On the Hardness of Probabilistic Inference Relaxations*

Supratik Chakraborty
Dept of Computer Science & Engineering
Indian Institute of Technology, Bombay

Kuldeep S. Meel
School of Computing
National University of Singapore

Moshe Y. Vardi
Department of Computer Science
Rice University

Abstract

A promising approach to probabilistic inference that has attracted recent attention exploits its reduction to a set of model counting queries. Since probabilistic inference and model counting are #P-hard, various relaxations are used in practice, with the hope that these relaxations allow efficient computation while also providing rigorous approximation guarantees.

In this paper, we show that contrary to common belief, several relaxations used for model counting and its applications (including probablistic inference) do not really lead to computational efficiency in a complexity theoretic sense. Our arguments proceed by showing the corresponding relaxed notions of counting to be computationally hard. We argue that approximate counting with multiplicative tolerance and probabilistic guarantees of correctness is the only class of relaxations that provably simplifies the problem, given access to an NP-oracle. Finally, we show that for applications that compare probability estimates with a threshold, a new notion of relaxation with gaps between low and high thresholds can be used. This new relaxation allows efficient decision making in practice, given access to an NP-oracle, while also bounding the approximation error.

1 Introduction

Decision making with uncertain data is increasingly becoming common in today's world. In this setting, techniques like probabilistic inference (Koller and Friedman 2009) and statistical hypothesis testing (Moyé 2006; Walpole et al. 1993) are crucial for informed decision making. Probabilistic graphical models, viz. Bayesian networks, Markov logic networks and the like provide an elegant formalism for representing conditional independences between variables in a system (Koller and Friedman 2009). Probabilistic inference on graphical models gives algorithmic techniques to determine the posteriori probability of an event, given observations or evidence. It is well known that probabilistic inference can be reduced to a set of (literal-weighted) model counting queries[1] (Roth 1996; Chavira and Darwiche 2008). Recent advances in model counting techniques (Chakraborty, Meel, and Vardi 2013b; Chakraborty et al. 2014; Ermon et al. 2013; Chakraborty, Meel, and Vardi 2016; Meel et al. 2016) therefore yield a promising approach for probabilistic inference that is complementary to variational methods (Wainwright and Jordan 2008), loopy belief propagation (Murphy, Weiss, and Jordan 1999), Markov Chain Monte Carlo (MCMC)-based techniques (Neal 1993) and the like. Indeed, the model counting formulation has recently been hailed as a promising *assembly language* for inferencing in probabilistic graphical models.

Unfortunately, model counting and probabilistic inference are computationally hard (Valiant 1979; Roth 1996). Hence it is unlikely that efficient algorithms exist for exact probabilistic inference, and various approximations and relaxations are used in practice (viz. (Neal 1993; Sarkhel et al. 2016; Fink, Huang, and Olteanu 2013; Wexler and Meek 2008; Gordon et al. 2014)). The widespread use of relaxations is premised on the belief that such relaxations offer computational simplicity vis-a-vis exact inference, while being reasonably accurate in practice. Indeed, informal arguments and experimental measurements on benchmarks are often presented as "proof" of effectiveness of these relaxations. While this may be acceptable in several contexts, it is clearly not in others where the consequence of machine-computed probabilistic inference can be dramatic, viz. clinical diagnosis (Seixas, B. Zadrozny, and Saade 2014) or military decision support systems (Antonucci et al. 2013). In such cases, it is imperative that we are cognizant of worst-case bounds on approximation errors, when choosing a relaxation based on the belief that it offers computational simplicity.

As machine intelligence affects increasing aspects of our lives, some with far-reaching consequences, a deeper understanding of such precision-complexity tradeoff provided by different relaxations is almost a necessity. The appropriateness of the tradeoff offered by a relaxation may differ from one context to another; hence a data scientist must know where on the tradeoff curve a specific relaxation lies, before using it in an application. For example, Bayesian network models have been studied for diagnosis of serious diseases (Seixas, B. Zadrozny, and Saade 2014; Onisko, Druzdzel, and Wasyluk 1999; Cruz-Ramirez et al. 2007). Diagnosing an individual as having (or not having) a disease based on such complex probabilistic models, and asking "what-if" questions with respect to the treatment regime

*The author list has been sorted alphabetically by last name; this should not be used to determine the extent of authors' contributions.

[1]Literal-weighted model counting, in turn, can be reduced to (unweighted) model counting (Chakraborty et al. 2015).

requires choosing a relaxation that does not permit large approximation errors, even if the computational time required is moderately high. The price of mis-classification (a disease-free person being labeled as diseased or vice versa) due to large approximation errors can indeed be very high in such cases. On the other hand, a biometric access-control system must decide in real-time whether to allow an individual access to a facility, depending on the percentage match of the stored fingerprint data with the noisy output of a fingerprint scanner. Depending on the facility, occasional denial of access to authorized individuals may not have significant ramifications; hence relaxations that are computationally efficient may be preferred in such applications. Thus, depending on the application, relaxations with different accuracy/efficiency tradeoffs may be appropriate, which is why it is critical to understand this tradeoff.

The literature contains several approximation techniques that buy efficiency by sacrificing guarantees of bounded error, but often work well in practice. Examples include loopy belief propagation (Murphy, Weiss, and Jordan 1999; Ihler, III, and Willsky 2005) and algorithms that use MCMC sampling but truncate the steps well before theoretical (exponential) bounds guaranteeing convergence are reached (Neal 1993). We exclude such techniques from the scope of our study, since approximations without guarantees of bounded errors do not permit a fair comparison of accuracy vs complexity. Furthermore, performing probabilistic inference with *no* error bound guarantees is simply inappropriate in many contexts, e.g., for the diagnosis of a life-threatening disease with a high-risk therapy.

Yet another category of relaxations provide guarantees of bounded approximation errors, and are commonly believed to be computationally more efficient than exact techniques. In this paper, we undertake a rigorous study of the complexity-accuracy tradeoffs of a variety of such relaxation techniques. Surprisingly, we find that most relaxations used in the probabilistic-inference (via model counting) literature actually do not provide any computational simplification from a complexity theoretic perspective. Our arguments proceed by showing the corresponding relaxations of model counting to be computationally hard. The inter-reducibility of probabilistic inference and model counting then establishes the hardness of the relaxed versions of inference as well.

