

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

$$\varphi = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x)$$

$FOMC(\varphi, [n]) = \#$ undirected graphs on n vertices

FOMC: Examples

$$\varphi = \forall x \exists^1 y R(x, y)$$

$FOMC(\varphi, [n]) = \#$ functions on a set of n elements

FOMC: Examples

$$\varphi = \forall x \exists^{=1} y R(x, y) \wedge \forall y \exists^{=1} x R(x, y)$$

$FOMC(\varphi, [n]) = \#$ permutations on a set of n 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 ω is given by

$$W(\omega) = \prod_{a \in HB: \omega \models a} w(Pred(a)) \cdot \prod_{a \in HB: \omega \models \neg a} \bar{w}(Pred(a))$$

- 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”):
 - **FO²** [Van den Broeck, 2011, Van den Broeck, Meert & Darwiche, 2014], **S²FO²** and **S²RU** [Kazemi, Kimmig, Van den Broeck, Poole, 2016], **FO² + 1Func** [Kuusisto & Lutz, 2018], **C²** [Kuzelka, 2021], **C² + Tree** [van Bremen & Kuzelka, 2021], **C² + LinOrder** [Toth & Kuzelka, 2023]....
- The area studying such tractable fragments is known as **lifted inference**.

The Sampling Problem

FOMS

- **FOMS** = First-Order Model Sampling
- **Problem:** Given a first-order logic sentence Γ and a set of domain elements Δ , sample a model of Γ over Δ uniformly at random.

FOMS: An Example

- **Example:**

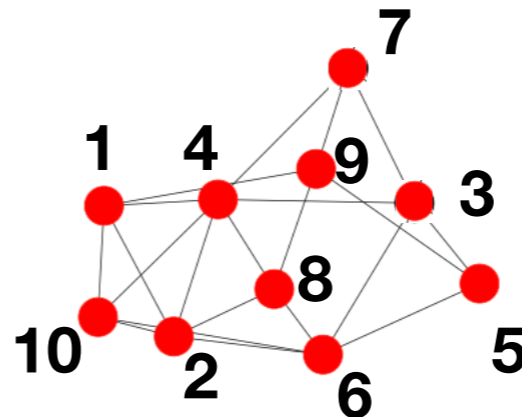
$$\Gamma = \forall x \forall y : E(x, y) \Rightarrow E(y, x) \wedge \forall x : \neg E(x, x)$$

$$\Delta = \{1, 2, 3, \dots, n\}$$

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

For instance:

$$\Delta = \{1, 2, 3, \dots, 10\}$$



WFOMS

- **WFOMS** = Weighted First-Order Model Sampling
- **Problem:** Given a first-order logic sentence Γ and a set of domain elements Δ , sample a model ω of Γ over Δ with probability proportional to its weight:

$$W(\omega) = \prod_{a \in HB: \omega \models a} w(Pred(a)) \cdot \prod_{a \in HB: \omega \models \neg a} \bar{w}(Pred(a)).$$

- 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 $i = 1, \dots, n$

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

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 $i = 1, \dots, n$

1. Pick x_i and compute $C_0 = MC(\Gamma \wedge x_i = 0)$,
 $C_1 = MC(\Gamma \wedge x_i = 1)$.
 2. Sample the value v_i from *Bernoulli*($C_1 / (C_0 + C_1)$).
 3. Set $\Gamma := \Gamma \wedge x_i = v_i$.
- **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 **FO²** sentence.
- **Theorem (Stronger version):** WFOMS can be done in time polynomial in the size of the domain for any sentence

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

- 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 **C²** 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

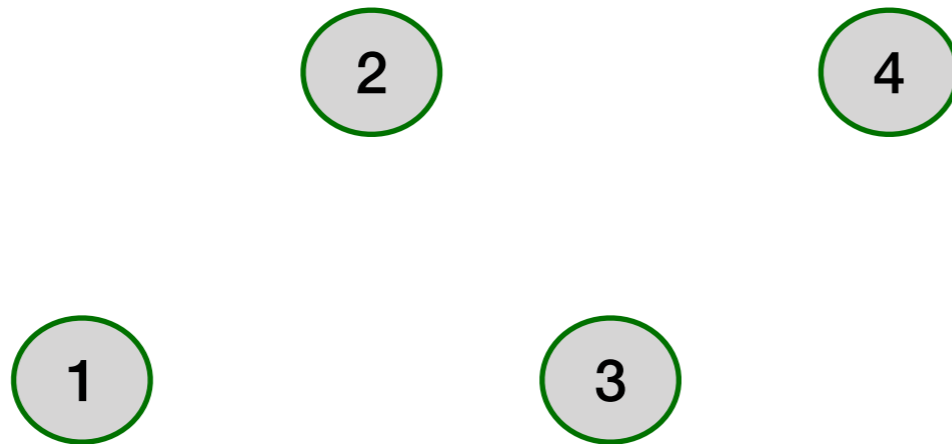
$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \textit{Blue}(y))$$

