Hashing-Based Approximate Probabilistic Inference in Hybrid Domains[*]

Vaishak Belle
Dept. of Computer Science
KU Leuven
Belgium
vaishak@cs.kuleuven.be

Guy Van den Broeck
Dept. of Computer Science
KU Leuven
Belgium
guy.vandenbroeck@cs.kuleuven.be

Andrea Passerini
DISI
University of Trento
Italy
passerini@disi.unitn.it

Abstract

In recent years, there has been considerable progress on fast randomized algorithms that approximate probabilistic inference with tight tolerance and confidence guarantees. The idea here is to formulate inference as a counting task over an annotated propositional theory, called weighted model counting (WMC), which can be partitioned into smaller tasks using universal hashing. An inherent limitation of this approach, however, is that it only admits the inference of discrete probability distributions. In this work, we consider the problem of approximating inference tasks for a probability distribution defined over discrete and continuous random variables. Building on a notion called weighted model integration, which is a strict generalization of WMC and is based on annotating Boolean and arithmetic constraints, we show how probabilistic inference in hybrid domains can be put within reach of hashing-based WMC solvers. Empirical evaluations demonstrate the applicability and promise of the proposal.

1 INTRODUCTION

Weighted model counting (WMC) on a propositional knowledge base is an effective and general approach to probabilistic inference in a variety of formalisms, including Bayesian and Markov Networks. It extends the model counting task, or #SAT, which is to count the number of assignments (that is, models) that satisfy a given logical sentence (Gomes *et al.*, 2009). In WMC, one accords a weight to every model, and computes the sum of the weights of all models. The WMC formulation has recently emerged as an assembly language for probabilistic reasoning, offering a basic formalism for encoding various inference problems. State-of-the-art reasoning algorithms for Bayesian networks (Chavira and Darwiche, 2008), their relational extensions (Chavira *et al.*, 2006), factor graphs (Choi *et al.*, 2013), probabilistic programs (Fierens *et al.*, 2013), and probabilistic databases (Suciu *et al.*, 2011) reduce their inference problem to a WMC computation. The task has been generalized to first-order knowledge bases as well (Van den Broeck *et al.*, 2011; Gogate and Domingos, 2011). Exact WMC solvers are based on knowledge compilation (Darwiche, 2004; Muise *et al.*, 2012) or DPLL search with component caching (Sang *et al.*, 2005).

However, exact inference is #P-hard (Valiant, 1979), and so, there is a growing interest in approximate model counters. Beginning with Stockmeyer (1983), who showed that approximating model counting with a tolerance factor can be achieved in deterministic polynomial time using a Σ_2^P-oracle, a number of more recent results show how random polynomial-time realizations are possible using an NP-oracle (e.g., a SAT solver) (Jerrum *et al.*, 1986; Karp *et al.*, 1989; Bellare *et al.*, 2000; Gomes *et al.*, 2006; Ermon *et al.*, 2013b, 2014; Chakraborty *et al.*, 2013a,b). The central idea here is the use of random parity constraints, in the form of *universal hash functions* (Sipser, 1983), that partition the model counting solution space in an inexpensive manner. Most of the recent work in the area, moreover, come with strong tolerance-confidence guarantees (introduced later), and scale well by leveraging SAT technology.

The popularity of WMC can be explained as follows. Its formulation elegantly decouples the logical or symbolic representation from the statistical or numeric one, which is encapsulated in the weight function. When building solvers, this allows us to reason about logical equivalence and reuse SAT solving technology (such as constraint propagation and clause learning). WMC also makes it more natural to reason about deterministic, hard constraints in a probabilistic context. Nevertheless, WMC has a fundamental *limitation*: it is purely Boolean. This means that the advantages mentioned above only apply to *discrete proba-*

[*]This is a revised paper that corrects errors in the original implementation and describes the experimental setup more clearly. The changes only affect the experiments in Section 4.

bility distributions.

To counter this, in a companion paper (Belle *et al.*, 2015), we proposed the notion of *weighted model integration* (WMI). It is based on *satisfiability modulo theories* (SMT), which enable us to, for example, reason about the satisfiability of linear constraints over the rationals. The WMI task is defined on the models of an SMT theory Δ, containing mixtures of Boolean and continuous variables. For every assignment to the Boolean and continuous variables, the WMI problem defines a weight. The total WMI is computed by integrating these weights over the domain of solutions of Δ, which is a mixed discrete-continuous space. Consider, for example, the special case when Δ has no Boolean variables, and the weight of every model is 1. Then, the WMI simplifies to computing the volume of the polytope encoded in Δ. Overall, weighted SMT theories admit a natural encoding of hybrid Markov and Bayesian networks, analogous to the encodings of discrete graphical networks using weighted propositional theories.

In this work, we consider the problem of approximating inference tasks for a probability distribution defined over discrete and continuous random variables. Formulated as a WMI task, we address the question as to whether fast hashing-based approximate WMC solvers can be leveraged for hybrid domains. What we show is that an NP-oracle can indeed effectively partition the model counting solution space of the more intricate mixed discrete-continuous case using universal hashing. (Of course, volume computation is still necessary, but often over very small spaces.) In this sense, hybrid domains can now be put within reach of approximate WMC solvers. In particular, the hashing approach that we consider here builds on the recent work of Chakraborty *et al.* (2014) on approximate WMC, and inherits their tolerance-confidence guarantees. In our empirical evaluations, the approximate technique is shown to be significantly faster than an exact WMI solver. We then demonstrate the practical efficacy of the system on a complex real-world dataset where we compute conditional queries over intricate arithmetic constraints that would be difficult (or impossible) to realize in existing formalisms.

Let us finally mention that current inference algorithms for hybrid graphical models often make strong assumptions on the form of the potentials, such as Gaussian distributions (Lauritzen and Jensen, 2001), or approximate using variational methods (Murphy, 1999; Lunn *et al.*, 2000), for which quality guarantees are difficult to obtain. There is also a recent focus on *piecewise-polynomial* potentials (Shenoy and West, 2011; Sanner and Abbasnejad, 2012; Wang *et al.*, 2014), which are based on generalizations of techniques such as the join-tree algorithm. Such piecewise-polynomials can also be represented in the WMI context, but in a general framework allowing arbitrary Boolean connectives and deterministic hard constraints.

2 PRELIMINARIES

We begin with probabilistic models, and then turn to the necessary logical background, WMC and WMI.

