First-Order Model Counting in a Nutshell

Guy Van den Broeck
Computer Science Department
University of California, Los Angeles
guyvdb@cs.ucla.edu

Abstract

First-order model counting recently emerged as a computational tool for high-level probabilistic reasoning. It is concerned with counting satisfying assignments to sentences in first-order logic and upgrades the successful propositional model counting approaches to probabilistic reasoning. We give an overview of model counting as it is applied in statistical relational learning, probabilistic programming, databases, and hybrid reasoning. A short tutorial illustrates the principles behind these solvers. Finally, we show that first-order counting is a fundamentally different problem from the propositional counting techniques that inspired it.

1 Why First-Order Model Counting?

Model counting comes in many flavors, depending on the form of logic being used. In *propositional model counting*, or #SAT, one counts the number of assignments ω that satisfy a propositional sentence Δ, denoted $\omega \models \Delta$ [Gomes *et al.*, 2009]. In weighted model counting (WMC), one accords a weight to every model, and computes the sum of the weights of all models. The weight of a model is often factorized into weights of assignments to individual variables. 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], and factor graphs [Choi *et al.*, 2013]. Exact WMC solvers are based on knowledge compilation [Darwiche, 2004; Muise *et al.*, 2012] or exhaustive DPLL search [Sang *et al.*, 2005]. (Approximate WMC algorithms use local search [Wei and Selman, 2005] or sampling [Chakraborty *et al.*, 2014; Ermon *et al.*, 2014].)

The popularity of WMC can be explained as follows. Its formulation elegantly decouples the logical or symbolic representation from the statistical or numeric representation, which is encapsulated in the weight function. This is a separation of concerns that is quite familiar in AI:

$$\text{Prob. Distribution} = \text{Qualitative} + \text{Quantitative}$$
$$\text{Bayesian network} = \text{DAG} + \text{CPTs}$$
$$\text{Factor Graph} = \text{Bipartite Graph} + \text{Potentials}$$

Weighted model counting takes this to another level, where any logical language can now specify the qualitative structural properties of what constitutes a model. Independently, the weight function quantifies how probable each model is in the probability space.

$$\text{Prob. Distribution} = \text{Logic Sentence} + \text{Weights}$$
$$\text{WMC} = \text{SAT Formula} + \text{Weights}$$

Formally, we have the following.

Definition 1 (Weighted Model Count)**.** The WMC of
- a *sentence* Δ in propositional logic over literals $\mathcal{L}$, and
- a *weight function* $\mathrm{w} : \mathcal{L} \to \mathbb{R}$,

is defined as $\mathrm{WMC}(\Delta, \mathrm{w}) = \sum_{\omega \models \Delta} \prod_{l \in \omega} \mathrm{w}(l)$.

One benefit of this approach is that it leverages decades of work on formal semantics and logical reasoning. When building solvers, this allows us to reason about logical equivalence and reuse solver technology (such as constraint propagation and clause learning). WMC also naturally reasons about deterministic, hard constraints in a probabilistic context.

First-Order Generalizations of Model Counting

The model counting philosophy has recently transformed several areas of uncertainty reasoning.

First-Order Logic Within statistical relational learning [Getoor and Taskar, 2007], lifted inference algorithms aim for efficient probabilistic inference in relational models [Poole, 2003], in particular for Markov logic networks (MLNs) [Richardson and Domingos, 2006]. State-of-the art lifted inference algorithms reduce the problem to a *weighted first-order model counting* (WFOMC) problem, where the models (satisfying assignments) are defined by a sentence in finite-domain first-order logic [Van den Broeck *et al.*, 2011; Gogate and Domingos, 2011].

$$\text{WFOMC} = \text{First-Order Sentence} + \text{Weights}$$

A classical example is the first-order sentence

$$\forall x, \forall y,\ \texttt{Smokes}(x) \wedge \texttt{Friends}(x, y) \Rightarrow \texttt{Smokes}(y).$$

over a finite domain of n people. First-order model counting asks how many friendship and smokes relations exist that satisfy the formula. WFOMC additionally assigns a weight to each of these models to represent a distribution over them.