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

How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \textit{Blue}(y))$$

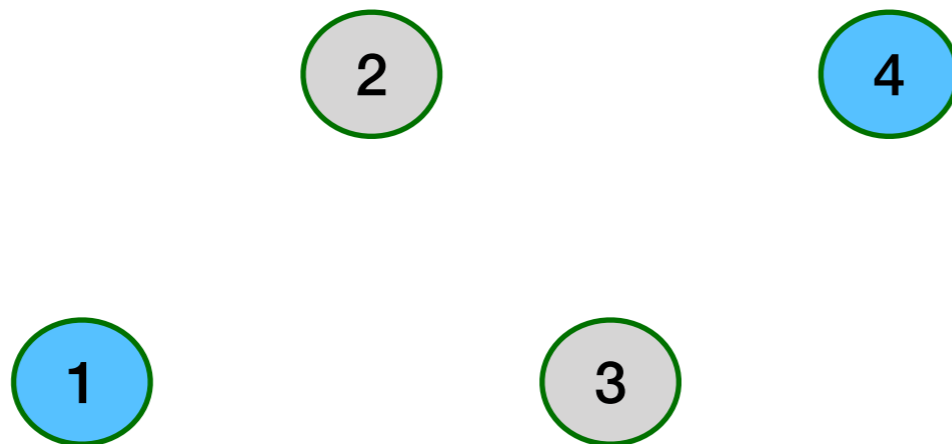
$$\Delta = \{1, 2, 3, 4\}$$



How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \textit{Blue}(y))$$

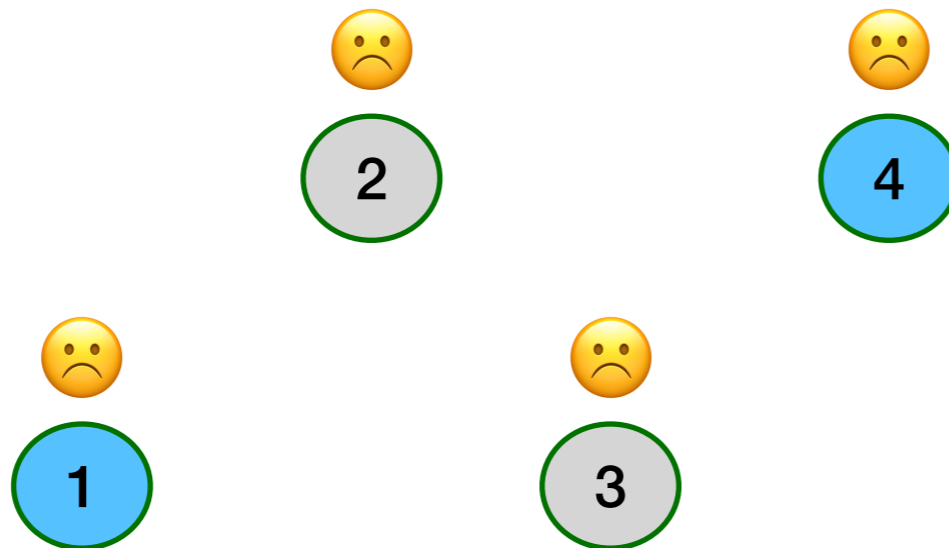
$$\Delta = \{1, 2, 3, 4\}$$



How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \textit{Blue}(y))$$