Recently, an excellent survey of parameterized complexity of approximate Bayesian inference techniques has been provided by Kwisthout (Kwisthout 2018). In this work, Kwisthout considers only (randomized) polynomial time algorithms as "efficient", and shows that most variants of inferencing cannot be solved efficiently, unless constraints are imposed on several parameters. In view of the spectacular progress made in propositional satisfiability (SAT) solving technology over the past three decades, and inspired by the success of randomized counting and inferencing algorithms that use SAT solvers as oracles/black boxes, *we expand the notion of practically "efficient" algorithms to include the complexity class* $\mathsf{BPP}^{\mathsf{NP}}$. Informally, this class admits algorithms that run in polynomial time modulo calls to a black-box SAT solver, while guaranteeing a low probability of error. With this notion of efficiency, not all hope is lost in the search for meaningful relaxations. We propose a new relaxation suitable for settings like statistical hypothesis testing where a probability measure is compared with a threshold to arrive at a decision. We show that by introducing a gap between the low and high thresholds used for comparison, we can provably simplify the decision making problem, while also providing strong guarantees of approximation. A salient feature of our study is that our arguments largely employ standard tools and techniques. This strengthens the case for demanding rigorous complexity-accuracy tradeoff analyses (as opposed to informal arguments and experimental observations) when new relaxations are proposed by researchers.

The remainder of the paper is organized as follows. We discuss notations and preliminaries in Section 2, and present a survey of related work in Section 3.We then discuss hardness of counting relaxations in Section 4, where we show that several relaxations used in the probabilistic-inference literature actually do not provide any computational simplification from a complexity theoretic perspective. In Section 5, we propose a new computationally efficient relaxation with provable error bounds for use in applications that compare probability estimates with a threshold. Finally, we conclude the paper in Section 6

2 Preliminaries

We denote a propositional constraint or formula by φ and call the set of variables in φ the *support* of φ, or $\mathsf{Sup}(\varphi)$. A satisfying assignment or model of φ is an assignment of truth values to all variables in $\mathsf{Sup}(\varphi)$ such that φ evaluates to True. We use $\mathsf{Sol}(\varphi)$ to denote the set of all models of φ. The model counting problem can now be stated as follows.

1. ***Model Counting*** **(**MC**):** Given φ, find $|\mathsf{Sol}(\varphi)|$.

Let Z be a random variable with a specified probability distribution. Let $\bowtie \in \{=, \leq, <, >, \geq\}$ be a relational operator and let a be a value in the range of Z. We use $\mathsf{Pr}[\, Z \bowtie a \,]$ to denote the probability that $Z \bowtie a$ holds. We also use $\mathsf{E}[Z]$ and $\mathsf{Var}[Z]$ to denote the expectation and variance, respectively, of Z.

We study several relaxations of the model counting problem. Broadly, we divide these relaxations into three categories: additive, multiplicative and threshold. We now discuss each of these categories in detail. In the following, c denotes an estimate of $|\mathsf{Sol}(\varphi)|$, ε represents an additive tolerance bound, ρ represents a multiplicative tolerance factor, and $1 - \delta$ $(0 < \delta \leq 1)$ represents a probabilistic confidence bound.

2.1 Additive relaxations

Relaxations in this category focus on approximating the model count within an additive error bound provided as an input. Two sub-categories of additive relaxations can be identified, depending on whether the approximation is deterministic or randomized.

2. **Additively-Approximate Model Counting** (AAMC): Given φ and ε, find a deterministic estimate c (≥ 0) such that $(|\mathsf{Sol}(\varphi)| - \varepsilon) \leq c \leq (|\mathsf{Sol}(\varphi)| + \varepsilon)$.

3. ***Probably Additively-Approximate Model Counting*** **(PAAMC):** Given φ, ε and δ, find a random estimate c (≥ 0) such that $\mathsf{Pr}\left[\ (|\mathsf{Sol}(\varphi)| - \varepsilon) \ \leq\ c \leq (|\mathsf{Sol}(\varphi)| + \varepsilon)\ \right] \geq 1 - \delta$

AAMC and PAAMC relaxations have been used in different contexts in the literature, viz. (Sarkhel et al. 2016) and (Fink, Huang, and Olteanu 2013) among others.

2.2 Multiplicative relaxations

In this category, the relaxations focus on obtaining a multiplicative approximation of the count. Once again, two subcategories can be identified, depending on whether the approximation is deterministic or randomized.

4. ***Multiplicatively-Approximate Model Counting*** **(MAMC):** Given φ and ρ, find a deterministic estimate c (≥ 0) such that $\frac{|\mathsf{Sol}(\varphi)|}{1+\rho} \ \leq\ c \ \leq\ |\mathsf{Sol}(\varphi)| \cdot (1 + \rho)$
5. ***Probably Approximately Correct Model Counting*** **(PACMC):** Given φ, ε and δ, find a random estimate c (≥ 0) such that $\mathsf{Pr}\left[\ \frac{|\mathsf{Sol}(\varphi)|}{1+\rho} \ \leq\ c \ \leq\ |\mathsf{Sol}(\varphi)| \cdot (1 + \rho)\ \right] \geq 1 - \delta$

MAMC relaxations are used in (Fink, Huang, and Olteanu 2013; Wexler and Meek 2008) among others, while PACMC relaxations are used in several recent work, viz. (Zhu and Ermon 2015; Ermon et al. 2013; Chakraborty, Meel, and Vardi 2013b; 2016; Chakraborty et al. 2016).

2.3 Threshold relaxations

These relaxations are probabilistic in nature and seek to inquire the probability of an estimate (or threshold) satisfying a relation with the exact count. The relation is denoted by an operator in $\{<, \leq, >, \geq\}$.

6. ***Probabilistic Model Thresholding*** **(PMT):**. Given φ, c, δ, and a relational operator $\bowtie \in \{<, \leq, >, \geq\}$, output a random variable $Y \in \{0, 1\}$ such that $\mathsf{Pr}\left[\ Y = 1\ \right] \geq 1 - \delta$ if $|\mathsf{Sol}(\varphi)| \bowtie c$ holds, and $\mathsf{Pr}\left[\ Y = 1\ \right] \leq \delta$ otherwise.

PMT is used in different applications, viz. statistical hypothesis testing in diverse contexts (Moyé 2006; King, Rosopa, and Minium 2010; Zongming 2009), reasoning in probabilistic programming frameworks (Gordon et al. 2014; Bornholt, Mytkowicz, and McKinley 2014) and the like.

2.4 Complexity classes

We briefly review notions from computational complexity theory that are of relevance to our work. For a more detailed exposition, the reader is referred to (Arora and Barak 2009).

