Towards Practical First-Order Model Counting

Ananth K. Kidambi[1] Guramrit Singh[1] **Paulius Dilkas**[2,3]
Kuldeep S. Meel[4,2]

[1]IIT Bombay, India

[2]University of Toronto, Canada

[3]Vector Institute, Canada

[4]Georgia Tech, USA

SAT 2025

Motivation

Example Setting

- Let Δ be a set of cardinality n
- Suppose we want to count all $P \subseteq \Delta^2$ (as a function of n) that are:
  - functions,
  - bijections,
  - partial orders,
  - symmetric,
  - transitive,
  - etc.

Motivation

Example Setting

- Let Δ be a set of cardinality n
- Suppose we want to count all $P \subseteq \Delta^2$ (as a function of n) that are:
  - functions,
  - bijections,
  - partial orders,
  - symmetric,
  - transitive,
  - etc.

- Propositional model counting (#SAT) is #P-complete
- But many of these counting problems have efficient solutions
- And we can find them using first-order model counting
  - i.e., reasoning about sets, subsets, and arbitrary elements without grounding them

First-Order Model Counting

The Problem with CRANE

A Solution Produced for the Bijection-Counting Problem

$$f(m, n) = \sum_{l=0}^{n} \binom{n}{l} (-1)^{n-l} g(l, m),$$
$$g(l, m) = g(l-1, m) + m g(l-1, m-1)$$

The Problem with Crane

A Solution Produced for the Bijection-Counting Problem

$$f(m, n) = \sum_{l=0}^{n} \binom{n}{l} (-1)^{n-l} g(l, m),$$
$$g(l, m) = g(l-1, m) + m g(l-1, m-1)$$

Issues

Completeness: what are the base cases of g?

Usability: how do I compute, e.g., $f(7, 7)$?

Knowledge Compilation Workflow (1/2)

1. Use CRANE to compile ϕ into a set of equations $\mathcal{E}$

Knowledge Compilation Workflow (1/2)

1. Use CRANE to compile ϕ into a set of equations $\mathcal{E}$
2. Simplify them, e.g.,

$$g(l, m) = \sum_{k=0}^{m} [0 \le k \le 1] \binom{m}{k} g(l-1, m-k)$$

becomes

$$g(l, m) = g(l-1, m) + mg(l-1, m-1)$$

Knowledge Compilation Workflow (1/2)

1. Use CRANE to compile ϕ into a set of equations $\mathcal{E}$
2. Simplify them, e.g.,

$$g(l, m) = \sum_{k=0}^{m} [0 \leq k \leq 1] \binom{m}{k} g(l-1, m-k)$$

becomes

$$g(l, m) = g(l-1, m) + mg(l-1, m-1)$$

3. Identify a sufficient set of base cases
   - e.g., $\{ g(0, m), g(l, 0) \}$

Knowledge Compilation Workflow (2/2)

4. For each base case:

$g(0, m)$ $g(l, 0)$

Knowledge Compilation Workflow (2/2)

4. For each base case:

4.1. Construct the corr. sentence

$$g(0, m) \longrightarrow \forall x \in \Gamma.\ S(x) \lor \neg S(x)$$

$$g(l, 0) \longrightarrow \top$$

Knowledge Compilation Workflow (2/2)

4. For each base case:

4.1. Construct the corr. sentence

4.2. Recurse

$$
\begin{array}{ccc}
g(0, m) & & g(l, 0) \\
\downarrow & & \downarrow \\
\forall x \in \Gamma.\ S(x) \lor \neg S(x) & & \top \\
\downarrow & & \downarrow \\
g(0, m) = ??? & & g(l, 0) = ???
\end{array}
$$

Knowledge Compilation Workflow (2/2)

4. For each base case:

4.1. Construct the corr. sentence

4.2. Recurse

$$\begin{array}{ccc}
g(0, m) & & g(l, 0) \\
\downarrow & & \downarrow \\
\forall x \in \Gamma.\ S(x) \lor \neg S(x) & & \top \\
\downarrow & & \downarrow \\
g(0, m) = 0^m & & g(l, 0) = 1
\end{array}$$

Knowledge Compilation Workflow (2/2)

