# Probabilistic Inference with Algebraic Constraints:

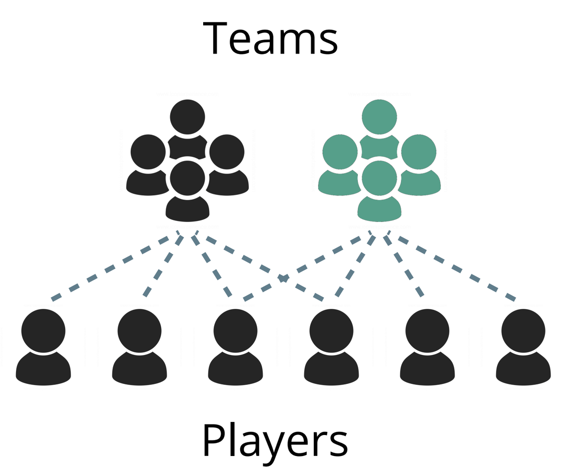
## *Theoretical Limits and Practical Approximations*


Zhe Zeng*[1] Paolo Morettin*[2] Fanqi Yan*[3]
Antonio Vergari[1] Guy Van den Broeck[1]

[1]University of California, Los Angeles
[2]KU Leuven [3]University of Texas at Austin


*December – NeurIPS 2020 Spotlight*

***TL;DR:*** *Hybrid probabilistic inference with constraints is HARD. Efficient approximate inference is necessary!*




**Constraints**

$$\bigwedge_i 0 \le X_{P_i} \le 10$$
$$\bigwedge_j \bigwedge_{i \in T_j} | X_{T_j} - X_{P_i} | < 1$$
$$\bigwedge_j (B_{S_j} \Rightarrow X_{T_j} > 2)$$

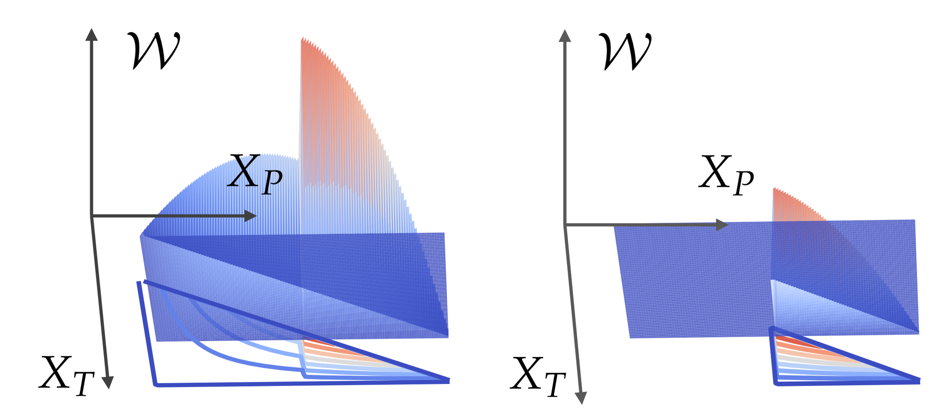
**+**

**weights**

$$\begin{cases} w(X_{P_i}), & \text{if } 0 \le X_{P_i} \le 10 \\ w(X_{T_j}, X_{P_i}), & \text{if } | X_{T_j} - X_{P_i} | < 1 \\ w(B_{S_j}, X_{T_j}), & \text{if } B_{S_j} \Rightarrow X_{T_j} > 2 \end{cases}$$