2.1 PROBABILISTIC MODELS

Let $\mathcal{B}$ and $\mathcal{X}$ denote sets of Boolean and real-valued random variables, that is, $b \in \mathcal{B}$ is assumed to take values from $\{0,1\}$ and $x \in \mathcal{X}$ takes values from $\mathbb{R}$. We let $(\mathbf{b},\mathbf{x}) = (b_1,\ldots,b_m,x_1,\ldots,x_n)$ be an element of the probability space $\{0,1\}^m \times \mathbb{R}^n$, which denotes a particular assignment to the random variables from their respective domains. We let the joint probability density function be denoted by Pr. So $\Pr(\mathbf{b},\mathbf{x})$ determines the probability of the assignment vector. When these random variables are defined by a set of dependencies, as can be represented using an *undirected graphical model* (that is, Markov network), the density function is compactly *factorized*. See Koller and Friedman (2009) for details.

2.2 LOGICAL BACKGROUND

Propositional satisfiability (SAT) is the problem of deciding whether a logical formula over Boolean variables and logical connectives can be satisfied by some truth value assignment of the Boolean variables. Given a formula ϕ and assignment (or model or world) M, we write $M \models \phi$ to denote *satisfaction*. We write $l \in M$ to denote the literals (that is, propositions or their negations) that are satisfied at M. We often write $\mathcal{M}(\phi)$ to mean the set of models of ϕ.

A generalization to this decision problem is that of *Satisfiability Modulo Theories* (SMT). In SMT, we are interested in deciding the satisfiability of a (typically quantifier-free) first-order formula with respect to some decidable background theory $\mathcal{T}$, such as linear arithmetic over the rationals ($\mathcal{LRA}$). Standard first-order models can be used to formulate SMT; see Barrett *et al.* (2009) for details. Moreover various background theories, like $\mathcal{LRA}$ and linear arithmetic over the integers ($\mathcal{LIA}$), can be combined. In this paper we are interested in a combination of $\mathcal{LRA}$ and propositional logic, for which satisfaction is defined in an obvious way.

Our formulation will also use the concepts of *formula abstraction* and *refinement* (Barrett *et al.*, 2009). Here, first, a bijection is established between ground first-order atoms and a propositional vocabulary; abstraction proceeds by replacing the atoms by propositions, and refinement replaces the propositions with the atoms. In the sequel, we refer to the propositional abstraction of an SMT formula ϕ as ϕ^- and the refinement of ϕ as ϕ^+. For example, if $\Delta = (x \leq 4) \wedge (x \leq 5)$, then $\Delta^- = p \wedge q$ where (say) p denotes $x \leq 4$ and q denotes $x \leq 5$; also, $q^+ = x \leq 5$.

2.3 WEIGHTED MODEL COUNTING

Weighted model counting (Chavira and Darwiche, 2008) is an extension of model counting (Gomes *et al.*, 2009). In model counting, also known as #SAT, one counts the number of satisfying assignments of a propositional sentence. In WMC, each assignment has an associated weight and the task is to compute the sum of the weights of all satisfying assignments. WMC has applications in probabilistic inference in discrete graphical models.

Definition 1: Given a formula Δ in propositional logic over literals $\mathcal{L}$, and a *weight function* $w : \mathcal{L} \rightarrow \mathbb{R}$, the *weighted model count* (WMC) is defined as:

$$\text{WMC}(\Delta, w) = \sum_{M \models \Delta} w(M)$$

where, $w(M)$ is shorthand for $\prod_{l \in M} w(l)$.

Intuitively, the weight of a formula is given in terms of the total weight of its models; the weight of a model is defined in terms of the literals true in that model.

We are often interested in computing the probability of a query q given evidence e in a Boolean Markov network N, for which we use:

$$\Pr_N(q \mid e) = \frac{\text{WMC}(q \wedge e \wedge \Delta, w)}{\text{WMC}(e \wedge \Delta, w)}$$

where Δ encodes N and w encodes the potentials; see, for example, Chavira and Darwiche (2008).

2.4 WEIGHTED MODEL INTEGRATION

As noted before, an inherent limitation of WMC is that it only admits the inference of discrete probability distributions. To remedy this, in a companion paper (Belle *et al.*, 2015), we introduced the notion of *weighted model integration* as a strict generalization of WMC. The main idea here is to take a logical theory with rational and Boolean variables, that is, from a combination of $\mathcal{LRA}$ and propositional logic, and annotate it with weights. As before, propositional assignments are denoted using M.

Definition 2: Suppose Δ is an SMT theory over Boolean and rational variables $\mathcal{B}$ and $\mathcal{X}$, and literals $\mathcal{L}$. Suppose $w : \mathcal{L} \rightarrow \text{EXPR}(\mathcal{X})$, where $\text{EXPR}(\mathcal{X})$ are expressions over $\mathcal{X}$. Then the *weighted model integral* (WMI) is defined as:

$$\text{WMI}(\Delta, w) = \sum_{M \models \Delta^-} \text{VOL}(M, w)$$

$$\text{where, } \text{VOL}(M, w) = \int_{\{l^+ : l \in M\}} w(M)\, d\mathcal{X}.$$

The main feature of the definition is how it casts the weighted model counting problem over SMT in standard propositional logic. The intuition is as follows. The WMI of an SMT theory Δ is defined in terms of the models of its propositional abstraction Δ^-. For each such model, we compute its volume, that is, we integrate the weight values of the literals that are true at the model. The interval of the integral is obtained from the refinement of each literal.[1] The mathematical expression for conditional probabilities is as before.

The general idea with $\text{EXPR}(\mathcal{X})$ is that the weight function maps an expression e to its *density function*, which is usually another expression mentioning the variables in e. We note that the input language for a WMI task is easily seen to capture constraints involving discrete and continuous random variables over arbitrary Boolean connectives.

To see WMI in action, consider a simple example:

Example 3: Suppose Δ is the following formula:

$$p \vee (0 \leq x \leq 10)$$

For weights, let $w(p) = .1$, $w(\neg p) = 2x$, $w(q) = 1$ and $w(\neg q) = 0$, where q is the propositional abstraction of $(0 \leq x \leq 10)$. Roughly, this can be seen to say that x is uniformly distributed when p holds and otherwise it is characterized by a triangular distribution in the interval $[0, 10]$. There are three models of Δ^-, for which we calculate $\text{VOL}(\cdot, w)$:

1. $\text{VOL}(\{p, \neg q\}, w) = 0$ because $w(\neg q) = 0$;
2. $\text{VOL}(\{\neg p, q\}, w) = \int_{0 \leq x \leq 10} 2x\, dx = \left[x^2\right]_0^{10} = 100.$
3. $\text{VOL}(\{p, q\}, w) = \int_{0 \leq x \leq 10} .1\, dx = [.1 \cdot x]_0^{10} = 1.$

Thus, $\text{WMI}(\Delta, w) = 100 + 1 = 101$.

