample oracle.
 227 One call to the SAT-Sample oracle will be called a SAT-Sample query. By abuse of notation,
 228 we sometimes say “ A is queried” to refer to the formula ϕ_A .

229 k -wise independent hash functions

230 Let n, m, k be positive integers and let $H(n, m, k)$ denote the family of k -wise independent
 231 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
 232 $h^{-1}(\alpha)$ denote the set $\{s \in \{T, F\}^n \mid h(s) = \alpha\}$.

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

237 Concentration inequalities for limited independence

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

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

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

242 Basics of information theory

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

$$245 \quad I(X; Y) = H(X) - H(X|Y) \leq H(X) \quad (1)$$

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

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

$$251 \quad KL(P||Q) := \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$$

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

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

$$255 \quad I(X; Y) := KL(Q_{X,Y}||Q_X \times Q_Y).$$

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

$$258 \quad I(X; Y|Z) := \mathbb{E}_Z[KL(Q_{(X,Y)|Z}||Q_{X|Z} \times Q_{Y|Z})].$$

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

$$261 \quad I(X; (Y, Z)) = I(X; Y) + I(X; Z|Y).$$

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

$$\begin{aligned}
 263 \quad \mathbb{E}_Z[KL(Q_{(X,Y)|Z} || Q_{X|Z} \times Q_{Y|Z})] &= \sum_{z \in \mathcal{Z}} Q_Z(z) \cdot KL(Q_{(X,Y)|Z=z} || Q_{X|Z=z} \times Q_{Y|Z=z}) \\
 264 \quad &= \sum_{z \in \mathcal{Z}} Q_Z(z) \cdot I(X; Y | Z = z). \\
 265
 \end{aligned}$$

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

$$268 \quad I(X, Z) = \sum_x P_X(x) KL(P_{Z|X}(\cdot|x) || P_Z) \leq \max_x KL(P_{Z|X}(\cdot|x) || Q_Z).$$

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

$$271 \quad \Pr[\hat{X} \neq X] \geq 1 - \frac{I(X; \hat{X}) + \log 2}{\log |\mathcal{V}|}.$$

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

$$274 \quad I(X; \hat{X}) \leq I(X; Z). \tag{2}$$

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

277 Minimax theorem

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

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

$$286 \quad \max_{X \in \mathcal{X}} \Pr[R \text{ gives wrong answer on } X] \geq \min_{T \in \mathcal{T}} \Pr_{X \sim \Gamma}[T \text{ gives wrong answer on } X].$$

287 **3 Lower Bound on the number of queries to SAT-Sample oracle**

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

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

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

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

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

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

316 3.1 Semi-oblivious counter

317 Suppose given a formula ϕ over n variables, a counter Alg makes q calls to the SAT-Sample
 318 oracle with queried formulas $\phi_{A_1}, \dots, \phi_{A_q}$ respectively, where each $A_i \subseteq \{T, F\}^n$. (Recall,
 319 $\phi_{A_i} = \phi \wedge \psi_{A_i}$, where ψ_{A_i} denote the formula having $\text{sol}(\psi_{A_i}) = A_i$.) Note, the i -th
 320 SAT-Sample oracle call by the counter Alg is specified by the set A_i . During the i -th call (for
 321 $1 \leq i \leq q$), suppose the counter Alg receives a sample $s_i \in A_i \cup \{\perp\}$. Note that the oracle
 322 calls made by Alg can be *adaptive*, i.e., the sets A_1, \dots, A_q are not fixed in advance – the
 323 counter Alg fixes A_i only after seeing the samples s_1, \dots, s_{i-1} (outcomes of all the previous
 324 oracle calls).

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

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

336 Given a family of sets $\mathcal{A} = \{A_1, \dots, A_i\}$, (where $A_i \subseteq \{T, F\}^n$), the *atoms* generated by \mathcal{A} ,
 337 denoted by $\text{At}(\mathcal{A})$, are (at most) 2^i distinct sets of the form $\cap_{j=1}^i C_j$ where $C_j \in \{A_j, \{T, F\}^n \setminus$
 338 $A_j\}$. For example, if $i = 2$, then $\text{At}(A_1, A_2) = \{A_1 \cap A_2, A_1 \setminus A_2, A_2 \setminus A_1, (A_1 \cup A_2)^c\}$.