Logic Programs In the closely-related field of *probabilistic programming*, the goal is to augment programming languages with the ability to represent probability distributions. One prominent approach augments logic programming languages with probabilities. In their purest form, these representations express the distribution as the models of a logic program, with associated weights [Fierens *et al.*, 2015].

$$\text{Problog} = \text{Logic Program} + \text{Weights}$$

Consider for example, the following inductive definition.

$$\texttt{Smokes}(x) \texttt{ :- } \texttt{Stress}(y).$$
$$\texttt{Smokes}(x) \texttt{ :- } \texttt{Friends}(x, y), \texttt{Smokes}(y).$$

The program defines a set of models (in a higher-order logic). By assigning weights, for example to express the probability that people have stress and are friends, we can capture probabilistic spread of influence through a social network. This view of probabilistic programming has indeed spurred the development of probabilistic inference algorithms that use advanced logical reasoning techniques [Vlasselaer *et al.*, 2015].

Databases Another form of first-order model counting can be found in the probabilistic database community [Suciu *et al.*, 2011]. Here, the database query is a sentence in first-order logic (usually a monotone DNF sentence, called UCQ). The goal is to compute the probability of the query given a database that assigns probabilities to tuples.

$$\text{Prob. Database} = \text{First-Order Query} + \text{Tuple Weights}$$

Within this field, the model counting perspective has shown to be fruitful, for identifying logical transformations that empower the probabilistic reasoning system. For example, first-order resolution on the query can turn hard queries into easy ones [Gribkoff *et al.*, 2014]. Moreover, efficient first-order solvers can significantly speed up querying of real-world knowledge graphs [Ceylan *et al.*, 2016].

SMT One limitation of the WMC approaches above is that they operate on discrete distributions. This limitation is addressed in Belle *et al.*; Chistikov *et al.* [2015a; 2015]. The notion of *weighted model integration* (WMI) is based on *satisfiability modulo theories* (SMT), which enable us to, for example, reason about the satisfiability of linear constraints over the rationals ($\mathcal{LRA}$). The WMI task is defined on the models of an SMT theory Δ, containing mixtures of Boolean and continuous variables.

$$\text{WMI} = \text{SMT}(\mathcal{LRA}) + \text{Weights}$$

For every assignment to the Boolean and continuous variables, the WMI problem defines a weight, for example as a polynomial density. 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.

For example, the following SMT theory from Belle *et al.* [2015a] talks about travel times j_i on various road segments, and how they relate to the time of day. Three Boolean features f_i are defined. Their weight, together with a density on the continuous variables, defines a probability distribution.

$$f_1 \Leftrightarrow \quad [morn \Rightarrow j_1 + j_2 + j_3 \leq 800]$$
$$f_2 \Leftrightarrow \quad [aft \Rightarrow avg(s_1, s_2, s_3) \geq 80]$$
$$f_3 \Leftrightarrow \quad [morn \vee eve \Rightarrow 700 \leq (j_1 + j_2 + j_3) \leq 900]$$

Overall, weighted SMT theories admit a natural encoding of hybrid Markov and Bayesian networks, analogous to the encodings of discrete graphical networks using WMC.

Again, these encodings empower the probabilistic reasoning algorithms with the semantics and solvers for the logic at hand. For example, Belle *et al.* [2015b] use a highly efficient SMT solver to sample possible worlds from this distribution.

2 A First-Order Reasoning Tutorial

The give a flavor for the reasoning that is required in first-order model counting, we will go through simple examples and identify the necessary principles. Algorithmic details can be found in Van den Broeck [2013]. For the sake of simplicity, the examples are non-weighted.

