On Exact Sampling in the Two-Variable Fragment of First-Order Logic

Yuanhong Wang, Juhua Pu, Yuyi Wang, Ondrej Kuzelka
Outline

1. The Counting Problem
2. The Sampling Problem
3. Main Results
4. Applications
The Counting Problem
FOMC

- FOMC = First-Order Model Counting
- FOMC is the problem of counting the number of models of a given sentence over a given set of domain elements.
FOMC: Examples

\[ \varphi = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \]

\[ FOMC(\varphi, [n]) = \# \text{ undirected graphs on } n \text{ vertices} \]
FOMC: Examples

\[ \varphi = \forall x \exists^{=1} y \ R(x, y) \]

\[ FOMC(\varphi, [n]) = \# \text{ functions on a set of } n \text{ elements} \]
FOMC: Examples

\[ \varphi = \forall x \exists^1 y R(x, y) \land \forall y \exists^1 x R(x, y) \]

\[ \text{FOMC}(\varphi, [n]) = \# \text{ permutations on a set of } n \text{ elements} \]

**Note:** Many combinatorics problems can be solved in this way. For instance, we showed recently that FOMC algorithms can be used to automatically construct a database of “interesting” combinatorial sequences [Svatos et al, IJCAI 2023].
WFOMC

- WFOMC computes the weighted count of models of a given sentence over a given set of domain elements, where the weight of a model $\omega$ is given by

$$W(\omega) = \prod_{a \in HB: \omega \models a} w(Pred(a)) \cdot \prod_{a \in HB: \omega \not\models \neg a} \overline{w}(Pred(a))$$

- WFOMC has applications in statistical relational learning.
Tractable Fragments

• Starting with the work of Van den Broeck [2011], several fragments of FOL were identified as tractable for WFOMC ("domain-liftable"):

  • $\text{FO}^2$ [Van den Broeck, 2011, Van den Broeck, Meert & Darwiche, 2014], $\text{S}^2\text{FO}^2$ and $\text{S}^2\text{RU}$ [Kazemi, Kimmig, Van den Broeck, Poole, 2016], $\text{FO}^2 + 1\text{Func}$ [Kuusisto & Lutz, 2018], $\text{C}^2$ [Kuzelka, 2021], $\text{C}^2 + \text{Tree}$ [van Bremen & Kuzelka, 2021], $\text{C}^2 + \text{LinOrder}$ [Toth & Kuzelka, 2023]….

• The area studying such tractable fragments is known as lifted inference.
The Sampling Problem
• **FOMS** = First-Order Model Sampling

• **Problem:** Given a first-order logic sentence $\Gamma$ and a set of domain elements $\Delta$, sample a model of $\Gamma$ over $\Delta$ uniformly at random.
FOMS: An Example

- Example:

$$\Gamma = \forall x \forall y : E(x, y) \Rightarrow E(y, x) \land \forall x : \neg E(x, x)$$
$$\Delta = \{1, 2, 3, \ldots, n\}$$

In this case, it is equivalent to sampling undirected graphs on $n$ vertices uniformly at random.

For instance:

$$\Delta = \{1, 2, 3, \ldots, 10\}$$
WFOMS

- **WFOMS** = Weighted First-Order Model Sampling

- **Problem:** Given a first-order logic sentence $\Gamma$ and a set of domain elements $\Delta$, sample a model $\omega$ of $\Gamma$ over $\Delta$ with probability proportional to its weight:

  \[ W(\omega) = \prod_{a \in H_B: \omega \models a} w(Pred(a)) \cdot \prod_{a \in H_B: \omega \not\models \neg a} \overline{w}(Pred(a)). \]

- It can be used, e.g., to sample from Markov Logic Networks.

- **Note:** In what follows we will talk about FOMS for simplicity but all results will hold for WFOMS as well.
Using FOMC for FOMS?

• **Idea:** In the propositional setting, one can sample as follows given an oracle for model counting (Jerrum et al, 1986):

For $i = 1, \ldots, n$

1. Pick $x_i$ and compute $C_o = MC(\Gamma \land x_i = 0)$,
   
   $C_2 = MC(\Gamma \land x_i = 1)$.

2. Sample the value $v_i$ from $\text{Bernoulli}(C_1/(C_0 + C_1))$.

3. Set $\Gamma := \Gamma \land x_i = v_i$. 
Using FOMC for FOMS?

- **Idea:** In the propositional setting, one can sample as follows given an oracle for model counting (Jerrum et al, 1986):

  For $i = 1, \ldots, n$

  1. Pick $x_i$ and compute $C_o = MC(\Gamma \land x_i = 0)$, $C_2 = MC(\Gamma \land x_i = 1)$.
  2. Sample the value $\nu_i$ from $Bernoulli(C_1/(C_0 + C_1))$.
  3. Set $\Gamma := \Gamma \land x_i = \nu_i$.