A *probabilistic Turing machine* is a non-deterministic Turing machine that can choose between alternative transitions at every step based on a probability distribution. For our purposes, the probability distribution is assumed to be uniform. A decision problem is said to be in the complexity class BPP if there exists a probabilistic Turing machine M such that for all instances of the problem of size n, the machine M runs in time polynomial in n, and provides an answer that is correct with probablity strictly greater than 0.5 (the exact value is unimportant). For a problem in BPP, the probability of error can be reduced exponentially by running the probabilistic Turing machine multiple times independently and by accepting the majority decision output by these runs. Hence, BPP represents a large class of problems that admit efficient practical algorithms. It is easy to see that $\mathsf{P} \subseteq \mathsf{BPP}$.

An *oracle machine* is an abstract computational machine that can be viewed as a (probabilistic) Turing machine equipped with a black-box oracle that can be queried at any step, and that answers each such query correctly in one unit of time. If $\mathcal{C}_1$ and $\mathcal{C}_2$ represent two complexity classes, $\mathcal{C}_1^{\mathcal{C}_2}$ is the complexity class representing problems that can be solved by a (probabilistic) Turing machine for a problem in $\mathcal{C}_1$ if it is equipped with an oracle for a problem in $\mathcal{C}_2$.

The complexity classes P and NP are well-known and not described further here. coNP is the class of decision problems, the complement of which are in NP. For example, propositional satisfiability is in NP, while propositional unsatisfiability is in coNP. The polynomial hierarchy generalizes the classes P, NP and coNP to oracle machines. Specifically, we define two (related) hierarchies Σ_i^{P} and Π_i^{P} as follows: $\Sigma_0^{\mathsf{P}} = \Pi_0^{\mathsf{P}} = \mathsf{P}$, and for all $i \geq 0$, $\Sigma_{i+1}^{\mathsf{P}} = \mathsf{NP}^{\Sigma_i^{\mathsf{P}}}$ and $\Pi_{i+1}^{\mathsf{P}} = \mathsf{coNP}^{\Sigma_i^{\mathsf{P}}}$. Thus, $\mathsf{NP} = \Sigma_1^{\mathsf{P}}$ and $\mathsf{coNP} = \Pi_1^{\mathsf{P}}$. It is also known that Σ_i^{P} (resp. Π_i^{P}) represents exactly the set of decision problems that can be polynomially reduced to checking the satisfiability of quantified Boolean formulas (QBF) of the form $Q_1 x_1 \ldots Q_n x_n \varphi(x_1, \ldots, x_n)$, where each $Q_i \in \{\exists, \forall\}$ and there are i levels of quantifier alternations starting with an $\exists$ (resp. $\forall$) block. The polynomial hierarchy PH is the union of all classes Σ_i^{P} and Π_i^{P} for $i \geq 0$. It is a long-standing open question whether the classes $\Sigma_{i+1}^{\mathsf{P}}$ and Π_{i+1}^{P} strictly generalize Σ_i^{P} and Π_i^{P} respectively, for any $i \geq 0$.

Given an instance of a problem in NP, the corresponding counting problem asks how many accepting paths exist in the non-deterministic Turing machine that solves the problem. The complexity class #P is defined to be the set of counting problems associated with all decision problems in NP. By a result of Toda (Toda 1989)), one call to #P is sufficient to solve every problem in the polynomial hierarchy; formally, $\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}}$. It is therefore widely believed that problems in #P are significantly harder than those in PH.

A complexity class of immense interest to us in this paper is $\mathsf{BPP}^{\mathsf{NP}}$. A seminal result of Sipser, Gács and Lautemann states that $\mathsf{BPP} \subseteq \Sigma_2^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}}$ (Sipser 1983; Lautemann 1983). Therefore, $\mathsf{BPP}^{\mathsf{NP}}$ lies in $\Sigma_3^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}}$ – fairly low in the polynomial hierachy.

2.5 Remark on weighted model counting

While the MC problem introduced earlier simply counts the number of solutions of φ, one can also assign a non-negative weight to each assignment of variables in $\mathsf{Sup}(\varphi)$ and ask for the total weight of all satisfying assignments of φ. This is called *weighted model counting*, also denoted as WMC. Note that MC is a special case of WMC, where the weight of every assignment is 1. Therefore, hardness results for MC, such as the ones presented in Section 4, immediately lift to WMC as well. The reduction of probabilistic inference to model count-

ing queries (Roth 1996; Chavira and Darwiche 2008) uses a specific form of WMC, known as literal-weighted model counting. In literal-weighted WMC, weights are assigned to literals, and the weight of an assignment is the product of weights of its literals. Chakraborty et al (2015) showed that literal-weighted WMC can be reduced to MC in time polynomial in the number of bits used to represent weights of literals. Therefore, (polynomial or higher) complexity upper bounds of model-counting relaxations, such as the one presented in Section 5, also apply to literal-weighted WMC with binary representation of literal weights.

3 Related work

Probabilistic inference and model counting are known to be #P-hard (Roth 1996; Chavira and Darwiche 2008); in addition, approximating probabilistic inference within given error bounds is also NP-hard (Dagum and Luby 1993). Therefore, several approximate inferencing techniques have been developed over the past few decades. The reader is referred to (Kwisthout 2018) for an excellent survey of parameterized complexity of different approximate Bayesian inference techniques.

In statistical hypothesis testing (Walpole et al. 1993; Moyé 2006; King, Rosopa, and Minium 2010; Zongming 2009), we have a null hypothesis and an alternative hypothesis. Based on the system's characteristics, we must then estimate the probability (p-value) of making an observation that is more extreme in the direction of the alternative hypothesis than is actually observed, assuming the null hypothesis was true. If the p-value thus estimated is below a threshold, we reject the null hypothesis and accept the alternative hypothesis; otherwise, the null hypothesis cannot be rejected. Therefore, the central problem is to determine if the conditional probability of a hypothesized event is below (or above) a threshold (i.e., PMT). The same problem arises in the analysis of probabilistic programs (Gordon et al. 2014; Bornholt, Mytkowicz, and McKinley 2014), where we must determine whether a conditional statement must execute based on a randomly sampled value. The complexity-accuracy tradeoff of relaxation techniques that estimate if the probability of an event exceeds (or falls below) a threshold is therefore useful for these applications.

In (Sarkhel et al. 2016), a technique for training Markov logic networks is presented that uses approximate counting to "simplify" the most computationally expensive step of estimating the number of groundings of a first-order logic formula that evaluates to True given a truth assignment of all random variables. The authors assume access to an approximate counting procedure that returns the model count of a formula to within a specified additive tolerance ϵ. They presume implicitly, and perhaps naturally, that counting models within a specified additive tolerance (i.e., AAMC and PAAMC) is computationally simpler than computing the exact count. Unfortunately, as we show later, this presumption is false for values of ϵ below a threshold. Thus, although performance improvements have been reported in (Sarkhel et al. 2016), using approximate counts with additive tolerance doesn't really yield efficient algorithms in the worst-case.