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

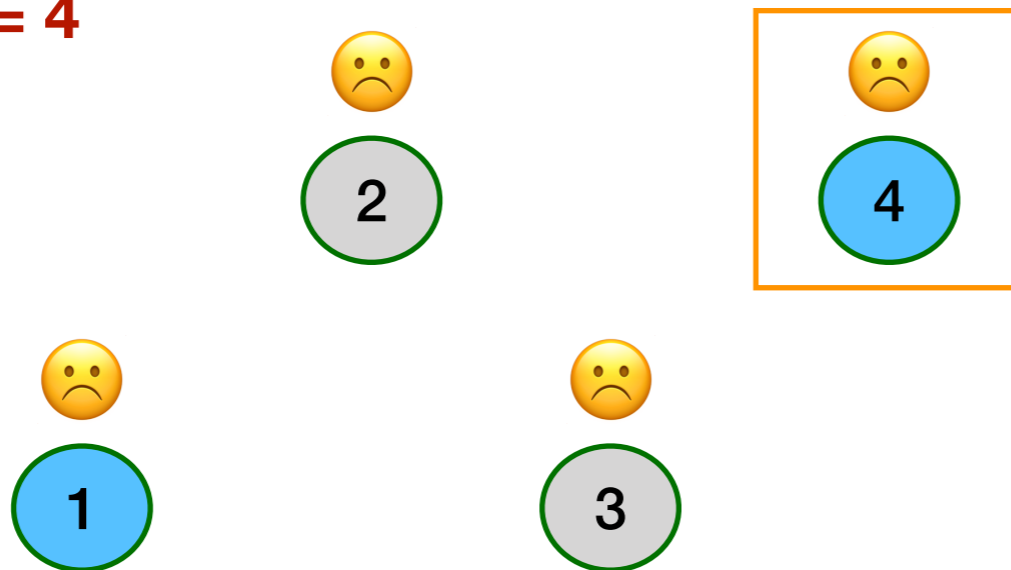


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \textit{Blue}(y))$$

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

Iteration: i = 4

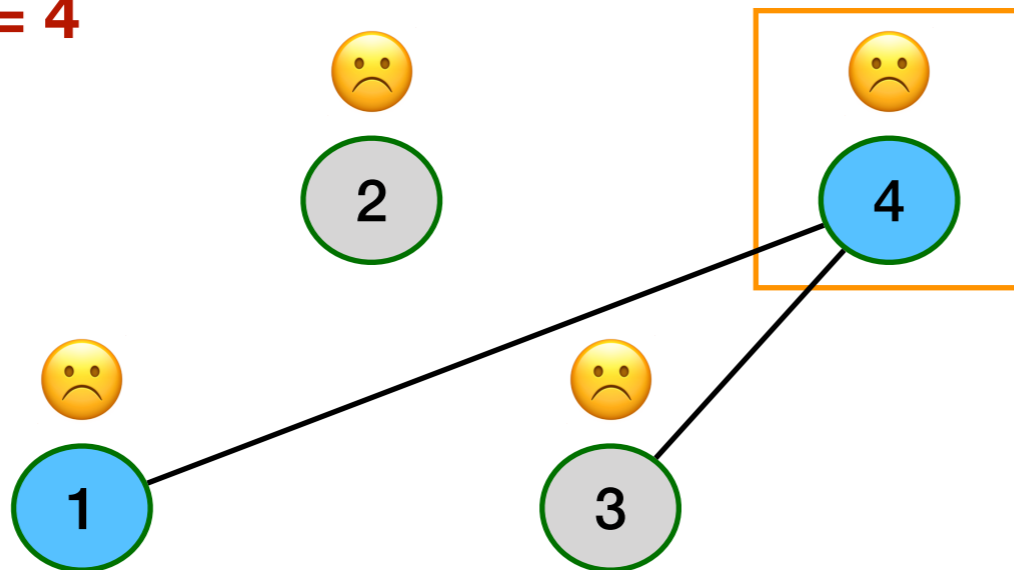


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \text{Blue}(y))$$

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