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

341 ■ **Semi-oblivious choice:** Let $\mathcal{A}_{i-1} = \{A_1, \dots, A_{i-1}\}$, $S_{i-1} = \{s_1, \dots, s_{i-1}\}$, $C_{i-1} =$
 342 $\{c_1, \dots, c_{i-1}\}$ be the set of first $i-1$ query sets, the set of first $i-1$ samples obtained,
 343 the set of first $i-1$ configurations, respectively. Only based on C_{i-1} (without knowing the
 344 set S_{i-1}), T does the following:

- 345 ■ For each $A \in \text{At}(\mathcal{A}_{i-1})$, it generates an integer k_i^A between 0 and $|A \setminus S_{i-1}|$. (k_i^A
 346 indicates how many unseen elements from the atom A of the previous query sets are to
 347 be included in the next query set.)
- 348 ■ It chooses a set of indices $K_i \subseteq \{1, \dots, i-1\}$. (K_i specifies the index set of previous
 349 samples that are to be included in the next query set.)

350 ■ **Query set generation:** In this phase, it decides the query set A_i as follows:

- 351 ■ Let us define the candidate unseen set family as

$$352 \quad \mathcal{U}_i := \{U \subseteq \{T, F\}^n \setminus S_{i-1} \mid \forall A \in \text{At}(\mathcal{A}_{i-1}), |U_i \cap A| = k_i^A\}.$$

353 The algorithm T chooses a set U_i uniformly at random from the candidate unseen set
 354 family \mathcal{U}_{i-1} .

- 355 ■ Let us denote $O_i := \{s_j \mid j \in K_i\}$. The algorithm T decides the query set to be
 356 $A_i = U_i \cup O_i$.
- 357 ■ **Oracle call:** It places a query to the SAT-Sample oracle with the formula ϕ_{A_i} . Let the
 358 i -th configuration c_i specify whether $s_i = \perp$, or for which $j \in K_i$, $s_i = s_j$, or for which
 359 $A \in \text{At}(\mathcal{A}_{i-1})$, $s_i \in A \cap U_i$.

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

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

366 ► **Lemma 5.** *If there is an algorithm that, given any input ϕ on n variables, outputs an*
 367 *(ϵ, δ) -approximation of $|\text{sol}(\phi)|$ while placing at most $q = q(n)$ SAT-Sample oracle calls,*
 368 *then there also exists a (randomized) semi-oblivious counter that, given input ϕ , outputs an*
 369 *(ϵ, δ) -approximation of $|\text{sol}(\phi)|$ while also placing at most q SAT-Sample oracle calls.*

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

- 377 (i) A path (from root) to any node v at depth i is associated with a sequence of query sets
 378 $\mathcal{A}_{i-1} = (A_1, \dots, A_{i-1})$ such that the sizes of all atoms $A \in \text{At}(\mathcal{A}_{i-1})$ are fixed.
- 379 (ii) The node v is labeled by a vector $\mathbf{k}_v = (k_i^A)_{A \in \text{At}(\mathcal{A}_{i-1})}$ and a set $K_v \subseteq [i-1]$ which are
 380 used to determine the next query set $A_i = O_i \cup U_i$. (Again, $|U_i| = \sum_{A \in \text{At}(\mathcal{A}_{i-1})} k_i^A$ and
 381 the set U_i is fixed.) A_i is used to place the next SAT-Sample oracle call.
- 382 (iii) For every possible value of the configuration at step i , there is a corresponding child of
 383 the node v , with the corresponding edge labeled by the value of the configuration.

129:10 Approximate Model Counting: Is SAT Oracle More Powerful than NP Oracle?

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

3.2 Hard instance

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

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

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

3.2.1 Properties of the hard instance

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

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

$$407 \quad \text{for all } \ell, \quad ||\text{sol}(\phi_\ell)| - f_\ell| \leq 2^{-n/10} f_\ell. \quad (3)$$

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

$$410 \quad \Pr \left[||\text{sol}(\phi_\ell)| - f_\ell| \geq 2^{-n/5} f_\ell \right] \leq \frac{2^{n/5}}{f_\ell} \leq \frac{2^{n/5} \cdot 2^\ell}{2^n} \leq 2^{-n/20}.$$

411 The lemma now follows from a union bound over all ℓ . ◀

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

$$412 \quad \mathbf{N}_\ell(S) = \mathbb{E}[|\text{sol}(\phi_\ell) \cap S|],$$

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

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

416 **► Lemma 8.** *With probability at least $1 - \frac{2^{O(q^2)}}{n^{(\log n)^4}}$, the following holds:*