Suppose that we are interested in the probability of the query $x \leq 3$ given that $\neg p$ is observed. Suppose r is the abstraction of $x \leq 3$. First, $\text{WMI}(\Delta \wedge \neg p, w)$ corresponds to the weight of a single interpretation, that of item 2, yielding a value of 100. Next, $\text{WMI}(\Delta \wedge \neg p \wedge x \leq 3, w)$ also corresponds to the weight of a single interpretation $\{\neg p, q, r\}$, an extension to that in item 2. In this case:

$$\text{VOL}(\{\neg p, q, r\}, w) = \int_{(0 \leq x \leq 10) \wedge (x \leq 3)} 2x\, dx = \left[x^2\right]_0^3 = 9.$$

[1] Although the interval is defined in terms of SMT literals, this is meant to denote standard integrals in an obvious fashion:

$$\int_{x \leq 6} \phi dx \doteq \int_{-\infty}^{6} \phi dx; \int_{x \geq 6} \phi dx \doteq \int_{6}^{\infty} \phi dx; \int_{5 \leq x \leq 6} \phi dx \doteq \int_{5}^{6} \phi dx$$

Likewise, over connectives:

$$\int_{x \leq 6 \wedge y \geq 5} \phi dxdy \doteq \int_{x \leq 6} \int_{y \geq 5} \phi dxdy.$$

When propositions appear as intervals, they are simply ignored. See Belle *et al.* (2015) for the general definition.

Therefore, the conditional probability is $9/100 = .09$. □

The correctness of WMI and that it is a strict generalization of WMC are argued elsewhere (Belle *et al.*, 2015).[2]

Let us conclude this section by remarking that although the definition of WMI is very general, for most practical purposes, we restrict densities to piecewise polynomials, where w maps $\mathcal{L}$ to polynomials over $\mathcal{X}$. Such piecewise polynomials can approximate a wide variety of continuous distributions, including Gaussians (Shenoy and West, 2011; Sanner and Abbasnejad, 2012). Moreover, Baldoni *et al.* (2011) show that for a fixed number of variables, the integration is efficient, even for polynomials of increasing degree. Thus, smooth function approximations are possible in practice, and come at a reasonable cost.

3 APPROXIMATING WMI

In this section, we identify how to approximate $\text{WMI}(\Delta, w)$ for an arbitrary Δ and non-degenerate (see below) w with strong theoretical guarantees by appealing to a SAT-oracle.

3.1 PROBLEM STATEMENT FOR WMC

To better understand the problem statement, let us begin with the case of WMC:

Definition 4: Given a propositional formula Δ and a weight function w, an *exact algorithm* for WMC returns $\text{WMC}(\Delta, w)$. An *approximate algorithm* for WMC given *tolerance* $\epsilon \in (0, 1]$ and *confidence* $1 - \delta \in (0, 1]$, simply called an (ϵ, δ)-algorithm, returns a value v such that

$$\Pr\left[\frac{\text{WMC}(\Delta, w)}{1+\epsilon} \leq v \leq (1+\epsilon)\text{WMC}(\Delta, w)\right] \geq 1 - \delta$$

Intuitively, when the weight of every model is 1, an exact algorithm returns the size of the set $\mathcal{M}(\Delta) = \{M \mid M \models \Delta\}$ while an approximate one samples from that solution space. Exact algorithms are #P-hard (Valiant, 1979) but for the approximate case random polynomial time realizations are known (Jerrum *et al.*, 1986; Karp *et al.*, 1989).[3]

[2]In an independent and recent effort, Chistikov *et al.* (2015) also introduce the notion of approximate model counting for SMT theories. The most significant difference between the proposals is that they focus only on unweighted model counting. Moreover, they define model counting as a measure on first-order models. Our approach is a simpler one based on propositional abstractions, which (as we will see) allows us to cast statements for WMI as WMC in a direct way.

[3]The class of (ϵ, δ)-algorithms that we are after follows the terminology of Karp *et al.* (1989). These can be contrasted to *bounding* counters only parameterized by confidence probabilities, such as (Kroc *et al.*, 2011).

3.2 PROBLEM STATEMENT FOR WMI

To see how the above notions apply to our task, consider an SMT theory Δ and weight function w. We observe that

$$\text{WMI}(\Delta, w) = \text{WMC}(\Delta^-, u)$$

where, for any model M of Δ^-, u is a weight function such that $u(M) = \text{VOL}(M, w)$. More precisely, u is to be seen as a weight function that does not factorize over literals and directly maps interpretations to $\mathbb{R}$. (This is without any loss of generality.) Thus, our problem statement becomes:

Definition 5: An (ϵ, δ)-algorithm for a WMI problem over Δ and w is an (ϵ, δ)-algorithm for WMC over Δ^- and weight function u, where for any model M of Δ^-, $u(M) = \text{VOL}(M, w)$.

The idea is that by treating the volumes of models as weights over propositional interpretations, we can view WMI simply in terms of WMC. Theoretical results can then be imported for our purposes.

Given an (ϵ, δ)-algorithm for WMC, there are two caveats, however. First, weights of interpretations need to be actually computed using integration during inference, but (usually) over a small number of literals and their polynomial potentials true in a model. Second, such an algorithm samples feasible satisfying assignments for Δ^-, but these need not be $\mathcal{T}$-consistent. For example, if p denotes $x \leq 3$ and q denotes $x \leq 5$, then the interpretation $\{p, \neg q\}$ is not a model in $\mathcal{LRA}$ on refinement. In the formal machinery, the weight of this model is easily seen to be 0 (that is, the interval of the integral will be an empty set), and so these models can freely appear in the problem statement. In practice, these theory inconsistency models are rejected in the WMC calculation, and once found, the algorithm can be made to refine its search of feasible solutions by incorporating these models as blocked clauses.

3.3 APPROACH

In sum, what we are after is an (ϵ, δ)-algorithm for $\text{WMC}(\Delta, w)$ for a propositional theory Δ and weight function w. Consider classical model counting, that is, where the weight of every model is 1, which is #P-hard. Bellare *et al.* (2000) were the first to show that satisfying assignments can be generated uniformly in random polynomial-time using only an NP-oracle (e.g., a SAT solver), improving and complementing earlier results (Jerrum *et al.*, 1986; Karp *et al.*, 1989; Stockmeyer, 1983). Adapting these techniques further, Chakraborty *et al.* (2013a,b) introduce a scalable approximate model counter. See Gomes *et al.* (2006) and Ermon *et al.* (2013b) for closely related proposals.

3.3.1 Counting by Hashing

The central idea in Bellare *et al.* (2000) and Chakraborty *et al.* (2013b) is the use of *universal hash functions* (Sipser, 1983):

Definition 6 : A family of functions $\mathcal{H} = \{h \mid \{0,1\}^n \to \{0,1\}^m\}$ is called *uniform*, written $h \overset{R}{\leftarrow} \mathcal{H}$, if it holds that for any $y \in \{0,1\}^n$, the random variable $h(y)$ is uniformly distributed in $\{0,1\}^m$.