Iteration: i = 4

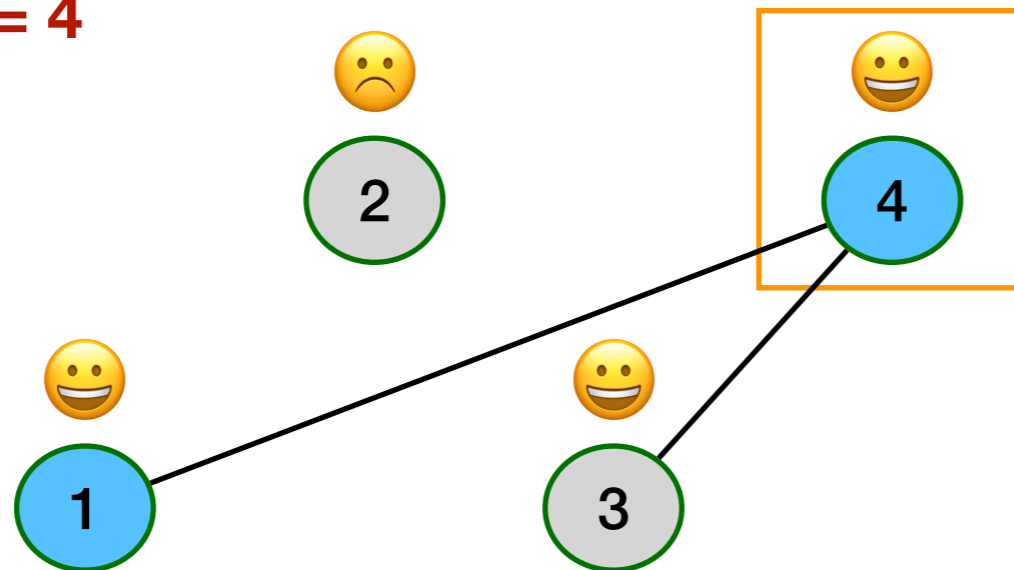


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \text{Blue}(y))$$

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

Iteration: i = 4

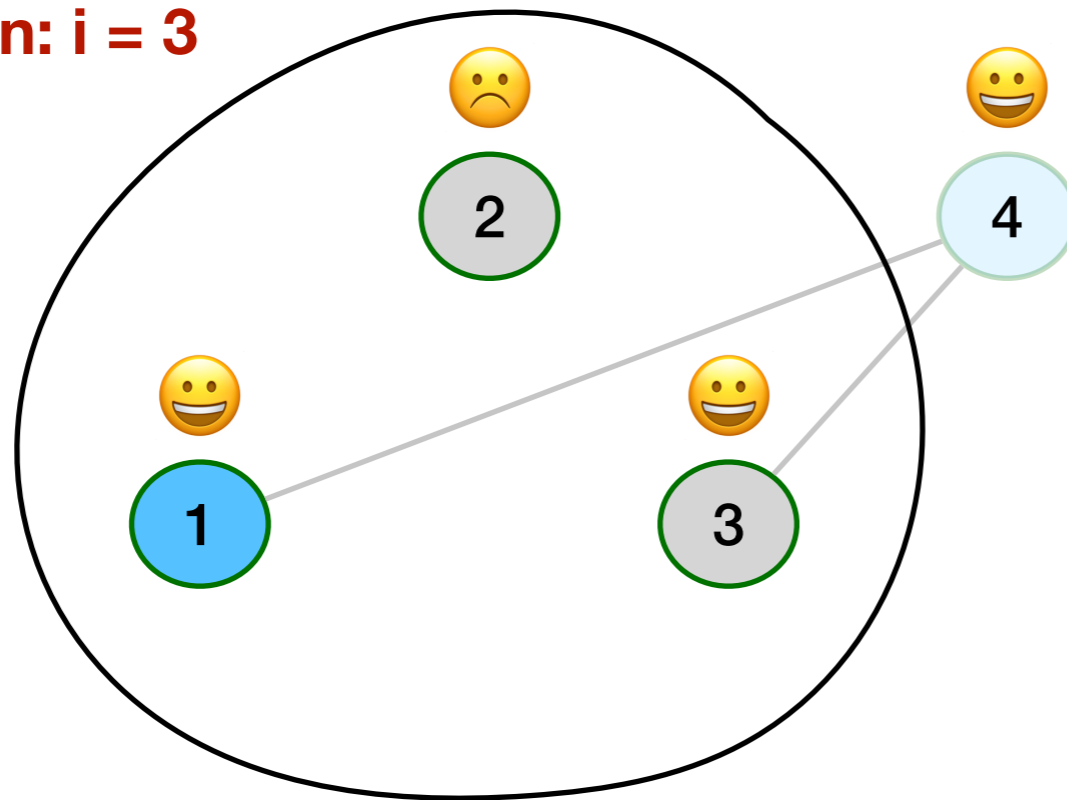


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \textit{Blue}(y))$$

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

Iteration: i = 3

