Generalising Weighted Model Counting

Paulius Dilkas

University of Edinburgh, UK

1st November 2022

THE UNIVERSITY of EDINBURGH informatics

EDINBURGH CENTRE FOR ROBOTICS

UKRI Engineering and Physical Sciences Research Council

Weighted Model Counting

Example

We have a biased coin that has a probability $p \in [0, 1]$ of landing heads. What is the probability that it lands heads at least once if we toss it three times?

In Propositional Logic...

- Formula: $x_1 \vee x_2 \vee x_3$
- Weights: $w(x_i) = p$, $w(\neg x_i) = 1 - p$ for $i = 1, 2, 3$
- Models: $\mathcal{P}(\{ x_1, x_2, x_3 \}) \setminus \{ \emptyset \}$

In First-Order Logic...

- Formula: $\exists x \in \{ 1, 2, 3 \}.\ P(x)$
- Weights: $w(P) = p$, $w(\neg P) = 1 - p$
- Models: $\mathcal{P}(\{ P(1), P(2), P(3) \}) \setminus \{ \emptyset \}$

Significance

Applications

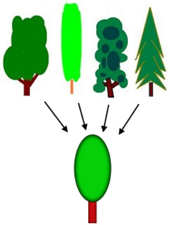
- Probabilistic inference: graphical models, statistical relational models, probabilistic programming
- Neural-symbolic artificial intelligence
- Bioinformatics
- Robotics
- Natural language processing
- Enumerative combinatorics

Impact

- Suitable WMC algorithm
- Appropriate input format
- Lifted reasoning
- Expressive data structures

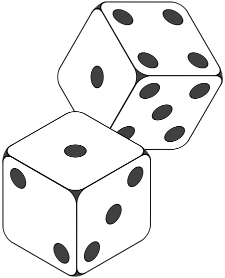
- $> 100\times$ speedup
- provable tractability

Contributions



Generalising Representations

- Beyond weights on literals
- Circuits for recursion



Random-Instance Experiments

- Application-specific parameters
  - ProbLog predicates, arities
- Parameters of hardness
  - density, primal treewidth

Generalising Representations

WMC and Measures on Boolean Algebras

Definition

A measure is a function $\mu\colon \mathcal{P}(\mathcal{P}(X)) \to \mathbb{R}_{\geq 0}$ such that:

- $\mu(\bot) = 0$;
- $\mu(x \vee y) = \mu(x) + \mu(y)$ whenever $x \wedge y = \bot$.

Observation

WMC corresponds to the process of calculating the value of $\mu(x)$ for some $x \in \mathcal{P}(\mathcal{P}(X))$.

Observation

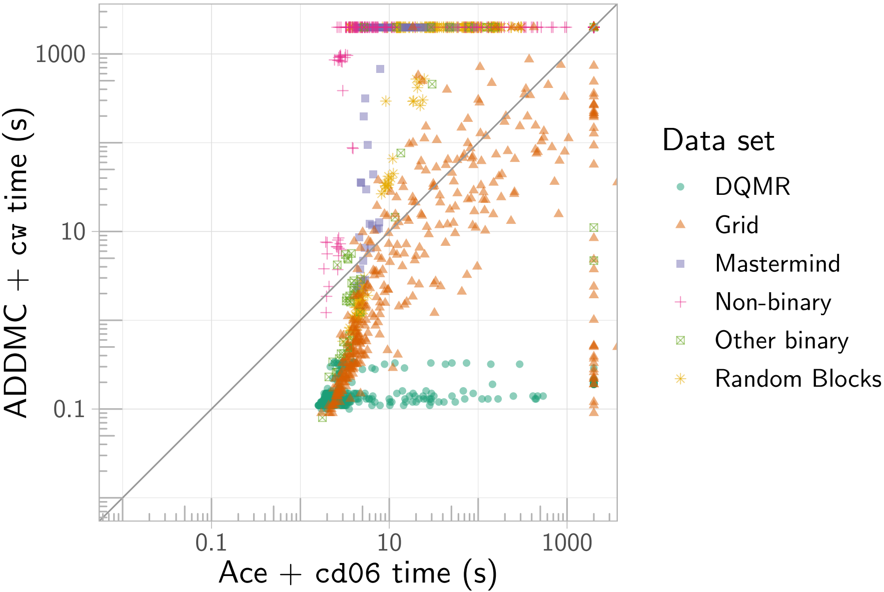
Classical WMC is only able to evaluate factorable measures (c.f., a collection of mutually independent random variables).

Theorem (Informal Version)

It is always possible to add more variables to turn a non-factorable measure into a factorable measure.

However, that is not necessarily a good idea!

Experiments with Bayesian Networks



Transforming Known WMC Encodings into PBP

For any propositional formula ϕ over a set of variables X and $p, q \in \mathbb{R}$, let $[\phi]^p_q \colon 2^X \to \mathbb{R}$ be the pseudo-Boolean function defined as

$$[\phi]^p_q(Y) \coloneqq \begin{cases} p & \text{if } Y \models \phi \\ q & \text{otherwise} \end{cases}$$

for any $Y \subseteq X$.