Definition 7 : A family of functions $\mathcal{H} = \{h \mid \{0,1\}^n \to \{0,1\}^m\}$ is called *t-wise independent* if it holds that for any $x_1, \ldots, x_t \in \{0,1\}^m$, any $y_1, \ldots, y_t \in \{0,1\}^n$, and any $h \overset{R}{\leftarrow} \mathcal{H}$, we have:

$$\Pr[h(y_1) = x_1 \wedge \ldots \wedge h(y_t) = x_t] = 2^{-m \cdot t}$$

For the sake of clarity, we denote this family as $\mathcal{H}(n, m, t)$. Now, suppose $x \in \{0,1\}^m$ and $h \overset{R}{\leftarrow} \mathcal{H}(n, m, t)$. Let $h^{-1}(x) = \{y \in \{0,1\}^n \mid h(y) = x\}$. Then, the idea is that for any propositional language over n variables, $\mathcal{M}(\Delta) \subseteq \{0,1\}^n$ and $x \in \{0,1\}^m$, the set $\mathcal{M}(\Delta) \cap h^{-1}(x)$ partitions $\mathcal{M}(\Delta)$ into a set of well-balanced *cells*, each one induced by a particular choice of h. Thus, by iterating over different samples of $h \overset{R}{\leftarrow} \mathcal{H}(n, m, t)$, we can perform a small number of computations on the cells and leverage that as an estimate for the solution space as a whole.

The work of Chakraborty *et al.* (2013b) uses an efficient family of hash functions, denoted $\mathcal{H}_{\text{xor}}(n, m, 3)$ below. Given $h \overset{R}{\leftarrow} \mathcal{H}(n, m, 3)$ and $y \in \{0,1\}^n$, let $h(y)[k]$ denote the kth component of the vector obtained by applying h to y, and $y[k]$ denote the kth component of the string y. The family of hash functions of interest is defined as

$$\begin{aligned}\mathcal{H}_{\text{xor}}(n, m, 3) = \{h \mid h(y)[i] = a_{i,0} \oplus (\textstyle\bigoplus_{l=1}^{n} a_{i,l} \cdot y[l]),\\ a_{i,j} \in \{0,1\}, 1 \leq i \leq m, 0 \leq j \leq n\}\end{aligned}$$

where $\oplus$ denotes the XOR operation. By choosing values of $a_{i,j}$ randomly and independently, we can effectively choose a random hash function from the family. Gomes *et al.* (2006) show that this family of hash functions is 3-wise independent.

3.3.2 WMC by Hashing

As argued by Ermon *et al.* (2013a), the one major limitation when applying approximate model counters for probabilistic inference is that weights play an important role in deeming which samples are interesting. Therefore, uniformly sampling from $\mathcal{M}(\phi)$ is not appealing, and would lead to poor estimates of conditional probabilities. The approach taken in Ermon *et al.* (2013a) is to reformulate the inference task by an embedding for which uniform sampling suffices. While this is an attractive option, it requires a factored representation of the probability distribution and appeals to solutions of optimization problems from a MPE query (that is, finding the most likely state). In our setting, these requirements are problematic. For one thing, note that even if the original input problem uses a factored representation, we only possess a WMC problem with a weight function for interpretations that (possibly) lacks any structure. For another, MPE queries will involve computing integrals (recall that weights are computed during inference) for a large number of states, a task we would like to avoid unless necessary.

Nonetheless, extending earlier results (Bellare *et al.*, 2000; Chakraborty *et al.*, 2013b), Chakraborty *et al.* (2014) (CFMSV henceforth) show how approximate model counters can be applied to weighted model counting problems by means of a parameter called *tilt.*

Definition 8: Suppose Δ is a propositional theory and w is a weight function mapping $\mathcal{M}(\Delta)$ to strictly positive numbers. Let $w_{\max} = \max_M w(M)$ and let $w_{\min} = \min_M w(M)$. We define the *tilt* θ to be the ratio $w_{\max}/w_{\min}$.

For our purposes, we adapt the notion as follows:

Definition 9: Suppose Δ is a SMT theory over literals $\mathcal{L}$ and continuous variables $\mathcal{X}$ and w is a weight function mapping $\mathcal{L}$ to $\text{EXPR}(\mathcal{X})$. Let $w_{\max} = \max_M \text{VOL}(M, w)$ and let $w_{\min} = \min_M \text{VOL}(M, w)$. We define the *tilt* θ to be the ratio $w_{\max}/w_{\min}$.

The idea is that in approximate model counting, the number of hash functions to sample (and thus, the number of cells to construct of $\mathcal{M}(\Delta)$) is guided by the confidence parameter δ. In the approach of Chakraborty *et al.* (2014), the tilt of a problem is used to additionally ensure that an appropriate number of cells are constructed in the weighted case. While the approach requires the modeler to provide an upper bound on the tilt, the parameter is agnostic about the form of the weight function, which is desirable in our setting. Nonetheless, when the tilt is large (i.e., when the weight of an interpretation is small relative to others), it would mean that a large number of cells are constructed, which may be inefficient, and alleviating this is an interesting avenue for the future. In practice, the problems we encountered had small tilts. (In our experimental evaluations, an upper bound of 5 was provided for the tilt.)

3.4 ALGORITHM

Putting it all together, we present the pseudecode for WMI computation. It can be seen as a simple reworking of CFMSV to solve WMI.[4] For the sake of completeness, we

[4]In that sentiment, we believe an important feature of our formulation is that other WMC approaches can be adapted for WMI along the same lines.

Algorithm 1 WeightMC($\phi, u, \varepsilon, \delta, \theta$)

```
1: $\mathrm{w}_{\max} \leftarrow 1$
2: pivot $\leftarrow 2 \times \lceil e^{3/2}\left(1+\frac{1}{\varepsilon}\right)^2 \rceil$
3: iter $\leftarrow \lceil 35 \log_2(3/\delta) \rceil$
4: for $i : 1, \ldots,$ iter do
5:     $(\mathcal{M}, c, w_{max}) \leftarrow$ WeightMCCore($\phi, u,$ pivot$, \theta, w_{\max}$)
6:     store $(c, w_{max})$ if $\mathcal{M} \neq \emptyset$
7: end for
8: return the median of $c \times w_{max}$ for stored tuples
```