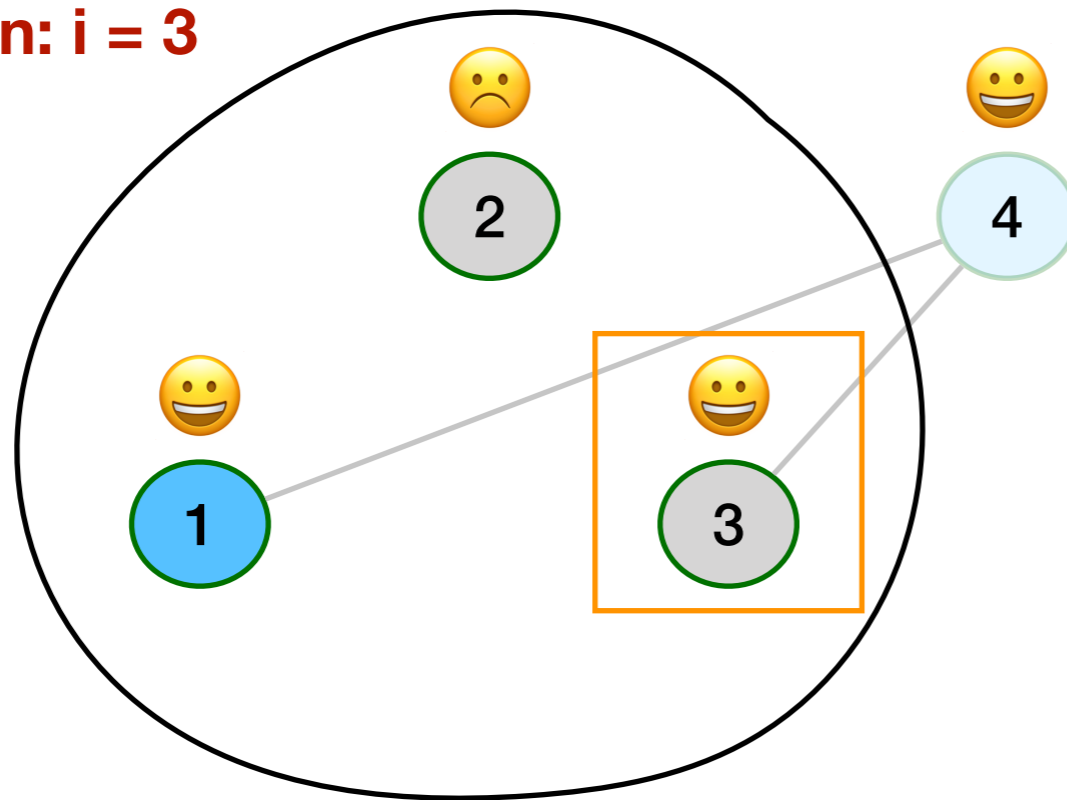


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \text{Blue}(y))$$

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

Iteration: i = 3

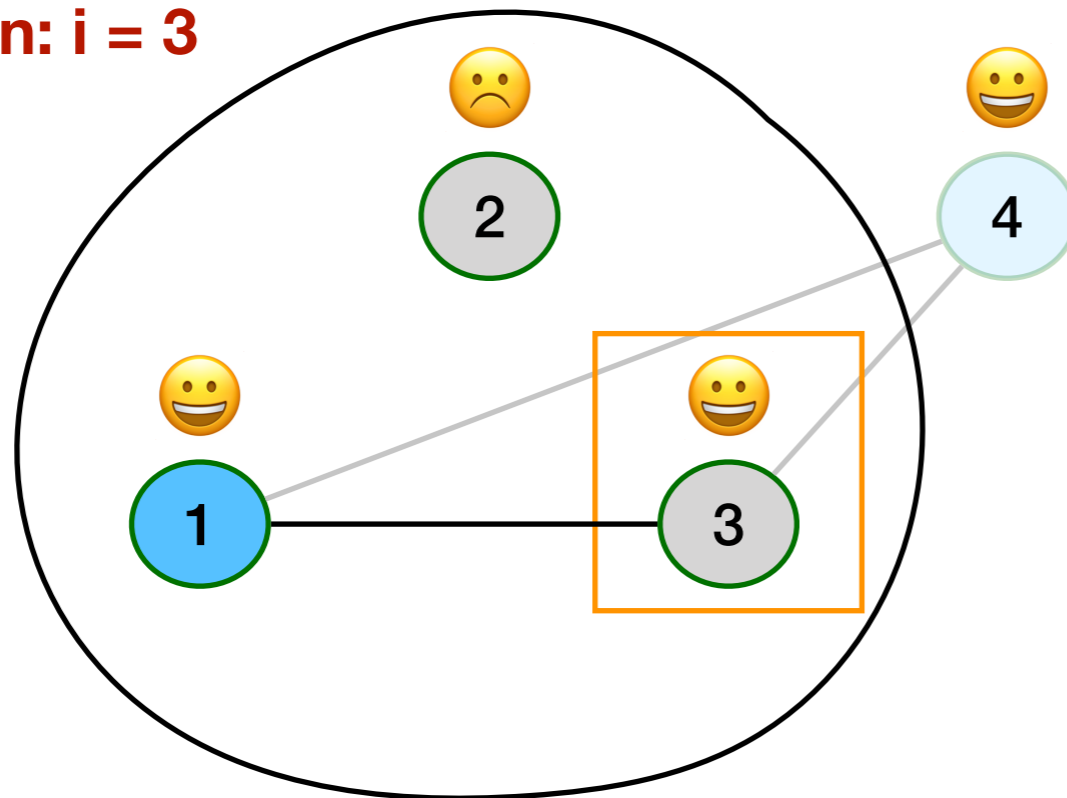


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \text{Blue}(y))$$

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

Iteration: $i = 3$

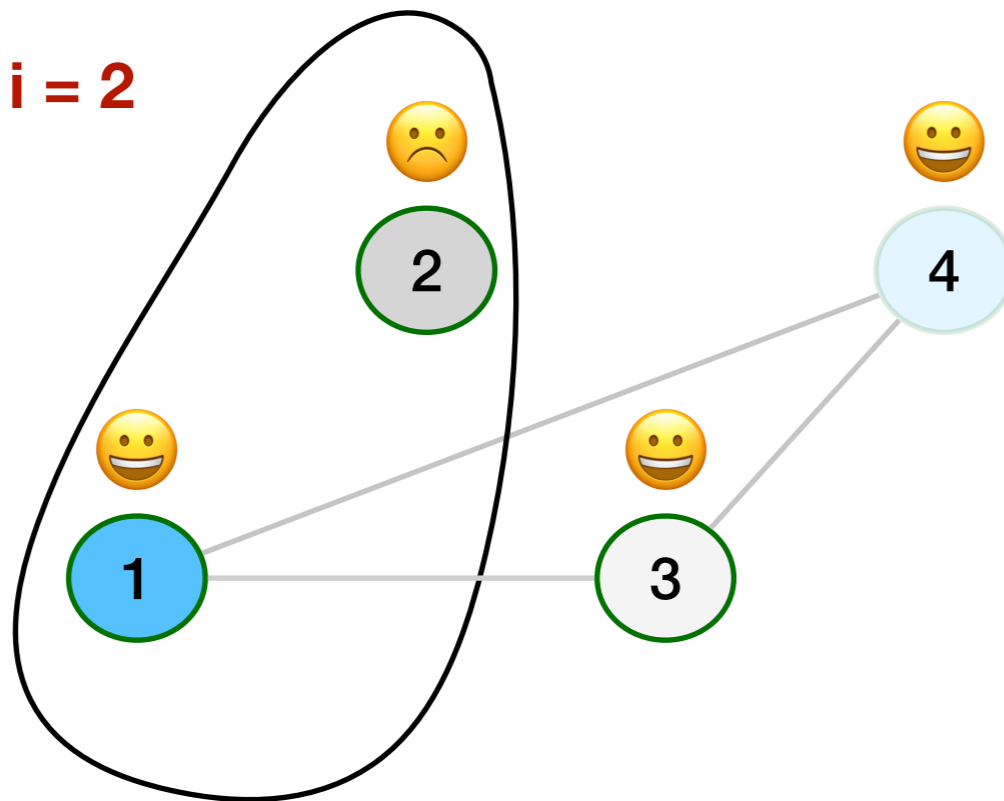