Similarly, in (Fink, Huang, and Olteanu 2013), a deterministic iterative anytime-approximate algorithm for relational query evaluation in probabilistic databases is presented. Instead of computing exact probabilities of results to a query, the proposed algorithm achieves efficiency by reporting probabilities that are within a specified tolerance ϵ (both additive and multiplicative tolerances are supported). Once again, the implicit premise is that deterministically computing approximate counts with specified additive/multiplicative tolerances (i.e., AAMC and MAMC) is computationally simpler than computing the exact count. We show that none of these relaxations yield efficient algorithms in the worst-case, even if we have access to a practically efficient SAT solver.

In (Wexler and Meek 2008), a multiplicative approximation scheme (MAS) for various probabilistic inferencing techniques is proposed. At the heart of this scheme is the notion of ϵ-decomposition, which allows the desired probability estimate to be calculated within a specified multiplicative tolerance. The authors show by means of experiments that this leads to significant performance improvements over existing inferencing techniques on a set of benchmarks. Our results, however, show that this cannot lead to efficient algorithms in the worst-case, even with access to a practically efficient SAT solver.

4 Hardness of Counting Relaxations

In his seminal work, Valiant defined the class #P and showed that MC is #P-complete (Valiant 1979). Since $\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}}$, it is unlikely that there is an efficient algorithm for any problem in #P. Therefore, MC is among the computationally hardest variants of counting in our list. At the other end of the spectrum, MAMC is known to be solvable in polynomial time with access to a Σ_2^{P} oracle (Stockmeyer 1983). This would yield an efficient algorithm if we had access to an efficient 2QBF solver that decides quantified Boolean formulas with $\exists^*\forall^*$ quantifier prefix. Unfortunately, the gap between the empirical performance of 2QBF and SAT solvers is still large; so we don't yet have any practically efficient algorithm for MAMC.

Building on (Stockmeyer 1983), Jerrum, Valiant and Vazirani (Jerrum, Valiant, and Vazirani 1986) showed that PACMC can be solved in probabilistic polynomial time with access to an NP oracle. Given the impressive progress made over the past two decades in the development of satisfiability solvers for propositional formulas, there has been significant recent interest in developing practical algorithms for problems that can be solved with polynomially many calls to an NP oracle. For example, a practically efficient algorithm for PACMC was proposed in (Chakraborty, Meel, and Vardi 2016), where the number of invocations of the NP oracle is restricted to be in $\mathcal{O}\left(\frac{1}{\varepsilon^2} \cdot \log(1/\delta) \cdot \log n\right)$.

In this section, we investigate whether the various relaxations discussed in Section 2 are amenable to efficient algorithms that have access to an NP oracle. Specifically, we show that MAMC and PACMC are the only counting relaxations in our list that are known to lie in the polynomial hierarchy. Of these, MAMC requires a 2QBF oracle, while PACMC can be solved using an NP oracle. All other relaxations are either

as hard as MC, or lie beyond the polynomial hierarchy unless the hierarchy itself collapses.

The hardness results imply that it is unlikely that efficient algorithms exist for these relaxations, even if we had access to efficient QBF solvers. This separates the class of relaxations that can benefit from the significant advances made in practical SAT solving from those that cannot. In situations where one must choose between alternate probabilistic relaxations, our results can provide a rigorous basis to help guide the choice depending on the context.

In the remainder of this section, we first investigate the hardness of PMT, and then use this result to establish the hardness of AAMC and PAAMC.

4.1 Hardness of PMT

In order to show the hardness of PMT, we first show that a related problem (that isn't really used in any application we are aware of) is hard. Specifically, given φ, c and δ, suppose we wish to output a random variable $Y \in \{0,1\}$ such that $\mathsf{Pr}[\, Y = 1 \,] \geq 1 - \delta$ if $|\mathsf{Sol}(\varphi)| = c$, and $\mathsf{Pr}[\, Y = 1 \,] \leq \delta$ otherwise. For notational convenience, we call this problem PMC (for probabilistic model counting). We first show that a PMC-oracle is as powerful as a #P-oracle, when used with an algorithm for a problem in BPP. Subsequently, we use this result to prove that PMT lies beyond PH unless PH collapses. We use Turing reductions in the proofs below.

Lemma 1. $\mathsf{BPP}^{\#\mathsf{P}} = \mathsf{BPP}^{\mathsf{PMC}}$

Proof. Since every query to a PMC-oracle can be answered by a #P-oracle, it is easy to see that $\mathsf{BPP}^{\mathsf{PMC}} \subseteq \mathsf{BPP}^{\#\mathsf{P}}$.

To show inclusion in the other direction, consider a language L in $\mathsf{BPP}^{\#\mathsf{P}}$. Choosing the probability threshold (> 0.5) to be $2/3$, there exists a probabilistic Turing machine M, and a polynomial $p(\cdot)$ such that given a string x, the following hold:

- M takes $p(|x|)$ steps to run, where one or more of these steps make a query to a #P-oracle.
- M outputs 1 with probability $\geq 2/3$ if $x \in L$.
- M outputs 1 with probability $\leq 1/3$ if $x \notin L$.

Now consider a probabilistic Turing machine M' that mimics the steps of M on x, except when making a query to a #P-oracle. Every query $\mathsf{MC}(\varphi)$ to the #P-oracle in M is replaced by r independent queries $\mathsf{PMC}^1(\varphi,\delta), \ldots \mathsf{PMC}^r(\varphi,\delta)$ to a PMC-oracle, where $r = \lceil 4.74 + 1.36 \log p(|x|) \rceil$, and $\delta = \frac{1}{k}$ for a suitably large constant k, viz. 10. After the r queries to the PMC-oracle are made, M' computes the majority of the resulting estimates, breaking ties arbitrarily if needed, and proceeds exactly as in M. Since each oracle query takes one unit of time, the machine M' takes $\mathcal{O}(p(|x|) \log p(|x|))$ steps to run on input x. Hence it is a polynomial-time Turing machine relative to a PMC oracle.