Algorithm 2 WeightMCCore($\phi, u,$ pivot$, \theta, w_{\max}$)

```
1: $i \leftarrow 0$; vol $\leftarrow 0$; $n \leftarrow$ number of variables in $\phi$
2: repeat
3:     $i \leftarrow i + 1$
4:     Choose $h \xleftarrow{R} \mathcal{H}_{\mathrm{xor}}(n, i, 3)$
5:     Choose $x \xleftarrow{R} \{0, 1\}^i$
6:     $(\mathcal{M},$ vol$, w_{\max}) \leftarrow$ BoundedWeightSAT($\phi, (h(b_1, \ldots, b_n) = x), u,$ pivot$, \theta, w_{\max}$)
7: until $(0 <$ vol$/w_{\max} \leq$ pivot or $i = n)$
8: if (vol$/w_{\max} >$ pivot or $\mathcal{M} = \emptyset$) then return $(\emptyset,$ vol$, w_{\max})$
9: else return $(\mathcal{M},$ vol $\cdot 2^{i-1}/w_{\max}, w_{\max})$
10: end if
```

present the essential components of the CFMSV algorithm, called WeightMC. Interested readers are referred to that work for full details. First, given an SMT theory Δ, weight function w, tolerance ϵ, confidence δ and an upper bound on the tilt θ, we compute:[5]

$$\mathrm{WMI}(\Delta, w, \epsilon, \delta, \theta) = \text{WeightMC}(\Delta^-, u, \epsilon, \delta, \theta)$$

where u is the weight function mapping interpretations to numbers and is calculated using:

$$u(M) = \mathrm{VOL}(M, w).$$

The WeightMC procedure is given in Algorithm 1. Basically, the given parameters δ and ϵ are used to determine the number of times WeightMCCore is invoked and the number of cells to induce on $\mathcal{M}(\Delta^-)$, respectively. What the procedure returns is the median of the volume estimates, obtained from WeightMCCore over these iterations. For any given iteration, if no model is found, then the estimates from WeightMCCore are ignored.

The procedure WeightMCCore applied to a propositional formula ϕ, detailed in Algorithm 2, partitions models of ϕ into cells. This is achieved by choosing 3-wise independent hash functions, and adding random parity constraints. The resulting logical formula is conjoined with

[5]CFMSV's WeightMC procedure also includes a parameter called the *independent support* of a propositional theory ϕ over variables $\mathcal{B}$. The support of ϕ is $\mathcal{B}$, and the independent support $\mathcal{I} \subseteq \mathcal{B}$ uniquely determine the truth values of variables from $\mathcal{B} - \mathcal{I}$. By choosing hash functions only on the independent support, rather than the full set of variables in $\mathcal{B}$, significant performance improvements can be gained. But since this is inessential to understanding the conceptual ideas of the algorithm, we omit further discussion on this matter.

Algorithm 3 BoundedWeightSAT($\phi, \chi, u,$ pivot$, \theta, w_{\max}$)

```
1: $w_{\min} \leftarrow w_{max}/\theta$; vol $\leftarrow 0$; $\mathcal{M} = \emptyset$; $\gamma = \phi \wedge \chi$
2: repeat
3:     $M \leftarrow$ SolveSAT($\gamma$)
4:     if $M =$ UNSAT then
5:         break
6:     end if
7:     cons $\leftarrow$ Consistent($\mathcal{T}, \{l^+ \mid l \in M\}$)
8:     if cons = INCONSISTENT then
9:         $\phi \leftarrow$ AddBlockClause($\phi, M$)
10:    else
11:        $\mathcal{M} \leftarrow \mathcal{M} \cup M$
12:        $\gamma \leftarrow$ AddBlockClause($\gamma, M$)
13:        Cache$[M] \leftarrow u(M)$ if Cache$[M] = \{\}$
14:        vol $\leftarrow$ vol + Cache$[M]$
15:        $w_{\min} \leftarrow \min(w_{\min},$ Cache$[M])$
16:    end if
17: until vol$/(w_{\min} \cdot \theta) >$ pivot
18: return $(\mathcal{M},$ vol$, w_{\min} \cdot \theta)$
```

ϕ, for which BoundedWeightSAT is invoked. The number of iterations of BoundedWeightSAT is bound by the number of propositional variables in ϕ, or when the current tilt exceeds an ϵ-based parameter. Among other things, BoundedWeightSAT returns the models of ϕ conjoined with a random parity constraint, and unless this is empty, volume estimates in WeightMCCore are returned to WeightMC. Both WeightMCCore and WeightMC are minor adaptations of the procedures from CFMSV in the following sense: in the CFMSV modules, the weights are directly used; for example, line 7 in Algorithm 2 would explicitly refer to $u(\mathcal{M}) = \sum_{M \in \mathcal{M}} u(M)$. In our setting, computing $u(M)$ involves integration which we would not want to repeat for every iteration. Thus, weights of (sets of) models are themselves returned when required. It is easy to see that the analysis of these procedures is unaffected by this adaptation.

Finally, we turn to BoundedWeightSAT in Algorithm 3, where a more significant adaptation of the CFMSV scheme occurs. First, one would observe that, different from CFMSV, the parity constraint χ is provided separate from the input formula (for reasons justified below). Nonetheless, the procedure essentially performs a bounded version of model counting on $\gamma = \phi \wedge \xi$, as would CFMSV, where a (θ, ϵ)-derived parameter determines this bound. As soon as no model is found, the procedure exits with the current estimates. If a model M is found, however, unlike CFMSV, we need to additionally ensure the refinement of this assignment is consistent w.r.t. the background theory $\mathcal{T}$. If it is not $\mathcal{T}$-consistent, then we can prevent this model from being considered for all iterations by adding it as a block clause to our input propositional formula ϕ. (Recall also that such a model would have zero volume, leading to an infinite tilt if it were to be considered.) On the other hand, if it is theory consistent, we compute its volume and then add it as a block clause to γ. (This achieves the model counting of γ.) In particular, we calculate $u(M) = \mathrm{VOL}(M, w)$ where

w is the actual weight function from the WMI problem, and cache this result. So, integration is performed only once for the model M. The procedure then returns the total volume of the set of models $\mathcal{M}$ identified.

What is perhaps interesting to realize of this adaptation of CFMSV is that it also does not affect the analysis of the procedures. Note that, if the problem instance is a propositional theory, then $\Delta^- = \Delta$, and line 7 of Algorithm 3 is trivially true because $\mathcal{T}$ is propositional logic. Consequently, the test in line 8 is trivially false. More formally:

Proposition 10: *Suppose ϕ is a propositional formula, u is a weight function, $\epsilon, \delta \in (0, 1]$, and θ is an upper bound on the tilt. Then* $\text{WeightMC}(\phi, u, \epsilon, \delta, \theta)$ *from CFMSV's original formulation returns c iff the current adaptation does.*

This allows us, very easily, to leverage the strong guarantees of CFMSV (our rewording):

Theorem 11: *[CFMSV] Suppose $\phi, u, \epsilon, \delta, \theta$ are as above. Then* $\text{WeightMC}(\phi, u, \epsilon, \delta, \theta)$ *is an (ϵ, δ)-algorithm for* $\text{WMC}(\phi, u)$*. Given a SAT-oracle, it runs in time polynomial in* $\log_2(1/\delta), \theta, |\phi|$ *and* $1/\epsilon$ *relative to the oracle.*