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \textit{Blue}(y))$$

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

Iteration: i = 2

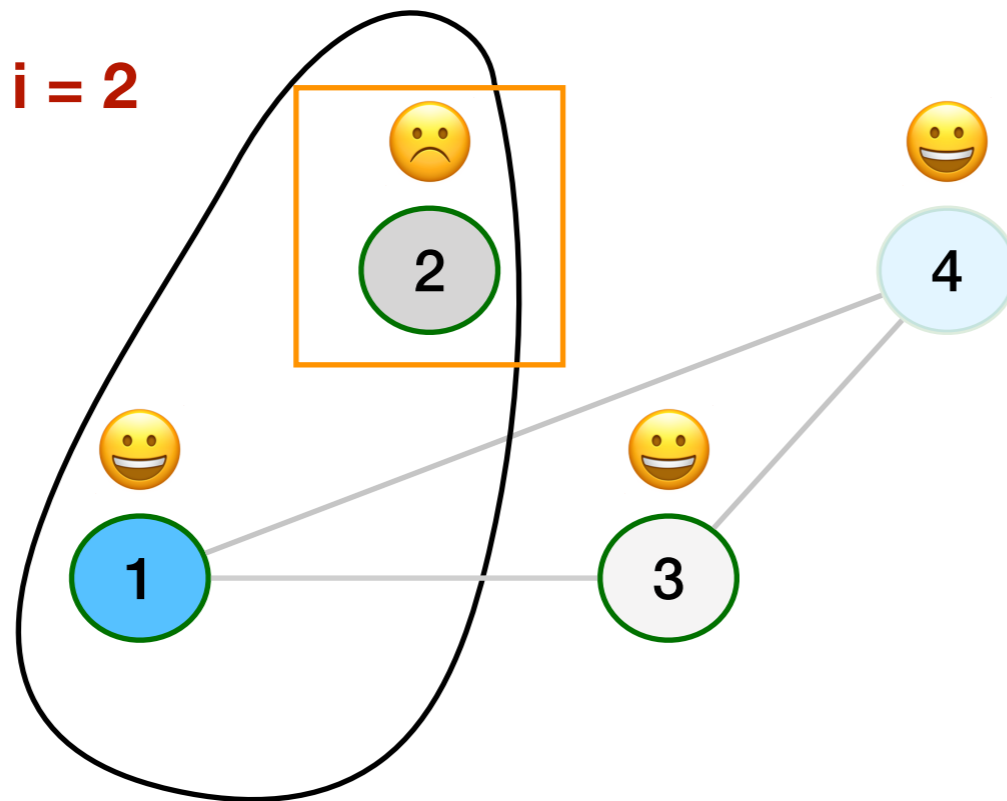


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \text{Blue}(y))$$

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

Iteration: i = 2

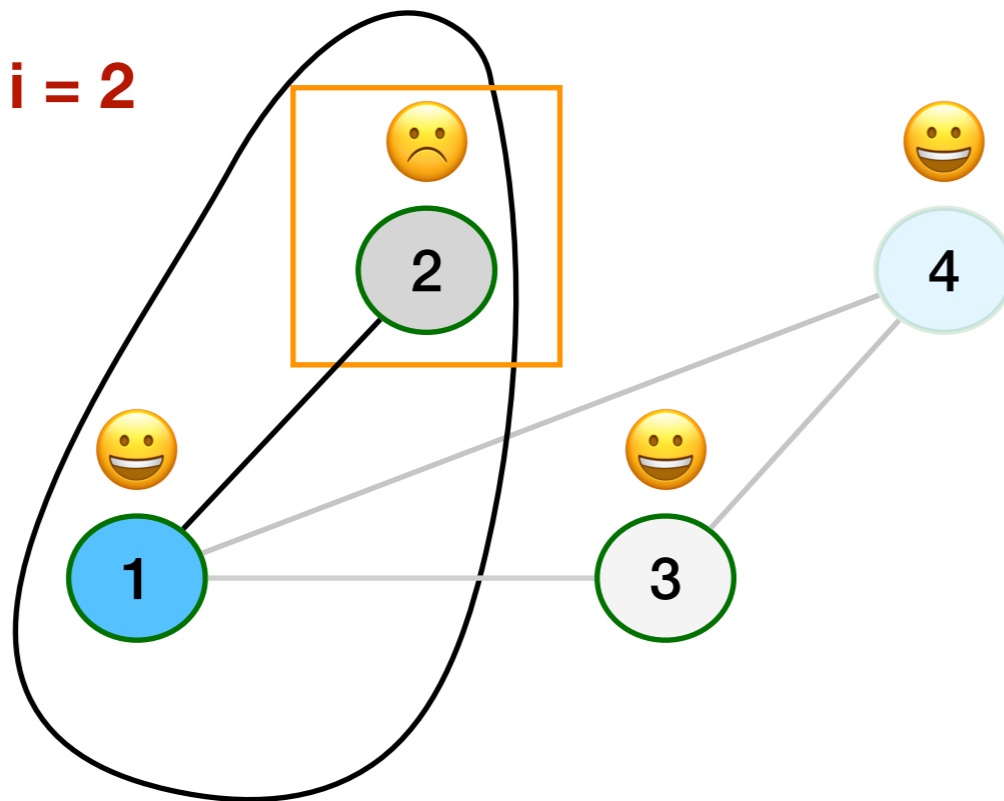


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \text{Blue}(y))$$