- **This does not work for tractable FOMS because conditioning on binary literals is $\#P$-hard [Van den Broeck and Davis, 2012].**
Main Results
Main Results

• **Theorem:** WFOMS can be done in time polynomial in the size of the domain for any $\text{FO}^2$ sentence.

• **Theorem (Stronger version):** WFOMS can be done in time polynomial in the size of the domain for any sentence

$$\Gamma = \Gamma_{\text{FO}^2} \land \bigwedge_{i=1}^{h} \forall x \exists^=^{k_i} y \ R_i(x, y) \land \bigwedge_{j=1}^{l} |R_j| = m_k.$$

• The above already allows us to sample k-regular graphs, but also l-coloured k-regular graphs and so on.

• **Theorem (After LICS):** WFOMS can be done in time polynomial in the size of the domain for any $\mathbf{C}^2$ sentence.
What Makes It Nontrivial

- The difficult part is handling existential quantifiers.

- Without them, the problem was already solved in our last year’s AAAI paper [Wang et al., 2022].

- **Q:** Why are existential quantifiers more difficult?

- **A:** The existing WFOMC algorithms use negative weights (inclusion-exclusion style idea from [Van den Broeck, Meert & Darwiche, 2014]) to support existential quantifiers, but that would make sampling ill-defined.
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y)) \]
\[ \Delta = \{1,2,3,4\} \]
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y)) \]

\[ \Delta = \{1,2,3,4\} \]
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land \text{Blue}(y)) \]

\[ \Delta = \{1, 2, 3, 4\} \]
How It Works

\[ \Gamma = \forall x \neg E(x,x) \land \forall x \forall y E(x,y) \Rightarrow E(y,x) \land \forall x \exists y (E(x,y) \land Blue(y)) \]

\[ \Delta = \{1,2,3,4\} \]
How It Works

$$\Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y))$$

$$\Delta = \{1, 2, 3, 4\}$$

**Iteration: i = 4**
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y)) \]
\[ \Delta = \{1,2,3,4\} \]

Iteration: i = 4
How It Works

\[ \Gamma = \forall x \neg \text{E}(x, x) \land \forall x \forall y \text{E}(x, y) \Rightarrow \text{E}(y, x) \land \forall x \exists y \text{E}(x, y) \land \text{Blue}(y) \] 
\[ \Delta = \{1, 2, 3, 4\} \]

Iteration: i = 4
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y)) \]
\[ \Delta = \{1, 2, 3, 4\} \]

Iteration: i = 3
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y)) \]
\[ \Delta = \{1,2,3,4\} \]

Iteration: i = 3
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land \text{Blue}(y)) \]
\[ \Delta = \{1, 2, 3, 4\} \]

**Iteration: i = 3**
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y)) \]
\[ \Delta = \{1,2,3,4\} \]

Iteration: i = 2
How It Works

$$\Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y))$$

$$\Delta = \{1, 2, 3, 4\}$$

Iteration: i = 2
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land \text{Blue}(y)) \]
\[ \Delta = \{1, 2, 3, 4\} \]

Iteration: i = 2
How It Works

\[ \Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y)) \]

\[ \Delta = \{1,2,3,4\} \]

Iteration: \( i = 2 \)
How It Works

$\Gamma = \forall x \neg E(x, x) \land \forall x \forall y E(x, y) \Rightarrow E(y, x) \land \forall x \exists y (E(x, y) \land Blue(y))$

$\Delta = \{1,2,3,4\}$

We are done!
Applications
It Works

Comparison with a state-of-the-art propositional model sampler UniGen [Soos, Gocht, Meel, 2020].
Generative Models

- Tractable WFOMC can be used for learning statistical-relational learning models, e.g., Markov Logic Networks.

- Now we can also use tractable WFOMS to sample from the trained models exactly.
Other Applications

• Programming-language libraries, e.g. NumPy, provide support for sampling simple combinatorial structures (permutations, combinations etc.)

• We can build a **declarative framework based on WFOMS for sampling more complex combinatorial structures in polynomial time** (those representable in a tractable fragment).

• *We just need to make the algorithms and implementations a “bit” faster.*
Conclusions

- We can sample models of $C^2$ sentences in time polynomial in the domain size.
Yuanhong Wang and Juhua Pu are supported by the National Key R&D Program of China (2021YFB2104800) and the National Science Foundation of China (62177002). Ondrej Kuzelka’s work is supported by the Czech Science Foundation project 20-19104Y, partially also 23-07299S (most of the work was done before the start of the latter project) and by and the OPVVV project CZ.02.1.01/0.0/0.0/16\_019/0000765 “Research Center for Informatics”.