Suppose the estimates returned by the independent oracle queries $\mathsf{PMC}^1(\varphi,\delta), \ldots \mathsf{PMC}^r(\varphi,\delta)$ are $c_1, \ldots c_r$, and let c be the majority estimate. We wish to bound the probability that c is incorrect. Clearly, if c is incorrect, at least $r/2$ of the estimates must be incorrect. For notational convenience, let $\eta(n,m,p)$ denote $\Sigma_{j=m}^{n} \binom{n}{j} p^j (1-p)^{n-j}$, where $0 \leq p \leq 1$. Therefore, $\mathsf{Pr}[\, c \text{ is incorrect} \,] = \eta(r, r/2, \delta)$. Since $\delta = \frac{1}{k} = 0.1$ in our case, we have $\mathsf{Pr}[\, c \text{ is incorrect} \,] \leq \eta(r, r/2, 0.1) \leq \frac{9}{8}.(0.6)^r$. The last inequality follows from bounds of the binomial coefficient and from summation of a geometric series. Since $r \geq 4.74 + 1.36 \log p(|x|)$, it is easy to see that $\mathsf{Pr}[\, c \text{ is incorrect} \,] \leq \frac{1}{10 \cdot p(|x|)}$.

Therefore, the probability that the machine M' gives a correct answer is $\geq \frac{2}{3} \times (\mathsf{Pr}[\, c \text{ is correct} \,])^{p(|x|)}$. This is at least as large as $\frac{2}{3} \cdot (1 - \frac{1}{10.p(|x|)})^{p(|x|)} \geq \frac{2}{3} \cdot (1 - \frac{1}{10}) \geq 0.59$.

Thus, for every x, the machine M' runs in time $\mathcal{O}(p(|x|) \log p(|x|))$ relative to a PMC oracle, and returns a correct answer with probability ≥ 0.59. □

Theorem 2. $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{PMT}}$*, and hence* PMT *lies beyond* PH *unless* PH *collapses.*

Proof. We first show that $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{PMC}}$. From Toda's theorem (Toda 1989), we know that $\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{BPP}^{\#\mathsf{P}}$. From Lemma 1, it then follows that $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{PMC}}$.

Next, we reduce an arbitrary instance of PMC to polynomially many instances of PMT. Specifically, consider an instance $\mathsf{PMC}(\varphi,\delta)$, with n being $|\mathsf{Sup}(\varphi)|$. Since $0 \leq |\mathsf{Sol}(\varphi)| \leq 2^n$, we can use binary search with at most n invocations of PMT (using any relational operator $\bowtie \in \{\leq, <, \geq, >\}$) to determine $|\mathsf{Sol}(\varphi)|$. Each such invocation of PMT is of the form $\mathsf{PMT}(\varphi, c', \delta')$, where c' is determined by the current status of the binary search, and $\delta' = \frac{\delta}{n}$. Let c^* be the estimate of the model count obtained by this binary search. It is then easy to see that $\mathsf{Pr}[\, |\mathsf{Sol}(\varphi)| = c^* \,] \geq (1-\delta')^n \geq 1 - n.\delta' = 1 - \delta$.

An invocation of $\mathsf{PMC}(\varphi, c', \frac{\delta}{n})$ can be simulated by taking the majority answer from $\mathcal{O}(\log(n/\delta))$ independent runs of $\mathsf{PMT}(\varphi, c', \beta)$, where β is any positive constant less than 0.5. Thus, every instance of $\mathsf{PMC}(\varphi,\delta)$ can be reduced to $\mathcal{O}(n.\log(n/\delta))$ instances of $\mathsf{PMT}(\varphi, c', \beta)$, where c' varies for different instances, as required by the binary search.

We have already shown above that $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{PMC}}$. It now follows that $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{PMT}}$. If possible, let PMT be in PH. Then, there exists a $k \geq 0$ such that PMT is in Σ_k^P. Since $\mathsf{BPP} \subseteq \Sigma_2^P$, this implies that $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{PMT}} \subseteq \Sigma_{k+2}^P$. Thus, if PMT is in PH, the whole of PH collapses to a finite hierarchy. □

4.2 Hardness of AAMC and PAAMC

We now show that AAMC is as hard as MC, and PAAMC is as hard as PMC. Therefore it is unlikely that efficient algorithms exist for these problems. Recall that an instance of AAMC has a specified additive tolerance $\varepsilon \geq 0$. If $\varepsilon = 0$, AAMC trivially reduces to MC. We show below that AAMC reduces to MC even for exponentially larger values of ε. We believe the following theorem to be a folklore result (de Campos, Stamoulis, and Weyland 2017) but provide the proof for completeness. Again, we rely on Turing reductions to prove the next two theorems.

Theorem 3. AAMC *with* $\varepsilon \leq 2^{\frac{n}{2}-2}$ *is* #P*-complete, where* n *denotes the size of the support of the input formula.*

Proof. It is easy to see that AAMC is in #P, since $\mathsf{AAMC}(\varphi, \varepsilon)$ can be solved by computing $|\mathsf{Sol}(\varphi)|$. To show that AAMC is #P-hard, we reduce MC to AAMC.

Let $\mathsf{MC}(\varphi)$ be an arbitrary instance of MC, and let n be $|\mathsf{Sup}(\varphi)|$. Let x be a variable not in $\mathsf{Sup}(\varphi)$. Construct the formula φ' defined as $(x \Rightarrow \varphi)$, and let φ'' be a version of φ' in which all variables (including x) have been renamed to fresh variables. Finally, let ψ be the formula $\varphi' \wedge \varphi''$.

It is easy to see that φ' and φ'' are formulas with disjoint supports of size $n+1$ each, while ψ is a formula with support of size $2(n+1)$. Moreover, $|\mathsf{Sol}(\varphi')| = |\mathsf{Sol}(\varphi'')| = |\mathsf{Sol}(\varphi)| + 2^n \geq 2^n$, and $|\mathsf{Sol}(\psi)| = |\mathsf{Sol}(\varphi')|^2$.