Exponentiation Consider Δ to be

$$\texttt{Stress}(\mathsf{A}) \Rightarrow \texttt{Smokes}(\mathsf{A}). \tag{1}$$

Assuming a finite domain $\mathbf{D} = \{\mathsf{A}\}$, every assignment to $\texttt{Stress}(\mathsf{A})$ and $\texttt{Smokes}(\mathsf{A})$ satisfies Δ, except when $\texttt{Stress}(\mathsf{A})$ is true and $\texttt{Smokes}(\mathsf{A})$ is false. Therefore, the model count is 3. Now let Δ be

$$\forall x,\ \texttt{Stress}(x) \Rightarrow \texttt{Smokes}(x). \tag{2}$$

Without changing $\mathbf{D}$, the model count is still 3. When we expand $\mathbf{D}$ to n people, we get n independent copies of Formula 1. For each person x, $\texttt{Stress}(x)$ and $\texttt{Smokes}(x)$ can take 3 values, and the total model count is 3^n.

This example already demonstrates the benefits of first-order counting. A *propositional* model counter on Formula 2 would detect that all n clauses are independent, recompute for every clause that it has 3 models, and multiply these counts n times. Propositional model counters have no elementary operation for exponentiation. A *first-order* model counter reads from the first-order structure that it suffices to compute the model count of a single ground clause, and then knows to exponentiate. It never actually grounds the formula, and given the size of $\mathbf{D}$, it runs in logarithmic time. This gives an exponential speedup over propositional counting, which runs in linear time.

These first-order counting techniques can interplay with propositional ones. Take for example Δ to be

$$\forall y,\ \texttt{ParentOf}(y) \wedge \texttt{Female} \Rightarrow \texttt{MotherOf}(y). \tag{3}$$

This sentence is about a specific individual who may be female, depending on the proposition $\texttt{Female}$. We can separately count the models in either case. When $\texttt{Female}$ is false, Δ is satisfied, and the $\texttt{ParentOf}$ and $\texttt{MotherOf}$ atoms can take any value. This gives 4^n models. When $\texttt{Female}$ is true, Δ is structurally identical to Formula 2, and has 3^n models. The total model count is then $3^n + 4^n$.

These concepts can be applied recursively to count more complicated formulas. Take for example

$$\forall x, \forall y,\ \texttt{ParentOf}(x, y) \wedge \texttt{Female}(x) \Rightarrow \texttt{MotherOf}(x, y).$$

There is now a partition of the ground clauses into n independent sets of n clauses, for values of x The formula for each specific x is structurally identical to Formula 3 and has count $3^n + 4^n$. The total model count is then $(3^n + 4^n)^n$.

Counting The most impressive improvements are attained when propositional model counters run in time *exponential* in n, yet first-order model counters run in *polynomial* time. To consider an example where this comes up, let Δ be

$$\forall x, \forall y,\ \texttt{Smokes}(x) \wedge \texttt{Friends}(x, y) \Rightarrow \texttt{Smokes}(y). \quad (4)$$

This time, the clauses in the grounding of Δ are no longer independent, and it would be wrong to exponentiate. Let us first assume we know that k people smoke, and that we know their identities. Then, how many models are there? Formula 4 encodes that a smoker cannot be friends with a non-smoker. Hence, out of n^2 $\texttt{Friends}$ atoms, $k(n-k)$ have to be false, and the others can take either truth value. Thus, the number of models is $2^{n^2-k(n-k)}$. Second, we know that there are $\binom{n}{k}$ ways to choose k smokers, and k can range from 0 to n. This results in the total model count of $\sum_{k=0}^{n} \binom{n}{k} 2^{n^2-k(n-k)}$. Evaluating this formula is polynomial in n. On the other hand, propositional algorithms require time exponential in n.

Skolemization The theories so far had only universal quantifiers. *Skolemization* is the procedure of eliminating existential quantifiers from a theory. Van den Broeck *et al.* [2013] introduce a Skolemization procedure that is sound for the WFOMC task. Suppose that we are eliminating the existential quantifier in the following sentence.

$$\forall p, \exists c, \texttt{Card}(p, c)$$

We can do so without changing the model count. First, introduce a new relation $\texttt{S}$ and replace the sentence by

$$\forall p, \forall c, \texttt{Card}(p, c) \Rightarrow \texttt{S}(p)$$

Second, extend the weight function w with $\mathrm{w}(\texttt{S}(\mathbf{y})) = 1$ and $\mathrm{w}(\neg\texttt{S}(\mathbf{y})) = -1$ for all $\mathbf{y}$.