Transforming Known WMC Encodings into PBP

For any propositional formula ϕ over a set of variables X and $p, q \in \mathbb{R}$, let $[\phi]^p_q \colon 2^X \to \mathbb{R}$ be the pseudo-Boolean function defined as

$$[\phi]^p_q(Y) \coloneqq \begin{cases} p & \text{if } Y \models \phi \\ q & \text{otherwise} \end{cases}$$

for any $Y \subseteq X$.

Example

Clauses
$\neg x \Rightarrow p$
$p \Rightarrow \neg x$
$x \Rightarrow q$
$q \Rightarrow x$
$\neg x$

Transforming Known WMC Encodings into PBP

For any propositional formula ϕ over a set of variables X and $p, q \in \mathbb{R}$, let $[\phi]^p_q \colon 2^X \to \mathbb{R}$ be the pseudo-Boolean function defined as

$$[\phi]^p_q(Y) \coloneqq \begin{cases} p & \text{if } Y \models \phi \\ q & \text{otherwise} \end{cases}$$

for any $Y \subseteq X$.

Example

Clauses	In CNF
$\neg x \Rightarrow p$	$x \vee p$
$p \Rightarrow \neg x$	$\neg x \vee \neg p$
$x \Rightarrow q$	$\neg x \vee q$
$q \Rightarrow x$	$x \vee \neg q$
$\neg x$	$\neg x$

Transforming Known WMC Encodings into PBP

For any propositional formula ϕ over a set of variables X and $p, q \in \mathbb{R}$, let $[\phi]^p_q \colon 2^X \to \mathbb{R}$ be the pseudo-Boolean function defined as

$$[\phi]^p_q(Y) \coloneqq \begin{cases} p & \text{if } Y \models \phi \\ q & \text{otherwise} \end{cases}$$

for any $Y \subseteq X$.

Example

Clauses	In CNF	Pseudo-Boolean Functions
$\neg x \Rightarrow p$	$x \vee p$	$[\neg x]^{0.2}_1$
$p \Rightarrow \neg x$	$\neg x \vee \neg p$	
$x \Rightarrow q$	$\neg x \vee q$	$[x]^{0.8}_1$
$q \Rightarrow x$	$x \vee \neg q$	
$\neg x$	$\neg x$	$[\neg x]^1_0$

Transforming Known WMC Encodings into PBP

For any propositional formula ϕ over a set of variables X and $p, q \in \mathbb{R}$, let $[\phi]^p_q \colon 2^X \to \mathbb{R}$ be the pseudo-Boolean function defined as

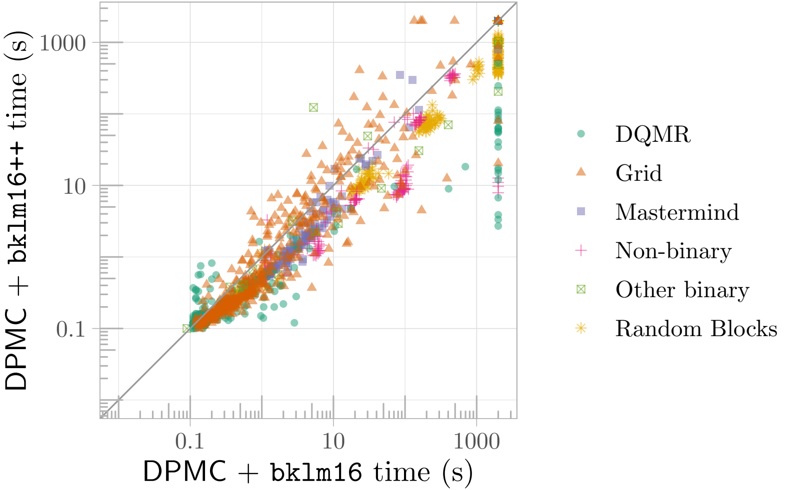
$$[\phi]^p_q(Y) := \begin{cases} p & \text{if } Y \models \phi \\ q & \text{otherwise} \end{cases}$$

for any $Y \subseteq X$.

Example

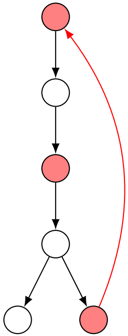
Clauses	In CNF	Pseudo-Boolean Functions	
$\neg x \Rightarrow p$	$x \vee p$	$[\neg x]^{0.2}_1$	
$p \Rightarrow \neg x$	$\neg x \vee \neg p$		$[x]^{0.8}_{0.2}$
$x \Rightarrow q$	$\neg x \vee q$	$[x]^{0.8}_1$	
$q \Rightarrow x$	$x \vee \neg q$		
$\neg x$	$\neg x$	$[\neg x]^1_0$	$[\neg x]^1_0$

Some Instances Become Tractable as a Result



First-Order Logic and Recursive Computations

Example (Counting $P\colon M \to N$ Injections)