417 *For any node v in the decision tree R and any atom $A \in \text{At}(U_v)$,*

- 418 1. If $\mathbf{N}_\ell(U_v) < \frac{1}{n^{(\log n)^4}}$ then $|U_v \cap \text{sol}(\phi_\ell)| = 0$. Similarly, if $\mathbf{N}_\ell(A) < \frac{1}{n^{(\log n)^4}}$ for any atom
 419 $A \in \text{At}(U_v)$ then $|A \cap \text{sol}(\phi_\ell)| = 0$
- 420 2. If $\mathbf{N}_\ell(U_v) \geq (\log n)^5$ then $\frac{1}{2}\mathbf{N}_\ell(U_v) \leq |U_v \cap \text{sol}(\phi_\ell)| \leq \frac{3}{2}\mathbf{N}_\ell(U_v)$. Similarly, if $\mathbf{N}_\ell(A) \geq$
 421 $(\log n)^5$ then $\frac{1}{2}\mathbf{N}_\ell(A) \leq |A \cap \text{sol}(\phi_\ell)| \leq \frac{3}{2}\mathbf{N}_\ell(A)$
- 422 3. If $\mathbf{N}_\ell(U_v) \leq (\log n)^5$ then $|U_v \cap \text{sol}(\phi_\ell)| \leq 2(\log n)^5$. Similarly, if $\mathbf{N}_\ell(A) \leq (\log n)^5$ then
 423 $|A \cap \text{sol}(\phi_\ell)| \leq 2(\log n)^5$.

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

$$425 \Pr[|U_v \cap \text{sol}(\phi_\ell)| \geq 1] \leq \Pr\left[|U_v \cap \text{sol}(\phi_\ell)| \geq \left(\frac{1}{\mathbf{N}_\ell(U_v)} - 1\right) \mathbf{N}_\ell(U_v)\right] \leq 2\mathbf{N}_\ell(U_v)$$

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

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

$$430 \Pr[|U_v \cap \text{sol}(\phi_\ell)| \geq \mathbf{N}_\ell(U_v)] \leq \exp\left(-\frac{\mathbf{N}_\ell(U_v)}{12}\right)$$

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

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

$$437 \Pr[|U_v \cap \text{sol}(\phi_\ell)| \geq \gamma_v \mathbf{N}_\ell(U_v)] \leq \exp\left(-\gamma_v \frac{\mathbf{N}_\ell(U_v)}{3}\right) \leq O\left(\frac{1}{n^{(\log n)^4}}\right).$$

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

441 3.3 Proof of Theorem 1.2

442 **Proof of Theorem 1.2.** By Lemma 5 and Yao's minimax theorem, we can assume that
 443 our SAT-Sample counter Alg is a (deterministic) semi-oblivious counter whose input is a
 444 randomly chosen formula $\phi \in \phi_n$, as per distribution Γ and Alg returns Est which is an
 445 $(\epsilon, 2/3)$ -approximation of $|\text{sol}(\phi)|$. We will prove that Alg must make $q = \tilde{\Omega}(\log n)$ many
 446 SAT-oracle calls.

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

449 Consider the path taken by the semi-oblivious counter Alg in the decision tree. Let the
 450 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
 451 be the configuration (denoted as c_i in Definition 4) of the sample from A_i . Note that the
 452 domain of Z_i is $\Omega_i := O_i \cup \text{At}(U_i) \cup \perp$.

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

$$455 \Pr[\text{Good}] = 1 - o(1). \tag{4}$$

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

$$459 \left| \text{Est} - |\text{sol}(\phi_\ell)| \right| \geq \left| \text{Est} - \frac{2^n}{2^\ell} \right| - \left| \frac{2^n}{2^\ell} - |\text{sol}(\phi_\ell)| \right|. \tag{5}$$

129:12 Approximate Model Counting: Is SAT Oracle More Powerful than NP Oracle?