From which we obtain:

Corollary 12: *Suppose Δ is an SMT theory, w is a weight function, and ϵ, δ, θ are as above. Suppose u is the derived weight function for Δ^-. Then,* $\text{WeightMC}(\Delta^-, u, \epsilon, \delta, \theta)$ *is an (ϵ, δ)-algorithm for* $\text{WMI}(\Delta, w)$*. Suppose we are given an oracle to the weight function u and a SAT-oracle. Then,* $\text{WeightMC}(\Delta^-, u, \epsilon, \delta, \theta)$ *runs in time polynomial in* $\log_2(1/\delta), \theta, |\Delta^-|$ *and* $1/\epsilon$ *relative to the oracles.*

Thus, we can inherit existing results for this WMC solver (and perhaps others). The oracle to u computes the volumes of $\mathcal{T}$-consistent models. Each instance of $\text{VOL}(M, w)$ involves the following (Belle *et al.*, 2015):

Proposition 13: *Suppose Δ is an SMT theory over continuous variables $\mathcal{X}$, M a model of Δ^-, and w is as above. Let k be the maximum degree of the polynomials in* $\{w(l) \mid l \in M\}$*. Then* $\text{VOL}(M, w)$ *integrates polynomials of degree* $k \cdot |M|$.

Basically, for any model M of Δ^-, $\text{VOL}(M, w)$ is formulated as the integration of the polynomial potentials of the literals true at M, which would be a product of $|M|$ polynomials, each of degree k. As mentioned earlier, Baldoni *et al.* (2011) show that when the number of variables is fixed (as determined by $|\mathcal{X}|$), integration is efficient. In practice, moreover, when encoding factored representations and piecewise polynomials, it is often the case that negated atoms are assigned constant weights and so we encounter polynomials of degree $k \cdot n$ where $n \ll |M|$. In essence, what we usually encounter are a small number of polynomial potentials corresponding to atoms that are true in the model.

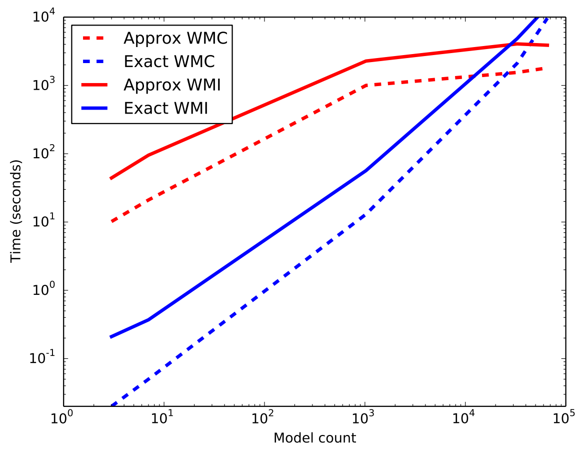


Figure 1: scaling behavior

4 EMPIRICAL EVALUATIONS

In this section, we discuss results on a prototype implementation of the approximate inference system.[6] In particular, the prototype builds on the approximate (unweighted) model counting system of Chakraborty *et al.* (2013b), extended to handle weights along the lines of Chakraborty *et al.* (2014).[7] The resulting inference system uses Z3 SMT solver v4.3.2 for testing satisfiability,[8] and the LattE software v1.6 for computing integrals.[9] All experiments were run using a system with a 2.83 GHz Intel Xeon processor and 32GB RAM.

4.1 SCALING BEHAVIOR

To better understand the cost of continuous variables and the benefits of approximate inference, we test scaling behavior using synthetic benchmarks. The intention here is to necessitate a search for all models of a theory in an exhaustive manner. For simplicity, we consider theories that are a conjunction of inequality constraints for continuous variables and clauses of the form $b_1 \vee \ldots \vee b_n$, where b_i are Boolean variables. For WMI, such a theory enables LATTE computations for every model.

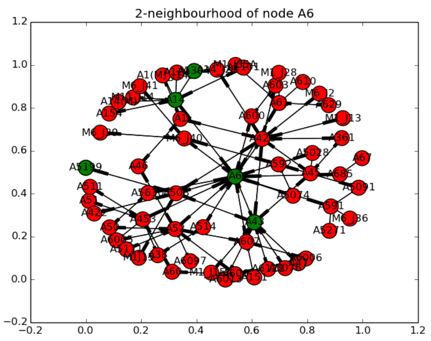
In our prior work (Belle *et al.*, 2015), an exact WMI solver

[6]The initial version of this paper reported on experiments with a prototype that contained some errors in the implementation of the pivot bounds (lines 7-8 of Algorithm 2 and line 17 of Algorithm 3). This revised section reports on a corrected prototype, available at `https://github.com/vaishakbelle/APPROXWMI`, and a new run of the experiments. We also remark that the propagation of theory-inconsistent models and the caching of volumes (line 9 and lines 13-14 of Algorithm 3 respectively) are not implemented in the prototypes.

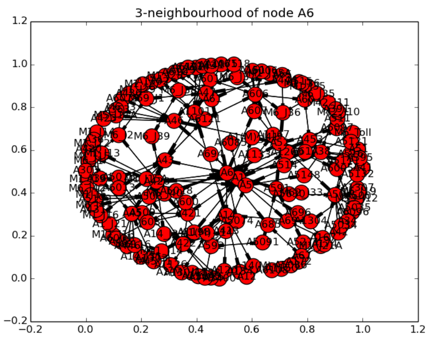
[7]Chakraborty *et al.* (2014) have distributed their own implementation for approximate weighted model counting.

[8]`http://z3.codeplex.com`

[9]`https://www.math.ucdavis.edu/~latte`



(a) at most 2 junctions



(b) at most 3 junctions

Figure 2: Strategic Road Network portions surrounding motorway A6.

was implemented using a block-clause strategy: in each iteration, if a theory-consistent model is found, a clause over the negations of the literals in the assignment is added as an additional constraint, and in this way, all models are enumerated. Using the above benchmark weighted SMT theories, we plot the approximate WMI system against the exact WMI implementation. (To assess the quality of the approximate versus exact computation of the partition function, we let the weight of every model be 1.) To further contrast this to the simpler setting of classical propositional logic, we also ran experiments for exact WMC versus approximate WMC by providing the propositional abstraction of the theory as input. (Recall that by providing a propositional theory with numeric weights, WMI reduces to WMC.)

The experiments were run with $\epsilon = .8$, $\delta = .2$ and $\theta = 1$ (because of the uniform weights). We observe in Figure 1 that while for smaller theories, Exact WMI and Exact WMC are faster, this is no longer the case for theory files with model counts greater than 2^{15}, that is, for theory files with roughly 15 (or more) Boolean variables. Finally, we remark that the answers computed by approximate WMI equals the answers computed by exact WMI ± 1, and likewise for WMC.