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

Iteration: i = 2

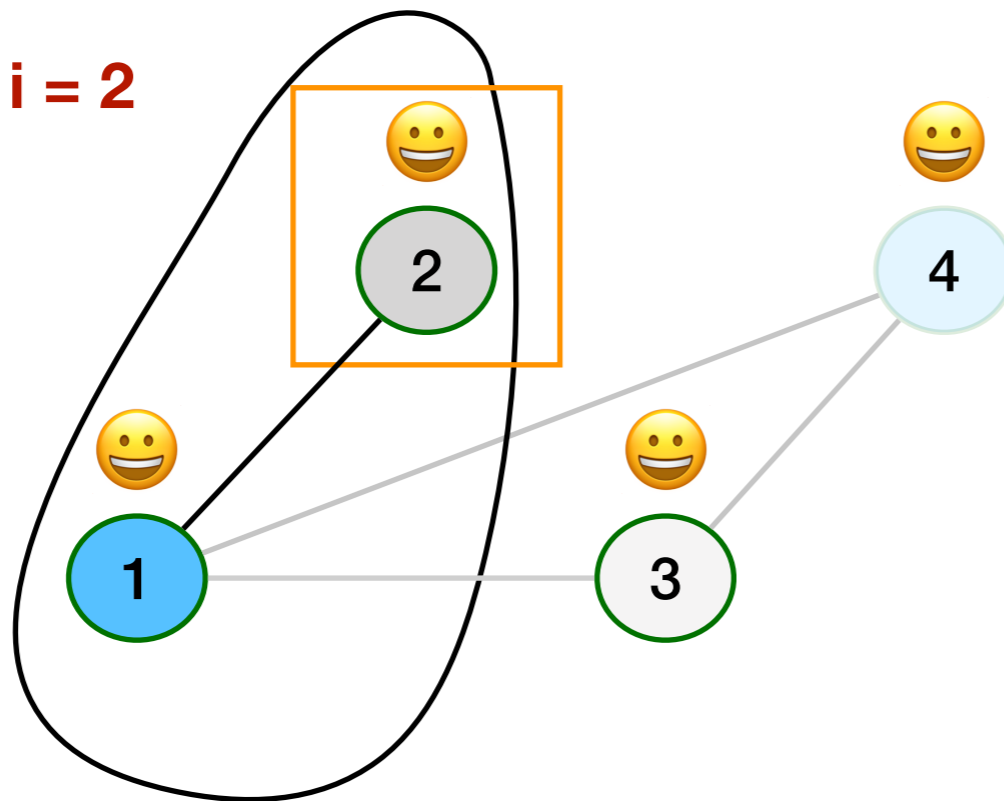


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \text{Blue}(y))$$

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

Iteration: i = 2

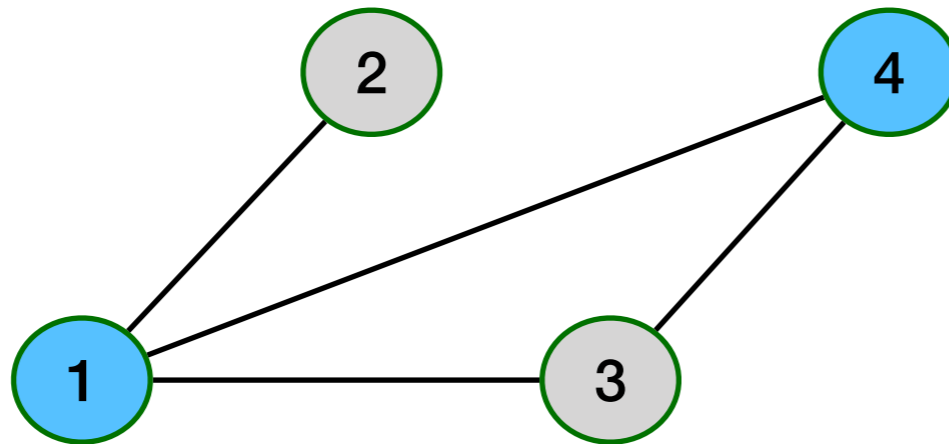


How It Works

$$\Gamma = \forall x \neg E(x, x) \wedge \forall x \forall y