460 By Lemma 6 we know that with probability at least $(1 - 1/6)$, we have $|\frac{2^n}{2^\ell} - |\text{sol}(\phi_\ell)|| \leq$
 461 $\frac{1}{2^{n/10}} \cdot \frac{2^n}{2^\ell}$. On the other hand, since Alg outputs an (ϵ, δ) -approximation of $|\text{sol}(\phi)|$ (with
 462 $\epsilon < 1/2$ and $\delta < 1/6$), Equation 5 implies that with probability at least $(1 - \frac{1}{6} - \delta) \geq \frac{2}{3}$ we
 463 have

$$464 \quad \left| \text{Est} - \frac{2^n}{2^\ell} \right| \leq \left(\epsilon + \frac{1}{2^{n/10}} \right) \frac{2^n}{2^\ell} \leq \frac{1}{2} \cdot \frac{2^n}{2^\ell}, \quad (6)$$

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 ℓ (that is
 $\hat{X} = X$) where,

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

465 Hence, assuming Good

$$466 \quad \frac{1}{3} \geq \Pr[\hat{X} \neq X]. \quad (7)$$

467 By Fano's Inequality (Theorem 3)

$$468 \quad \Pr[\hat{X} \neq X] \geq 1 - \frac{I(X; \hat{X})}{O(\log n)} \quad (8)$$

469 Since the final outcome of the algorithm is determined by the outcome at each step, i.e.,
 470 $\mathbf{Z} = (Z_1, \dots, Z_q)$, so by the data processing inequality (Equation 2), we have

$$471 \quad I(X; \hat{X}) \leq I(X; Z_1, \dots, Z_q). \quad (9)$$

472 Let Y_i be the random variable that defined as

$$473 \quad Y_i = \begin{cases} 1 & \text{if } \frac{1}{n^{(\log n)^4}} \leq N_\ell(U_i) \leq n^{(\log n)^4} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

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

$$475 \quad I(X; Z_1, \dots, Z_q) \leq I(X; Y_1, Z_1, \dots, Y_q, Z_q). \quad (11)$$

476 By the chain rule of mutual information, we have

$$477 \quad I(X; Y_1, Z_1, \dots, Y_q, Z_q) = \sum_{i \in [q]} I(X; Y_i, Z_i | Y_1, Z_1, \dots, Y_{i-1}, Z_{i-1}) \quad (12)$$

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

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

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

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

$$483 \quad \frac{1}{3} \geq \Pr[\hat{X} \neq X] \quad [\text{From Equation 7}]$$

$$484 \quad \geq 1 - \frac{I(X; \hat{X})}{O(\log n)} \quad [\text{From Equation 8}]$$

$$485 \quad \geq 1 - \frac{I(X; Z_1, \dots, z_q)}{O(\log n)} \quad [\text{From Equation 9}]$$

$$486 \quad \geq 1 - \frac{I(X; Y_1, Z_1, \dots, Y_q, Z_q)}{O(\log n)} \quad [\text{From Equation 11}]$$

$$487 \quad \geq 1 - \frac{I(X; Y_1, Z_1, \dots, Y_q, Z_q)}{O(\log n)} \quad [\text{From Equation 12}]$$

$$488 \quad \geq 1 - \frac{q \log \log n}{\log n} \quad [\text{From Lemma 9}]$$

489

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

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

which implies

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

490

491 3.3.1 Proof of Lemma 9

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

1. *Conditioned on event that $Y_j = 1$ for some $j \leq i$,*

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

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

3. *Conditioned on the event that $Y_1 = 0, \dots, Y_{i-1} = 0$,*

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

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

495

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

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

From (1), we have

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

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,

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

129:14 Approximate Model Counting: Is SAT Oracle More Powerful than NP Oracle?

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

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

496 This proves the first part.

497

Proof of Part 2: Since Y_i can take only binary values, we have

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

498 This proves Part 2.

499

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

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

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

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

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

$$505 \quad Q_{Z_i|Y_i}(z_i|y_i) = \begin{cases} \frac{1}{r}, & z_i \in O_i \cup \perp \\ \frac{1}{r} \cdot \frac{|z_i|}{|U_i|}, & z_i \in \text{At}(U_i) \end{cases}$$

506 Let $P_X, P_Z, P_{Z|X}$ be the marginal distributions corresponding to a pair (X, Z) . Con-
507 ditioned on $Y_j = 0$ for all $j \in [i]$ and $Z_j = z_j$ for all $j \in [i-1]$ for any $(z_1, \dots, z_{i-1}) \in$
508 $\Omega_1 \times \dots \times \Omega_{i-1}$, we have for any $\ell \in \mathcal{X}$ (note that, for brevity, we have ignored the conditioning
509 on $Y_1, Z_1, \dots, Y_{i-1}, Z_{i-1}$, in the expression below)

$$510 \quad KL(P_{Z_i|X}(\cdot|X = \ell) || Q_{Z_i}) = \sum_{z_i \in \Omega_i} P_{Z_i|X}(z_i|X = \ell) \log \frac{P_{Z_i|X}(z_i|X = \ell)}{Q_{Z_i}(z_i)} \quad (13)$$