We now invoke $\mathsf{AAMC}(\psi, \varepsilon)$, where ε is a non-negative constant $\leq 2^{n-1}$, represented using $\mathcal{O}(n)$ bits. Clearly, the input to AAMC is of size $\mathcal{O}(|\varphi|)$. Note also that $n - 1 = \frac{|\mathsf{Sup}(\psi)|}{2} - 2$, thereby satisfying the condition for the additive tolerance in the statement of the theorem. Suppose the estimate returned by an algorithm for this instance of AAMC is c. By definition, we have $(|\mathsf{Sol}(\psi)| - \varepsilon) \leq c \leq (|\mathsf{Sol}(\psi)| + \varepsilon)$. Since $|\mathsf{Sol}(\psi)| = |\mathsf{Sol}(\varphi')|^2$ and since $\varepsilon \leq 2^{n-1} \leq |\mathsf{Sol}(\varphi')|/2$, we also have $(|\mathsf{Sol}(\psi)| - \varepsilon) \geq |\mathsf{Sol}(\varphi')|^2 - |\mathsf{Sol}(\varphi')|/2 = |\mathsf{Sol}(\varphi')| \cdot (|\mathsf{Sol}(\varphi')| - 1/2) > (|\mathsf{Sol}(\varphi')| - 1/2)^2$, and $(|\mathsf{Sol}(\psi)| + \varepsilon) \leq |\mathsf{Sol}(\varphi')|^2 + |\mathsf{Sol}(\varphi')|/2 = |\mathsf{Sol}(\varphi')| \cdot (|\mathsf{Sol}(\varphi')| + 1/2) < (|\mathsf{Sol}(\varphi')| + 1/2)^2$. Therefore, $\left(|\mathsf{Sol}(\varphi')| - \frac{1}{2}\right)^2 < c < \left(|\mathsf{Sol}(\varphi')| + \frac{1}{2}\right)^2$. It follows that $(|\mathsf{Sol}(\varphi')| - 1/2) < \sqrt{c} < (|\mathsf{Sol}(\varphi')| + 1/2)$, Hence, rounding $\sqrt{c}$ to the nearest integer gives $|\mathsf{Sol}(\varphi')|$. The desired count, $\mathsf{Sol}(\varphi)$, is then obtained as $\sqrt{c} - 2^n$. □

Theorem 4. *Let* $\mathsf{PAAMC}[\tau]$ *denote* PAAMC *with* $\varepsilon \leq \tau$. *Then* $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{PAAMC}[2^{n/2}-2]}$, *where* n *denotes the size of the support of the input formula for* PAAMC. *Hence, unless* PH *collapses,* $\mathsf{PAAMC}[2^{n/2} - 2]$ *lies beyond* PH,

Proof. The proof of Theorem 3 can be easily adapted to give a polynomial time reduction from PMC to PAAMC with $\varepsilon \leq 2^{\frac{n}{2}-2}$. The result then immediately follows from Theorem 2.

□

5 Balancing Accuracy and Complexity

The discussion in the previous section tells us that *not all relaxations of model counting admit the existence of efficient algorithms, even if they have access to practically efficient propositional satisfiability solvers.* Hence, it is important to carefully balance accuracy and complexity when choosing relaxations. The literature contains several studies (viz. (Sarkhel et al. 2016; Fink, Huang, and Olteanu 2013; Wexler and Meek 2008) among others) that used relaxations as a means to avoid the complexity of exact counting and reported increased efficiency, although our analysis shows that the worst-case complexity of the relaxed problem continued to be high. We believe there are two possible explanations: (1) The soundness guarantees on relaxations were not crucial for the underlying applications, and therefore, the relaxations were good enough even though they may not have provided provably sound answers. (2) The specific problems of interest for those relaxations belonged to a subclass that was amenable to efficient techniques. We believe that in the context of our results, further investigation is warranted to better explain the success of relaxations with high worst-case complexity that have nevertheless been used in the literature.

It is natural to ask if there are relaxations that allow a fine balance between theoretical approximation guarantees and complexity. In recent years, PACMC has received a lot of attention, and algorithms that scale to formulas with almost a million variables in their support, while providing PAC-style guarantees (Valiant 2013), have been designed. We believe algorithms such as those in (Chakraborty, Meel, and Vardi 2013a; Ermon et al. 2013; Chakraborty, Meel, and Vardi 2016; Meel 2014) represent a "sweet spot" in the quest for balancing relaxations of counting requirements and development of practical algorithms. In this section, we present another promising "sweet spot" that is obtained by relaxing the requirements of PMT. Since PMT is widely used in several applications (see Sec 2.3), this contributes to the few known algorithms that provide guarantees required for important applications, and also scale to practical problems.

5.1 A Relaxation of PMT

Recall from Section 2 that if Y is a random variable denoting the output of $\mathsf{PMT}(\varphi, c, \bowtie, \delta)$, then we require the following: (i) if $\mathsf{Sol}(\varphi) \bowtie c$, then $\mathsf{Pr}[\, Y = 1 \,] \geq 1 - \delta$, and (ii) if $\mathsf{Sol}(\varphi) \not\bowtie c$, then $\mathsf{Pr}[\, Y = 0 \,] \geq 1 - \delta$. Note that this definition requires the random variable Y to change from being 1 with high probability to being 0 with high probablity, when c changes from $\mathsf{Sol}(\varphi) - \varepsilon$ to $\mathsf{Sol}(\varepsilon) + \varepsilon$, even for vanishingly small values of ε. Intuitively, this requires an algorithm for PMT to be aware of $\mathsf{Sol}(\varphi)$ with a high degree of certainty, making it computationally hard. The relaxation we propose introduces a gap in the ranges of c for which Y is required to be 1 (resp. 0) with high probability. We formally state this relaxed version of PMT below.

Definition 1 (α - PMT)**.** *Given* φ, *an integer* $c \geq 0$, *a fraction* α $(0 \leq \alpha < 1)$, *and* δ $(0 < \delta \leq 1)$, *output a random variable* $Y \in \{0, 1\}$ *such that*

- *If* $|\mathsf{Sol}(\varphi)| \geq c$, *then* $\mathsf{Pr}[\, Y = 1 \,] \geq 1 - \delta$
- *If* $|\mathsf{Sol}(\varphi)| \leq \alpha \cdot c$, *then* $\mathsf{Pr}[\, Y = 0 \,] \geq 1 - \delta$.

Note that Definition 1 says nothing about the random variable Y when $\alpha \cdot c < \mathsf{Sol}(\varphi) < c$. For the subsequent discussion, we say that an algorithm for relaxed PMT reports an incorrect answer if it either outputs $Y = 0$ when $|\mathsf{Sol}(\varphi)| \geq c$ or outputs $Y = 1$ when $|\mathsf{Sol}(\varphi) \leq \alpha \cdot c$. At this point, it is important to discuss the relation of α - PMT with PACMC. Recall that PACMC effectively conjoins two inequalities, i.e. whether $c \geq \frac{|\mathsf{Sol}(()\phi)|}{1+\varepsilon}$ and $c \leq (1+\varepsilon)(|\mathsf{Sol}(()\phi)|)$, and can be solved efficiently with the help of an NP oracle. Our definition of α - PMT separates out the two inequalities in the definition of PACMC. As we show below, we can provide an answer with high confidence efficiently (with the help of an NP oracle) even in this case.