4.2 REAL-WORLD DATASET

To demonstrate the expressivity of WMI in a complex real-world scenario, we consider the following novel application involving conditional queries over arithmetic constraints. It uses a data series released by the UK government that provides average journey time, speed and traffic flow information on all motorways, known as the Strategic Road Network, in England.[10] Motorways are split into junctions, and each information record refers to a specific junction, day and time period. In the following we consider the 2012 dataset, with over 7 million entries, and focus on the surroundings of the A6 motorway. Figures 2a and 2b show the portion of the network with at most two and three junctions respectively from A6. We extract statistics on journey time across each junction. For the sake of simplicity, we model a junction's journey time as a uniform distribution between the observed minimum and maximum travel time.

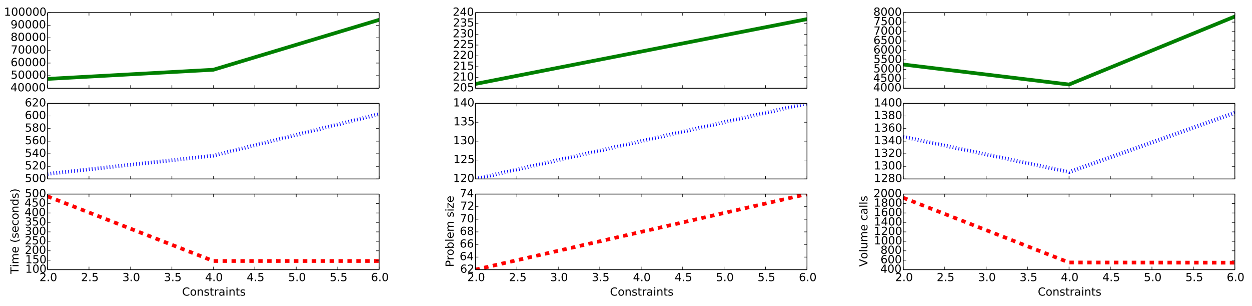
Consider a planning problem for a supply system for motorway service stations. The operations center (located, say, somewhere along A6) receives supply requests from a number of stations, and needs to predict whether the delivery vehicle will be able to reach all stations and return within a certain amount of time. Travel time between every pair of stations, and between stations and the operations center, is computed in terms of shortest paths across the network. We compute shortest paths for both minimum and maximum travel times, so as to get a distribution for the shortest path duration w.r.t. every pair of relevant points (stations and operations center), which, as noted, is uniform between these two extremes. Given a certain route between stations, the probability of completing it within the desired time can be computed by integrating over travel time distributions between consecutive stops. However, the optimal route can not be fixed a-priori (as in standard TSP problems), as the vehicle also performs deliveries between stations and to the center, depending on the current needs. These deliveries enforce various constraints on the allowed routes. The overall probability is thus obtained by summing over all valid routes, given the known constraints.

Consider, for example, the case in which stations at A14, A1304, A43, and A5199 need to be visited before returning to the operations center. Figure 2a depicts this case, showing the portion of the network with at most two junctions away from A6. (The nodes to be visited are colored green.) The probability of beginning from the operations center at 8 a.m. and completing the route by 9 a.m., considering all possible paths, is:

$$\Pr(T < 3600) = 0.765,$$

computed using the approximate WMI module, where T is the overall time measured in seconds. Now suppose a con-

[10]http://data.gov.uk/dataset/dft-eng-srn-routes-journey-times



(a) time (b) problem size (c) volume computations

Figure 3: Probabilistic reasoning about the Strategic Road Network surrounding motorway A6. Figures 3a-3c plot cycle lengths of 5 (dotted red), 6 (finely dotted blue) and 7 (solid green), relating constraints to time, problem size and volume computations respectively.

straint says that station A14 should be reached only after visiting A1304 (owing to a delivery request between these two stations). The probability then needs to be updated according to this evidence, which rules out part of the possible routes. Nonetheless, the probability of completing the task is almost unchanged:

$$\Pr(T < 3600 \mid t_{\mathrm{A14}} > t_{\mathrm{A1304}}) = 0.770,$$

where the minor increase is due to some slightly suboptimal routes being disallowed. Suppose further an additional constraint specifies that station A1304 should not be visited before 8:30 a.m. (say, because the package to deliver will not be available until then). This additional evidence brings success probability down to:

$$\Pr(T < 3600 \mid t_{\mathrm{A14}} > t_{\mathrm{A1304}} \wedge t_{\mathrm{A1304}} \geq 1800) = 0.557.$$

Finally, suppose a last constraint were to require the station A5199 to be also visited after 8:30 a.m. (say, when a package to be delivered to the operations center will be made available). This additional constraint makes it infeasible to complete the route in the required time:

$$\Pr(T < 3600 \mid t_{\mathrm{A14}} > t_{\mathrm{A1304}} \wedge \\ t_{\mathrm{A1304}} \geq 1800 \wedge t_{\mathrm{A5199}} \geq 1800) = 0.$$

Note that for such carefully constructed small-size problems in the 2-neighborhood case, exact and approximate procedures return the very same results in about the same time. (The exact procedure, however, quickly becomes infeasible for increasing cycle lengths.)

Using this example (which is to be seen as a cycle of length 5: A6–A14–A1304–A43–A5199–A6) as a template we randomly generated a number of test problems on the more complex 3-neighborhood setting, and plot an overview of these experiments in Figure 3.[11] These test problems are diverse in their modeling power: ranging from inequality constraints (e.g., $t_{\mathrm{A5199}} \geq 1800$) to ordering constraints (e.g., A5199 after A1304), and Boolean combinations thereof, often leading to more than 300 complex SMT formulas. (Intuitively, if an SMT formula has n SMT literals, these can possibly denote joint piecewise potentials of n continuous random variables.) The figures depict the behaviors for cycles of length 5, 6 and 7, ordered by constraints against the time taken, the problem size (which is the number of complex SMT formulas in the theory), and the number of calls of VOL$(\cdot, \cdot)$. Besides demonstrating the scalability of approximate WMI in such a setting, one also observes that while additional constraints increases the size of the theory, and thus, the number of random variables and volume computations, the time taken for conditional queries does not necessarily increase because suboptimal paths are eliminated (and so, marginalization is easier).