511

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,

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

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

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

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

Note that

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

Therefore, we have

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

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

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

513 when, $z_i \in \perp \cup O_i$ or $z_i \in \text{At}(U_i)$ such that $\mathbf{N}_\ell(z_i) \geq (\log n)^5$ or $\mathbf{N}_\ell(z_i) < \frac{1}{n(\log n)^4}$, is at most
514 $O(\log q)$.

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

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

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

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

and thus

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

Note that

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

Hence,

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

515 Therefore, we have

$$\begin{aligned} 516 & \sum_{z_i: \frac{1}{n(\log n)^4} < \mathbf{N}_\ell(z_i) \leq (\log n)^5} P_{Z_i|X}(z_i|X = \ell) \log \frac{P_{Z|X}(z_i|X = \ell)}{Q_{Z_i}(z_i)} \\ 517 & < \sum_{z_i: \frac{1}{n(\log n)^4} < \mathbf{N}_\ell(z_i) \leq (\log n)^5} \frac{4(\log n)^5}{\mathbf{N}_\ell(U_i)} \log(2q(\log n)^5 / \mathbf{N}_\ell(z_i)) \\ 518 & \leq 2^q \frac{8(\log n)^5}{n(\log n)^4} \log(2q(\log n)^5 n^{(\log n)^4}) \\ 519 & \leq \frac{2^{2q} \text{poly}(\log n)}{n(\log n)^4}. \\ 520 \end{aligned}$$

521 The second last inequality follows because there are at most 2^q possible values of such z_i ,
522 $\mathbf{N}_\ell(U_i) \geq n^{(\log n)^3} / 2$ and $\mathbf{N}_\ell(z_i) \geq \frac{1}{n(\log n)^3}$.

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

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

523

129:16 Approximate Model Counting: Is SAT Oracle More Powerful than NP Oracle?

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

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

524 By the chain rule of mutual information,

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

529 where the last inequality follows from Lemma 10.

530 Again by the chain rule of mutual information, we have

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

534

4 Conclusion

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

References

- 543 1 Teodora Baluta, Shiqi Shen, Shweta Shinde, Kuldeep S Meel, and Prateek Saxena. Quantitative
544 verification of neural networks and its security applications. In *Proceedings of the 2019 ACM*
545 *SIGSAC Conference on Computer and Communications Security*, pages 1249–1264, 2019.
- 546 2 Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of NP-witnesses using
547 an NP-oracle. *Information and Computation*, 163(2):510–526, 2000.
- 548 3 Sourav Chakraborty, Eldar Fischer, Yonatan Goldhirsh, and Arie Matsliah. On the power
549 of conditional samples in distribution testing. *SIAM J. Comput.*, 45(4):1261–1296, 2016.
550 doi:10.1137/140964199.
- 551 4 Supratik Chakraborty, Kuldeep S Meel, and Moshe Y Vardi. Algorithmic improvements in
552 approximate counting for probabilistic inference: From linear to logarithmic sat calls. Technical
553 report, 2016.
- 554 5 Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*.
555 The MIT Press and McGraw-Hill Book Company, 1989.
- 556 6 Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006.
- 557 7 Remi Delannoy and Kuldeep S Meel. On almost-uniform generation of SAT solutions: The
558 power of 3-wise independent hashing. In *Proceedings of the 37th Annual ACM/IEEE Symposium*
559 *on Logic in Computer Science*, pages 1–10, 2022.
- 560 8 Leonardo Duenas-Osorio, Kuldeep Meel, Roger Paredes, and Moshe Vardi. Counting-based
561 reliability estimation for power-transmission grids. In *Proceedings of the AAAI Conference on*
562 *Artificial Intelligence*, volume 31, 2017.