Input Formula

$$\forall x \in M.\ \exists y \in N.\ P(x, y)$$
$$\forall x \in M.\ \forall y, z \in N.\ P(x, y) \wedge P(x, z) \Rightarrow y = z$$
$$\forall w, x \in M.\ \forall y \in N.\ P(w, y) \wedge P(x, y) \Rightarrow w = x$$

Recursive Solution

$$f(m, n) = \begin{cases} 1 & \text{if } m = 0 \text{ and } n = 0 \\ 0 & \text{if } m > 0 \text{ and } n = 0 \\ f(m, n-1) + m \cdot f(m-1, n-1) & \text{otherwise.} \end{cases}$$

First-Order Knowledge Compilation

Workflow Before

1. Compile the formula to a circuit
2. Evaluate to get the answer

Workflow After

1. Compile the formula to a graph
2. Extract the definitions of functions
3. Simplify
4. Supplement with base cases
5. Evaluate to get the answer

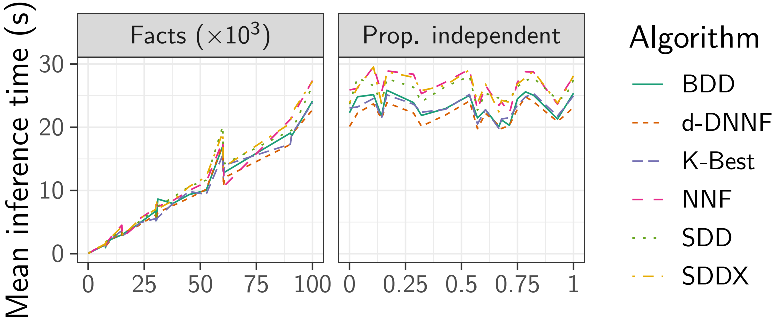
Random-Instance Experiments

A Constraint Model for (Probabilistic) Logic Programs

0.2 :: stress(P) :- person(P).
0.3 :: influences(P_1, P_2) :- friend(P_1, P_2).
0.1 :: cancer_spont(P) :- person(P).
0.3 :: cancer_smoke(P) :- person(P).
smokes(X) :- stress(X).
smokes(X) :- smokes(Y), influences(Y, X).
cancer(P) :- cancer_spont(P).
cancer(P) :- smokes(P), cancer_smoke(P).
person(*mary*).
person(*albert*).
friend(*albert*, *mary*).

- predicates, arities
- variables
- constants
- probabilities
- length
- complexity

ProbLog Inference Algorithms on Random Instances



Random WMC Instances

Key Idea

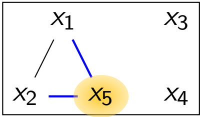
Parameter $\rho \in [0, 1]$ biases the probability distribution towards adding variables that would introduce fewer new edges in the primal graph.

Example

Partially-constructed formula:

$$(\neg x_5 \vee x_2 \vee x_1) \wedge (x_5 \vee ? \vee ?).$$

Its primal graph:



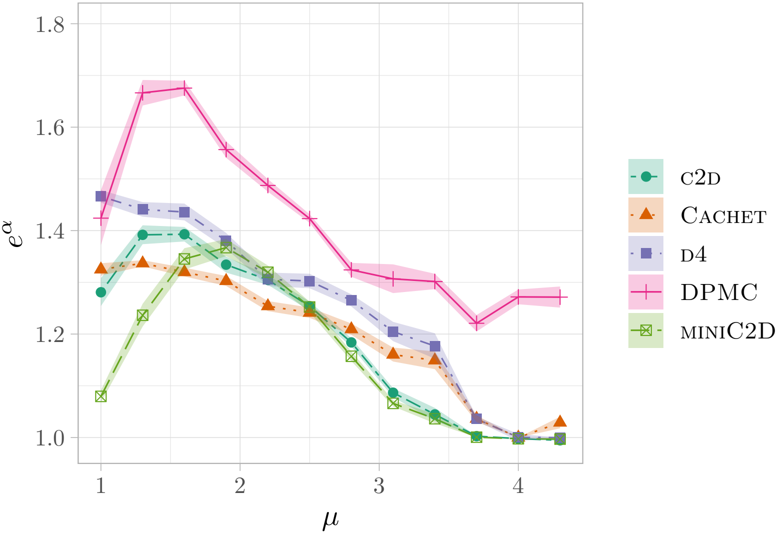
Base probability of each variable being chosen:

$$\frac{1-\rho}{4}.$$

Both x_1 and x_2 get a bonus probability of $\rho/2$ for each being the endpoint of one out of the two neighbourhood edges.

How WMC Algorithms Scale w.r.t. Primal Treewidth

We fit the model $\ln t \sim \alpha w + \beta$, i.e., $t \sim e^{\beta}(e^{\alpha})^{w}$, where t is runtime, and w is primal treewidth.



What Have We Learned?

- Pseudo-Boolean functions as an alternative to literal weights
- Cycles in graphs that encode recursive calls
- WMC is not always the bottleneck in probabilistic inference
- WMC algorithms based on algebraic decision diagrams are fundamentally different:
  - they can supports non-literal weights
  - their running time depends on the numerical values of weights
  - they peak at higher density
  - they scale worse w.r.t. primal treewidth