E(x, y) \Rightarrow E(y, x) \wedge \forall x \exists y (E(x, y) \wedge \textit{Blue}(y))$$

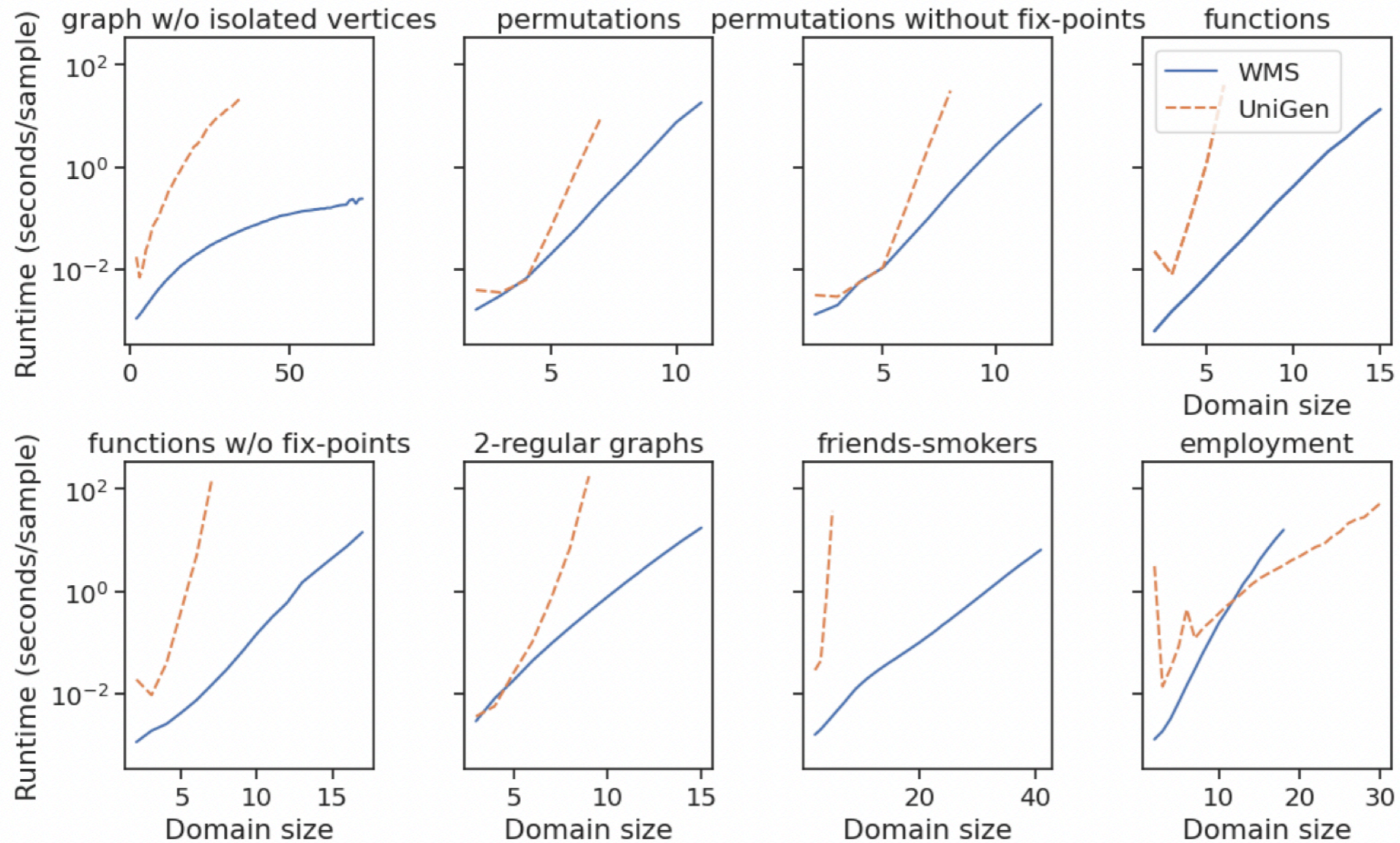
$$\Delta = \{1, 2, 3, 4\}$$

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 \mathbf{C}^2 sentences in time polynomial in the domain size.

Acknowledgements

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 OP VVV project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".