5 CONCLUSIONS

We introduced a novel way to leverage a fast hashing-based approximate WMC methodology for inference with discrete and continuous random variables. On the one hand, SAT technology can now be exploited in challenging inference and learning tasks in hybrid domains. On the other,

[11]The nature of this application makes it challenging to determine the range of the weight function for unconstrained problems and/or large cycle lengths. Our approach here is to begin with 2 constraints at least whilst assuming that we are given the $w_{\max}$ and a tilt of 5. It is important to note that, if these assumptions do not hold, the bounds of Corollary 12 do not apply. The results obtained here must then be viewed as approximations without formal guarantees (similar to, for example, the belief propagation algorithm). An alternative approach that can recover the formal guarantees while approximating the structure of the problem is as follows: we cap the range of the weight function from above and below in service of a small tilt. This has the effect of “flattening” the density landscape, and may prevent the excessive number of volume computations needed for large ($\geq$ 7) cycle lengths. A comprehensive investigation of such strategies is left for future work.

strong tolerance-confidence guarantees can be inherited in this more complex setting. WMI and weighted SMT theories allow a natural encoding of hybrid graphical networks while also admitting the specification of arithmetic constraints in conditional queries, all of which are difficult to realize in traditional representations. We demonstrated its practical efficacy in a complex novel application, and we believe, in general, the ideas of our approach would put hybrid domains within the reach of other WMC solvers.

Acknowledgements

The authors thank Kuldeep Meel, Supratik Chakraborty, and Moshe Vardi for their generous help. Vaishak Belle is partially funded by the Research Foundation-Flanders (FWO-Vlaanderen) project on Data Cleaning and the KU Leuven GOA on Declarative Modeling for Mining and Learning. Guy Van den Broeck is supported by the Research Foundation-Flanders.

References

Velleda Baldoni, Nicole Berline, Jesus De Loera, Matthias Köppe, and Michèle Vergne. How to integrate a polynomial over a simplex. *Mathematics of Computation*, 80(273):297–325, 2011.

Clark Barrett, Roberto Sebastiani, Sanjit A. Seshia, and Cesare Tinelli. Satisfiability modulo theories. In Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, chapter 26, pages 825–885. IOS Press, 2009.

Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of NP-witnesses using an NP-oracle. *Information and Computation*, 163(2):510 – 526, 2000.

Vaishak Belle, Andrea Passerini, and Guy Van den Broeck. Probabilistic inference in hybrid domains by weighted model integration. In *IJCAI*, 2015.

Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi. A scalable and nearly uniform generator of SAT witnesses. In *CAV*, pages 608–623, 2013.

Supratik Chakraborty, Kuldeep S. Meel, and Moshe Y. Vardi. A scalable approximate model counter. In *CP*, pages 200–216, 2013.

Supratik Chakraborty, Daniel J Fremont, Kuldeep S Meel, Sanjit A Seshia, and Moshe Y Vardi. Distribution-aware sampling and weighted model counting for SAT. *AAAI*, 2014.

Mark Chavira and Adnan Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 172(6-7):772–799, April 2008.

Mark Chavira, Adnan Darwiche, and Manfred Jaeger. Compiling relational Bayesian networks for exact inference. *International Journal of Approximate Reasoning*, 42(1-2):4–20, May 2006.

Dmitry Chistikov, Rayna Dimitrova, and Rupak Majumdar. Approximate counting in SMT and value estimation for probabilistic programs. In *TACAS*, volume 9035 of *LNCS*, pages 320–334. Springer Berlin Heidelberg, 2015.

Arthur Choi, Doga Kisa, and Adnan Darwiche. Compiling probabilistic graphical models using sentential decision diagrams. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 121–132. Springer, 2013.

Adnan Darwiche. New advances in compiling CNF to decomposable negation normal form. In *Proceedings of ECAI*, pages 328–332, 2004.

Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Embed and project: Discrete sampling with universal hashing. In *NIPS*, pages 2085–2093, 2013.

Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Taming the curse of dimensionality: Discrete integration by hashing and optimization. In *ICML*, pages 334–342, 2013.

Stefano Ermon, Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Low-density parity constraints for hashing-based discrete integration. In *ICML*, pages 271–279, 2014.

Daan Fierens, Guy Van den Broeck, Joris Renkens, Dimitar Shterionov, Bernd Gutmann, Ingo Thon, Gerda Janssens, and Luc De Raedt. Inference and learning in probabilistic logic programs using weighted Boolean formulas. *Theory and Practice of Logic Programming*, 2013.

Vibhav Gogate and Pedro Domingos. Probabilistic theorem proving. In *UAI*, pages 256–265, 2011.

Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Near-uniform sampling of combinatorial spaces using XOR constraints. In *NIPS*, pages 481–488, 2006.

Carla P. Gomes, Ashish Sabharwal, and Bart Selman. Model counting. In Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, chapter 20. IOS Press, 2009.

Mark R Jerrum, Leslie G Valiant, and Vijay V Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theor. Comput. Sci.*, 43(2-3):169–188, 1986.

Richard M. Karp, Michael Luby, and Neal Madras. Monte-carlo approximation algorithms for enumeration problems. *J. Algorithms*, 10(3):429–448, 1989.

D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

Lukas Kroc, Ashish Sabharwal, and Bart Selman. Leveraging belief propagation, backtrack search, and statistics for model counting. *Annals OR*, 184(1):209–231, 2011.

Steffen L Lauritzen and Frank Jensen. Stable local computation with conditional gaussian distributions. *Statistics and Computing*, 11(2):191–203, 2001.

David J Lunn, Andrew Thomas, Nicky Best, and David Spiegelhalter. Winbugs – a Bayesian modelling framework: concepts, structure, and extensibility. *Statistics and computing*, 10(4):325–337, 2000.

Christian Muise, Sheila A McIlraith, J Christopher Beck, and Eric I Hsu. Dsharp: fast d-DNNF compilation with sharpSAT. In *Advances in Artificial Intelligence*, pages 356–361. Springer, 2012.

Kevin P Murphy. A variational approximation for Bayesian networks with discrete and continuous latent variables. In *UAI*, pages 457–466, 1999.

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

Scott Sanner and Ehsan Abbasnejad. Symbolic variable elimination for discrete and continuous graphical models. In *AAAI*, 2012.

Prakash P Shenoy and James C West. Inference in hybrid Bayesian networks using mixtures of polynomials. *International Journal of Approximate Reasoning*, 52(5):641–657, 2011.

Michael Sipser. A complexity theoretic approach to randomness. In *STOC*, pages 330–335. ACM, 1983.

Larry Stockmeyer. The complexity of approximate counting. In *STOC*, pages 118–126, New York, NY, USA, 1983. ACM.

Dan Suciu, Dan Olteanu, Christopher Ré, and Christoph Koch. Probabilistic databases. *Synthesis Lectures on Data Management*, 3(2):1–180, 2011.

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

Guy Van den Broeck, Nima Taghipour, Wannes Meert, Jesse Davis, and Luc De Raedt. Lifted probabilistic inference by first-order knowledge compilation. In *IJCAI*, pages 2178–2185, 2011.

Shenlong Wang, Alexander G. Schwing, and Raquel Urtasun. Efficient inference of continuous Markov random fields with polynomial potentials. In *NIPS*, pages 936–944, 2014.