4. For each base case:

4.1. Construct the corr. sentence

4.2. Recurse

4.3. Add to $\mathcal{E}$

$$\begin{tikzcd}
g(0, m) \arrow[d] & & g(l, 0) \arrow[d] \\
\forall x \in \Gamma.\ S(x) \lor \neg S(x) \arrow[d] & & \top \arrow[d] \\
g(0, m) = 0^m \arrow[dr] & & g(l, 0) = 1 \arrow[dl] \\
& \mathcal{E} &
\end{tikzcd}$$

Knowledge Compilation Workflow (2/2)

4. For each base case:

4.1. Construct the corr. sentence

4.2. Recurse

4.3. Add to $\mathcal{E}$

$$\begin{tikzcd}
g(0, m) \arrow[d] & & g(l, 0) \arrow[d] \\
\forall x \in \Gamma.\ S(x) \lor \neg S(x) \arrow[d] & & \top \arrow[d] \\
g(0, m) = 0^m \arrow[dr] & & g(l, 0) = 1 \arrow[dl] \\
& \mathcal{E} \arrow[d, "5.\ \text{Compile to C++}"] & \\
& \boxed{\text{C++ code}} &
\end{tikzcd}$$

Knowledge Compilation Workflow (2/2)

4. For each base case:

$$
\begin{tikzcd}
g(0, m) \arrow[d] & & g(l, 0) \arrow[d] \\
\forall x \in \Gamma.\ S(x) \lor \neg S(x) \arrow[d] & & \top \arrow[d] \\
g(0, m) = 0^m \arrow[dr] & & g(l, 0) = 1 \arrow[dl] \\
& \mathcal{E} \arrow[d, "\text{5. Compile to C++}"] & \\
\text{Domain sizes} \arrow[r] & \boxed{\text{C++ code}} \arrow[r] & \text{Model count}
\end{tikzcd}
$$

4.1. Construct the corr. sentence

4.2. Recurse

4.3. Add to $\mathcal{E}$

Benchmarks

- Friends & Smokers

$$(\forall x, y \in \Delta.\ S(x) \wedge F(x, y) \rightarrow S(y)) \wedge (\forall x \in \Delta.\ S(x) \rightarrow C(x))$$

Benchmarks

- Friends & Smokers

$$(\forall x, y \in \Delta.\ S(x) \wedge F(x,y) \rightarrow S(y)) \wedge (\forall x \in \Delta.\ S(x) \rightarrow C(x))$$

- Functions

$$(\forall x \in \Gamma.\ \exists y \in \Delta.\ P(x,y)) \wedge$$
$$(\forall x \in \Gamma.\ \forall y, z \in \Delta.\ P(x,y) \wedge P(x,z) \rightarrow y = z)$$

Benchmarks

- Friends & Smokers

$$(\forall x, y \in \Delta.\ S(x) \wedge F(x, y) \rightarrow S(y)) \wedge (\forall x \in \Delta.\ S(x) \rightarrow C(x))$$

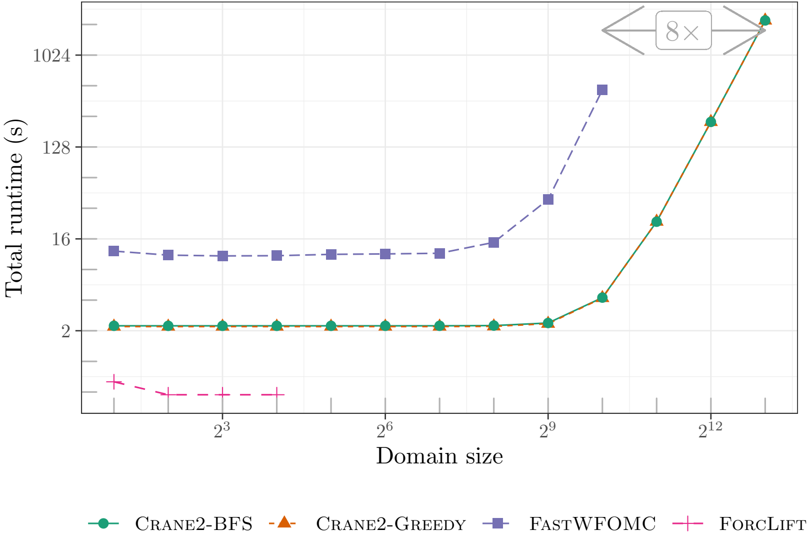
- Functions

$$\begin{gathered}(\forall x \in \Gamma.\ \exists y \in \Delta.\ P(x, y)) \wedge \\ (\forall x \in \Gamma.\ \forall y, z \in \Delta.\ P(x, y) \wedge P(x, z) \rightarrow y = z)\end{gathered}$$