Theorem 5. *For* $\alpha < 1/4$, α - $\mathsf{PMT} \in BPP^{NP}$ *and can be solved using* $\mathcal{O}\left(\log \frac{1}{\delta}\right)$ *calls to an* NP*-oracle.*

Proof. Towards a randomized algorithm for α-PMT, we propose using a 2-universal XOR-based hash function $h : \{0,1\}^n \to \{0,1\}^m$, comprised of $m = \lfloor \log_2 c \rfloor$ random parity constraints (Carter and Wegman 1977). Let $\beta \in \{0,1\}^m$ be a randomly chosen cell, and let $\mathsf{Sol}_{\varphi,h,\beta}$ denote the set of solutions of φ that are hashed to the cell β by h, i.e. $\mathsf{Sol}_{\varphi,h,\beta} = \{w \in \{0,1\}^n \mid \varphi(w) = \mathsf{True} \text{ and } h(w) = \beta\}$. Let Z denote the random variable $|\mathsf{Sol}_{\varphi,h,\beta}|$. We propose to output $Y = 1$ if $Z > 0$, and $Y = 0$ otherwise. We now analyze the probability of the above algorithm reporting an incorrect answer. It is easy to see that $\mathsf{E}[Z] = |\mathsf{Sol}(\varphi)|/2^m$. Since $m = \lfloor \log_2 c \rfloor$, we have $c/2 < 2^m \leq c$. Furthermore, it can be shown that $\mathsf{Var}[Z] < \mathsf{E}[Z]$. We have two cases to consider:

- If $|\mathsf{Sol}(\varphi)| \geq c$, then since $m = \lfloor \log_2 c \rfloor$, we have $\mathsf{E}[Z] = |\mathsf{Sol}(\varphi)|/2^m \geq 1$.
  By Paley-Zygmund inequality, we also know that for $0 \leq \theta \leq 1$, $\mathsf{Pr}[\, Z > \theta.\mathsf{E}[Z]\,] > \frac{1}{1+\frac{\mathsf{Var}[Z]}{(1-\theta)^2.\mathsf{E}[Z]^2}}$. Since $\mathsf{Var}[Z] < \mathsf{E}[Z]$, the denominator on the right hand side of the above inequality is $< \left(1 + \frac{1}{(1-\theta)^2.\mathsf{E}[Z]}\right)$. Furthermore since $\mathsf{E}[Z] \geq 1$, the denominator is $< \left(1 + \frac{1}{(1-\theta)^2}\right)$. Hence $\mathsf{Pr}[\, Z > \theta.\mathsf{E}[Z]\,] > \frac{1}{1+\frac{1}{(1-\theta)^2}}$. Letting $\theta = 0$, we get $\mathsf{Pr}[\, Z > 0\,] > 1/2$. Therefore, if $|\mathsf{Sol}(\varphi)| \geq c$, we have $\mathsf{Pr}[\, Y = 1\,] > 1/2$, or equivalently $\mathsf{Pr}[\, Y = 0\,] < 1/2$.
- If $|\mathsf{Sol}(\varphi)| \leq c/4$, then $\mathsf{E}[Z] = |\mathsf{Sol}(\varphi)|/2^m < 1/2$. By Markov inequality, we also know that $\mathsf{Pr}[\, Z \geq 1\,] \leq \frac{\mathsf{E}[Z]}{1} < 1/2$. Therefore, if $|\mathsf{Sol}(\varphi)| \leq c/4$, we have $\mathsf{Pr}[\, Y = 1\,] < 1/2$.

Note that in both cases above, the error probability is $< 1/2$. Therefore, by repeating the above algorithm $\mathcal{O}(\log(1/\delta))$ times, and by choosing Y to be the majority output obtained from these runs, the error probability can be reduced to $\leq \delta$. This gives us a $\mathsf{BPP}^{\mathsf{NP}}$ algorithm.The number of times an NP-oracle is invoked is in $\mathcal{O}(\log(1/\delta))$.

□

Interestingly, a direct reduction of α-PMT to PACMC is also possible. However, this does not yield an algorithm that solves α-PMT with $\mathcal{O}(\log(1/\delta))$ calls to an NP-oracle. The best known techniques for PAAMC require $\mathcal{O}(\log n \log(\frac{1}{\delta}))$ calls to an NP oracle (Meel 2017). As noted earlier, literal-weighted WMC can be polynomially reduced to MC (Chakraborty et al. 2015). It follows from Theorem 5 that the literal-weighted version of α-PMT is also in $\mathsf{BPP}^{\mathsf{NP}}$ and can be solved with $\mathcal{O}(\log(1/\delta))$ calls to an NP-oracle. We believe that (weighted and unweighted) α-PMT is likely to be useful in thredholding applications (viz. hypothesis testing), since it buys us significant computational efficiency by trading off confidence in only a restricted range of counts.

6 Concluding Remarks

Probabilistic inference forms the core of several decision making tools today; yet it is provably computationally hard. In this work, we formalize several relaxations based on model-counting that have been used earlier, and show that not all of them yield worst-case complexity benefits. In view of the significant progress made in SAT solving, and inspired by the success of randomized algorithms that make polynomially many SAT calls, we expand our space of efficient algorithms to include the class $\mathsf{BPP}^{\mathsf{NP}}$. This permits separating relaxations (viz. PACMC) that benefit from advances in SAT-solvers from those (viz. AAMC, PAAMC, MAMC) that are inherently much harder. We also propose a new relaxation called α-PMT that is in $\mathsf{BPP}^{\mathsf{NP}}$ and useful in settings like hypothesis testing. We hope that our results will enable the probablistic inference community to make relaxation choices grounded in rigorous complexity theoretic arguments.

While our work establishes that the worst-case complexity of algorithms implementing several counting relaxations (with strong guarantees) must be high, the observed behaviour of algorithms in prior work depends on several factors: mix of benchmarks, presence of structure/symmetry in benchmarks that render the problem easier to solve, match of heuristics with mix of benchmarks, performance of libraries used in the tool and the like. We believe our work highlights the need for investigating the root causes of discrepancy between worst-case and observed runtime performance of algorithms in prior work more closely. Note that such discrepancy is not uncommon in other domains, e.g. simplex vs interior points in linear programming, conflict-driven clause learning in SAT solving (Malik and Zhang 2009) etc.

Acknowledgments We are thankful to Jeffrey Dudek for valuable feedback on several of the proofs. This work was supported in part by NUS ODPRT Grant R-252-000-685-133 and AI Singapore Grant R-252-000-A16-490.

References

Antonucci, A.; Huber, D.; Zaffalon, M.; Luginbuhl, P.; Chapman, I.; and Ladouceur, R. 2013. CREDO: A military decision-support system based on credal networks. In *FUSION*, 1942–1949. IEEE.

Arora, S., and Barak, B. 2009. *Computational Complexity: A Modern Approach*. Cambridge Univ. Press.