We can verify this transformation as follows. For a fixed position p, consider two cases: $\exists c, \texttt{Card}(p, c)$ is either true or false. If it is true, then so is $\texttt{S}(p)$. All models of the Skolemized sentence are also models of the original sentence, and the models have the same weight. If $\exists c, \texttt{Card}(p, c)$ is false, then this does not correspond to any model of the original sentence. However, the Skolemized sentence is satisfied, and $\texttt{S}(p)$ can be true or false. Yet, for every model with weight w where $\texttt{S}(p)$ is true, there is also a model with weight $-w$ where $\texttt{S}(p)$ is false. These weights cancel out in the WFOMC, and the transformation is sound.

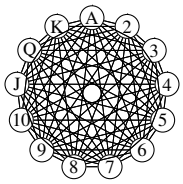

Figure 1: Graphical model for a shuffled deck of 13 cards

3 Why First-Order Model Counters?

Consider a randomly shuffled deck of 52 playing cards. Suppose that we are dealt the top card, and we want to answer: what is the probability that we get hearts? When the dealer reveals that the bottom card is black, how does our probability change? Basic statistics says it increases from $1/4$ to $13/51$.

To represent the distribution over all shuffled decks with a probabilistic graphical model, a natural choice is to have 52 random variables, one for every card. Each variable can take any of 52 values, one for every position in the deck. When the queen of hearts takes the top position, none of the other random variables are allowed to take that position. Such constraints are enforced by adding a factor between every pair of variables, setting the probability to zero that two cards are in the same position. Figure 1 depicts the graphical model for a small deck of 13 cards.

The graphical model for this problem is a completely connected. This means that classical inference algorithms will require time and space that is exponential in the number of playing cards. Indeed, for a full deck of cards, they will build a table with 52^{52} rows. The underlying reason for this poor performance is that the distribution has no conditional or contextual independencies. Our belief about the top card is affected by any new observation on the remaining cards. The reason why humans can still answer the above queries efficiently is different: the distribution exhibits exchangeability [Niepert and Van den Broeck, 2014]. In fact, Van den Broeck [2015] show that this distribution is intractable for any reasoning system that does not make the symmetries in this distribution explicit. More generally, Beame *et al.* [2015] study the complexity of first-order model counting.

The problem with this reasoning task is not one of representation. Many statistical relational languages, including Markov logic, can express the distribution concisely. It can even be written in classical first-order logic as

$$\forall p, \exists c,\ \ \texttt{Card}(p, c) \qquad \forall c, \exists p,\ \ \texttt{Card}(p, c)$$
$$\forall p, \forall c, \forall c', \neg\texttt{Card}(p, c) \vee \neg\texttt{Card}(p, c') \vee c = c', \quad (5)$$

where $\texttt{Card}(p, c)$ denotes that position p contains card c. Using first-order model counting, it can be solved efficiently as

$$\#SAT = \sum_{k=0}^{n} \binom{n}{k} \sum_{l=0}^{n} \binom{n}{l} (l+1)^k (-1)^{2n-k-l} = n!$$

This expression is evaluated in time polynomial in the number of cards n. It shows that lifted inference is strictly more powerful than propositional reasoning. The key difference is that WFOMC can assume certain symmetries that propositional techniques cannot take into account.

4 Conclusions

First-order model counting is an approach to probabilistic reasoning that has recenlty been applied to many representations of uncertainty, each time leading to highly efficient solvers. It presents a synthesis of ideas from various subfields of AI, for graphical models to knowledge representation and reasoning, and connects to fields outside of AI, in verification, databases, and theory.