- 563 9 Matthew Fredrikson and Somesh Jha. Satisfiability modulo counting: A new approach
564 for analyzing privacy properties. In *Proceedings of the Joint Meeting of the Twenty-Third*
565 *EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual*
566 *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–10, 2014.
- 567 10 Carla P Gomes, Ashish Sabharwal, and Bart Selman. Model counting: A new strategy for
568 obtaining good bounds. In *AAAI*, volume 10, pages 1597538–1597548, 2006.
- 569 11 Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial
570 structures from a uniform distribution. *Theoretical computer science*, 43:169–188, 1986.
- 571 12 Dan Roth. On the hardness of approximate reasoning. *Artificial Intelligence*, 82(1-2):273–302,
572 1996.
- 573 13 Tian Sang, Paul Beame, and Henry A Kautz. Performing bayesian inference by weighted
574 model counting. In *AAAI*, volume 5, pages 475–481, 2005.
- 575 14 Jonathan Scarlett and Volkan Cevher. An introductory guide to Fano’s inequality with
576 applications in statistical estimation. *arXiv preprint arXiv:1901.00555*, 2019.
- 577 15 Jeanette P Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff–Hoeffding bounds for
578 applications with limited independence. *SIAM Journal on Discrete Mathematics*, 8(2):223–250,
579 1995.
- 580 16 Larry Stockmeyer. The complexity of approximate counting. In *Proceedings of the fifteenth*
581 *annual ACM symposium on Theory of computing*, pages 118–126, 1983.
- 582 17 Leslie G Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on*
583 *Computing*, 8(3):410–421, 1979.
- 584 18 Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity.
585 In *18th Annual Symposium on Foundations of Computer Science (SFCS 1977)*, pages 222–227.
586 IEEE Computer Society, 1977.

587 **A** Proof of Lemma 5

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

597 \triangleright **Claim 11.** Suppose A_i is a good strategy for $\{(A_1, s_1), \dots, (A_{i-1}, s_{i-1})\}$. Consider
598 A'_i such that $A'_i \cap \{s_1, \dots, s_{i-1}\} = A_i \cap \{s_1, \dots, s_{i-1}\}$ and $|A'_i \cap A| = |A_i \cap A|$ for atoms
599 $A \in \text{At}(A_1, \dots, A_{i-1})$. Then A'_i is also a good strategy for $\{(A_1, s_1), \dots, (A_{i-1}, s_{i-1})\}$.

600 **Proof.** We denote by \mathcal{S}_N the symmetric group acting on a set of size N . Any $\sigma \in \mathcal{S}_N$ can
601 be thought of acting on any set of size N (by thinking the elements of the set as numbered
602 $1, \dots, N$ and σ acting on the set $[N]$). For any element x in the set, we will denote by $\sigma(x)$
603 the element after the action of σ . For any $\sigma \in \mathcal{S}_N$ and set A (with $|A| = N$) we denote by
604 $\sigma(A)$ the following set $\sigma(A) := \{\sigma(x) \mid x \in A\}$.

605 Let $\sigma \in \mathcal{S}_n$ be a permutation acting on the set $\{T, F\}^n$. For any ϕ observe that
606 $|\text{sol}(\phi)| = |\sigma(\text{sol}(\phi))|$. Since any counter estimates $|\text{sol}(\phi)|$ only, we observe that if A_i
607 is a good strategy for $\{(A_1, s_1), \dots, (A_{i-1}, s_{i-1})\}$ then $\sigma(A_i)$ is also a good strategy for
608 $\{(\sigma(A_1), \sigma(s_1)), \dots, (\sigma(A_{i-1}), \sigma(s_{i-1}))\}$ for any $\sigma : \{T, F\}^n \rightarrow \{T, F\}^n$ that preserves the
609 atoms $\text{At}(A_1, \dots, A_{i-1})$ and the elements $\{s_1, \dots, s_{i-1}\}$.

129:18 Approximate Model Counting: Is SAT Oracle More Powerful than NP Oracle?

610 Since $|A'_i \cap A| = |A_i \cap A|$ for atoms $A \in At(A_1, \dots, A_{i-1})$ and $A'_i \cap \{s_1, \dots, s_{i-1}\} =$
611 $A_i \cap \{s_1, \dots, s_{i-1}\}$, there exists a σ such that $\sigma(A_j) = A_j$, $\sigma(s_j) = s_j$ for all $j \leq$
612 $i - 1$ and also $\sigma(A_i) = A'_i$. By our earlier observation, A'_i is also a good strategy for
613 $\{(A_1, s_1), \dots, (A_{i-1}, s_{i-1})\}$. ◀