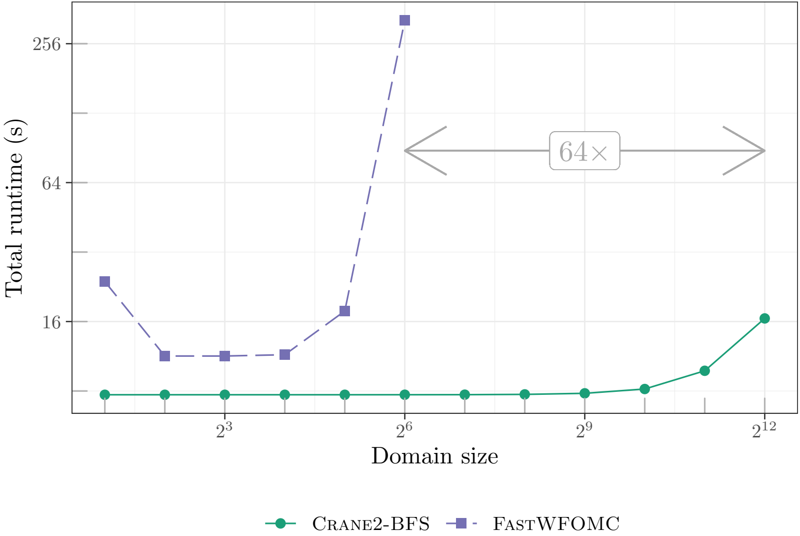
- Bijections

$$\begin{gathered}(\forall x \in \Gamma.\ \exists y \in \Delta.\ P(x, y)) \wedge \\ (\forall y \in \Delta.\ \exists x \in \Gamma.\ P(x, y)) \wedge \\ (\forall x \in \Gamma.\ \forall y, z \in \Delta.\ P(x, y) \wedge P(x, z) \rightarrow y = z) \wedge \\ (\forall x, z \in \Gamma.\ \forall y \in \Delta.\ P(x, y) \wedge P(z, y) \rightarrow x = z)\end{gathered}$$

Friends & Smokers

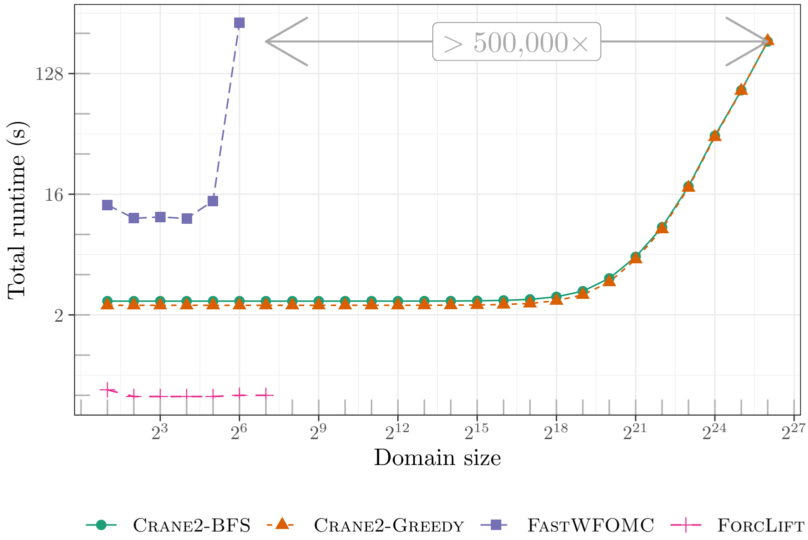


Bijections



Total runtime (s)
256
64
16
64×
2^3
2^6
2^9
2^12
Domain size
Crane2-BFS
FastWFOMC

Functions



> 500,000×
128
16
2
Total runtime (s)
2^3
2^6
2^9
2^{12}
2^{15}
2^{18}
2^{21}
2^{24}
2^{27}
Domain size
Crane2-BFS
Crane2-Greedy
FastWFOMC
ForcLift

Summary

TODO: and future work