References

[Beame *et al.*, 2015] Paul Beame, Guy Van den Broeck, Eric Gribkoff, and Dan Suciu. Symmetric weighted first-order model counting. In *Proceedings PODS*, 2015.

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

[Belle *et al.*, 2015b] Vaishak Belle, Guy Van den Broeck, and Andrea Passerini. Hashing-based approximate probabilistic inference in hybrid domains. In *UAI*, 2015.

[Ceylan *et al.*, 2016] Ismail Ilkan Ceylan, Adnan Darwiche, and Guy Van den Broeck. Open-world probabilistic databases. In *Proceedings of KR*, 2016.

[Chakraborty *et al.*, 2014] Supratik Chakraborty, Daniel J Fremont, Kuldeep S Meel, Sanjit A Seshia, and Moshe Y Vardi. Distribution-aware sampling and weighted model counting for sat. *Proceedings of AAAI*, 2014.

[Chavira and Darwiche, 2008] Mark Chavira and Adnan Darwiche. On probabilistic inference by weighted model counting. *Artificial Intelligence*, 2008.

[Chavira *et al.*, 2006] Mark Chavira, Adnan Darwiche, and Manfred Jaeger. Compiling relational Bayesian networks for exact inference. *IJAR*, 42(1-2):4–20, May 2006.

[Chistikov *et al.*, 2015] Dmitry Chistikov, Rayna Dimitrova, and Rupak Majumdar. Approximate counting in smt and value estimation for probabilistic programs. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 9035 of *Lecture Notes in Computer Science*, pages 320–334. Springer Berlin Heidelberg, 2015.

[Choi *et al.*, 2013] Arthur Choi, Doga Kisa, and Adnan Darwiche. Compiling probabilistic graphical models using sentential decision diagrams. In *ECSQARU*. 2013.

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

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

[Fierens *et al.*, 2015] 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. *TPLP*, 2015.

[Getoor and Taskar, 2007] L. Getoor and B. Taskar, editors. *An Introduction to Statistical Relational Learning*. MIT Press, 2007.

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

[Gomes *et al.*, 2009] Carla P Gomes, Ashish Sabharwal, and Bart Selman. Model counting. *Handbook of Satisfiability*, 185:633–654, 2009.

[Gribkoff *et al.*, 2014] Eric Gribkoff, Dan Suciu, and Guy Van den Broeck. Lifted probabilistic inference: A guide for the database researcher. *Bulletin of the Technical Committee on Data Engineering*, 2014.

[Muise *et al.*, 2012] 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.

[Niepert and Van den Broeck, 2014] Mathias Niepert and Guy Van den Broeck. Tractability through exchangeability: A new perspective on efficient probabilistic inference. *Proceedings of AAAI*, 2014.

[Poole, 2003] David Poole. First-order probabilistic inference. In *Proc. IJCAI*, 2003.

[Richardson and Domingos, 2006] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine learning*, 62(1-2):107–136, 2006.

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

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

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

[Van den Broeck *et al.*, 2013] Guy Van den Broeck, Wannes Meert, and Adnan Darwiche. Skolemization for weighted first-order model counting. *Proceedings of UAI*, 2013.

[Van den Broeck, 2013] Guy Van den Broeck. *Lifted Inference and Learning in Statistical Relational Models*. PhD thesis, KU Leuven, January 2013.

[Van den Broeck, 2015] Guy Van den Broeck. Towards high-level probabilistic reasoning with lifted inference. In *Proceedings of KRR*, 2015.

[Vlasselaer *et al.*, 2015] Jonas Vlasselaer, Guy Van den Broeck, Angelika Kimmig, Wannes Meert, and Luc De Raedt. Anytime inference in probabilistic logic programs with Tp-compilation. In *Proceedings of IJCAI*, 2015.

[Wei and Selman, 2005] W. Wei and B. Selman. A new approach to model counting. In *Theory and Applications of Satisfiability Testing*, pages 96–97. Springer, 2005.