Bornholt, J.; Mytkowicz, T.; and McKinley, K. S. 2014. Uncertain⟨T⟩: A first-order type for uncertain data. In *Proc. of ASPLOS*, 51–66.

Carter, J. L., and Wegman, M. N. 1977. Universal classes of hash functions. In *Proc. of STOC*, 106–112. ACM.

Chakraborty, S.; Fremont, D. J.; Meel, K. S.; Seshia, S. A.; and Vardi, M. Y. 2014. Distribution-aware sampling and weighted model counting for SAT. In *Proc. of AAAI*, 1722–1730.

Chakraborty, S.; Fried, D.; Meel, K. S.; and Vardi, M. Y. 2015. From weighted to unweighted model counting. In *Proc. of IJCAI*, 689–695.

Chakraborty, S.; Meel, K. S.; Mistry, R.; and Vardi, M. Y. 2016. Approximate probabilistic inference via word-level counting. In *Proc. of AAAI*.

Chakraborty, S.; Meel, K. S.; and Vardi, M. Y. 2013a. A scalable and nearly uniform generator of SAT witnesses. In *Proc. of CAV*, 608–623.

Chakraborty, S.; Meel, K. S.; and Vardi, M. Y. 2013b. A scalable approximate model counter. In *Proc. of CP*, 200–216.

Chakraborty, S.; Meel, K. S.; and Vardi, M. Y. 2016. Algorithmic improvements in approximate counting for probabilistic inference: From linear to logarithmic SAT calls. In *Proc. of IJCAI*.

Chavira, M., and Darwiche, A. 2008. On probabilistic inference by weighted model counting. *Artificial Intelligence* 172(6):772–799.

Cruz-Ramirez, N.; Acosta-Mesa, H. G.; Carrillo-Calvet, H.; Nava-Fernández, L. A.; and Barrientos-Martinez, R. E. 2007. Diagnosis of breast cancer using bayesian networks: A case study. *Computers in Biology and Medicine* 37(11):1553–1564.

Dagum, P., and Luby, M. 1993. Approximating probabilistic inference in bayesian belief networks is NP-hard. *Artificial Intelligence* 60(1):141–153.

de Campos, C. P.; Stamoulis, G.; and Weyland, D. 2017. A structured view on weighted counting with relations to counting, quantum computation and applications. *arXiv preprint arXiv:1701.06386*.

Ermon, S.; Gomes, C. P.; Sabharwal, A.; and Selman, B. 2013. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *Proc. of ICML*, 334–342.

Fink, R.; Huang, J.; and Olteanu, D. 2013. Anytime approximation in probabilistic databases. *The VLDB Journal* 22(6):823–848.

Gordon, A. D.; Henzinger, T. A.; Nori, A. V.; and Rajamani, S. K. 2014. Probabilistic Programming. In *Proc. of ICSE*.

Ihler, A. T.; III, J. W. F.; and Willsky, A. S. 2005. Loopy belief propagation: Convergence and effects of message errors. *Journal of Machine Learning Research* 6:905–936.

Jerrum, M.; Valiant, L.; and Vazirani, V. 1986. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science* 43(2-3):169–188.

King, B. M.; Rosopa, P. J.; and Minium, E. W. 2010. *Statistical Reasoning in the Behavioural Sciences*. Wiley.

Koller, D., and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT press.

Kwisthout, J. 2018. Approximate inference in Bayesian networks: Parameterized complexity results. *International Journal of Approximate Reasoning* 93:119–131.

Lautemann, C. 1983. BPP and the polynomial hierarchy. *Information Processing Letters* 17(4):215–217.

Malik, S., and Zhang, L. 2009. Boolean satisfiability from theoretical hardness to practical success. *Commun. ACM* 52(8):76–82.

Meel, K. S.; Vardi, M.; Chakraborty, S.; Fremont, D. J.; Seshia, S. A.; Fried, D.; Ivrii, A.; and Malik, S. 2016. Constrained sampling and counting: Universal hashing meets sat solving. In *Proc. of Beyond NP Workshop*.

Meel, K. S. 2014. *Sampling Techniques for Boolean Satisfiability*. M.S. Thesis, Rice University.

Meel, K. S. 2017. *Constrained Counting and Sampling: Bridging the Gap between Theory and Practice*. Ph.D. Dissertation, Rice University.

Moyé, L. A. 2006. *Statistical Reasoning in Medicine: The Intuitive P-value primer*. Springer.

Murphy, K. P.; Weiss, Y.; and Jordan, M. I. 1999. Loopy belief propagation for approximate inference: An empirical study. In *Proc. of UAI*, 467–475.

Neal, R. 1993. *Probabilistic inference using Markov Chain Monte Carlo methods*. Department of Computer Science, University of Toronto, Ontario, Canada.

Onisko, A.; Druzdzel, M. J.; and Wasyluk, H. 1999. A bayesian network model for diagnosis of liver disorders. In *Proc. of 11th Conference on Biocybernetics and Biomedical Engineering*, 842–846.

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

Sarkhel, S.; Venugopal, D.; Pham, T. A.; Singla, P.; and Gogate, V. 2016. Scalable training of Markov logic networks using approximate counting. In *Proc. of AAAI*, 1067–1073.

Seixas, F. L.; B. Zadrozny, J. Laks, A. C.; and Saade, D. C. M. 2014. A bayesian network decision model for supporting the diagnosis of dementia, alzheimer's disease and mild cognitive impairment. *Computers in Biology and Medicine* 51:140–158.

Sipser, M. 1983. A complexity theoretic approach to randomness. In *Proc, of STOC*, 330–335. ACM.

Stockmeyer, L. 1983. The complexity of approximate counting. In *Proc. of STOC*, 118–126.

Toda, S. 1989. On the computational power of PP and (+)P. In *Proc. of FOCS*, 514–519. IEEE.

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

Valiant, L. 2013. *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. Basic Books.

Wainwright, M. J., and Jordan, M. I. 2008. Graphical models, exponential families, and variational inference. *Found. Trends Machine Learning* 1(1-2):1–305.

Walpole, R. E.; Myers, R. H.; Myers, S. L.; and Ye, K. 1993. *Probability and statistics for engineers and scientists*, volume 8. Prentice Hall Upper Saddle Riverˆ eNJ NJ.

Wexler, Y., and Meek, C. 2008. MAS: a multiplicative approximation scheme for probabilistic inference. In *Proc. of NIPS*, 1761–1768.

Zhu, M., and Ermon, S. 2015. A hybrid approach for probabilistic inference using random projections. In *Proc. of ICML*, 2039–2047.

Zongming, L. 2009. The importance of hypothesis testing in quality management. *Quality Magazine*.