**=**

**Weighted Model Integration (WMI)**




$(I)\ B_S = \top$ $(II)\ B_S = \bot$

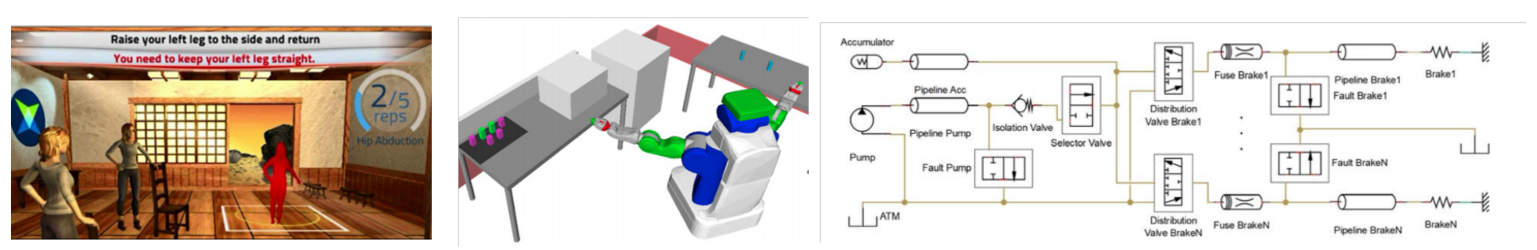
**Tractability**

**Approximation**

## How to model *Discrete + Continuous + Constraints?*

**Satisfiability Modulo Theories (SMT)**

SMT of linear real arithmetic as *representation language*



*To further make an SMT formula probabilistic ...*

**SMT** + **weights** = **Weighted Model Integration (WMI)**

**Definition (WMI).** Given an SMT formula $\Delta$ over continuous variables $X$ and discrete variables $B$, and weight function $\mathcal{W}$, the weighted model integration (WMI) is

$$\mathsf{WMI}(\Delta, \mathcal{W}; \mathbf{X}, \mathbf{B}) \triangleq \sum_{\boldsymbol{b} \in \mathbb{B}^{|\mathbf{B}|}} \int_{(\mathbf{x}, \boldsymbol{b}) \models \Delta} w(\mathbf{x}, \boldsymbol{b})\, d\mathbf{x}.$$

**Example.** Given a query:

*What is the probability of team T1 outperforming team T2, if T1 is a squad but T2 is not?*

We can answer it by WMI:

$$\Phi_S : (B_{S_1} = 1 \wedge B_{S_2} = 0) \implies T_1 \text{ is a squad, } T_2 \text{ is not}$$
$$\Phi_T : (X_{T_1} > X_{T_2}) \implies T_1 \text{ outperforms } T_2$$

$$\Pr_\Delta(\Phi_T \mid \Phi_S) = \frac{\mathsf{WMI}(\Delta \wedge \Phi_T \wedge \Phi_S, \mathcal{W})}{\mathsf{WMI}(\Delta \wedge \Phi_S, \mathcal{W})} = \frac{4,206}{7,225} \approx 58.22\%$$

## How hard is *inference?*

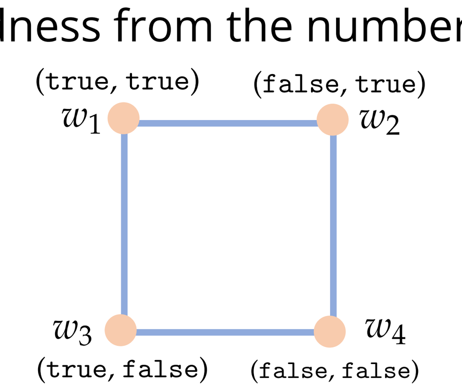
*Discrete cases vs. Hybrid cases*

We explore the *tractability* of WMI problems by looking at the ***primal graphs*** for SMT formulas.

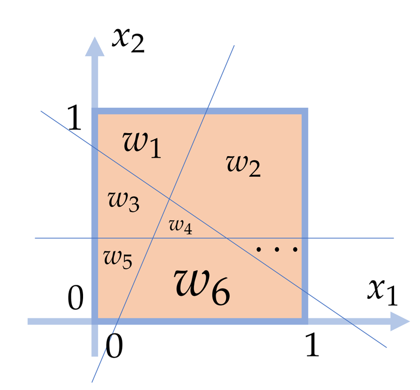
**Example (primal graph).**

SMT formula $(X_1 < X_2 + 1.1 \vee X_1 > X_2 - 0.1) \wedge (X_2 - 0.1 < X_3)$ Primal graph $X_1$ — $X_2$ — $X_3$

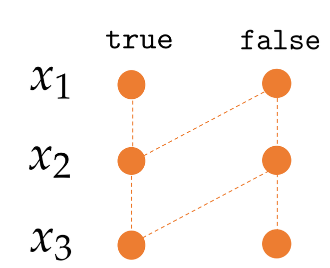
- Hardness from the number of constraints:



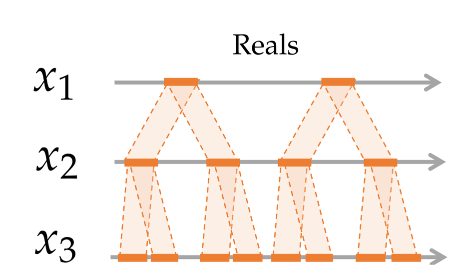

vs.




- Hardness from integration pieces not bounded by graph diameter:




vs.

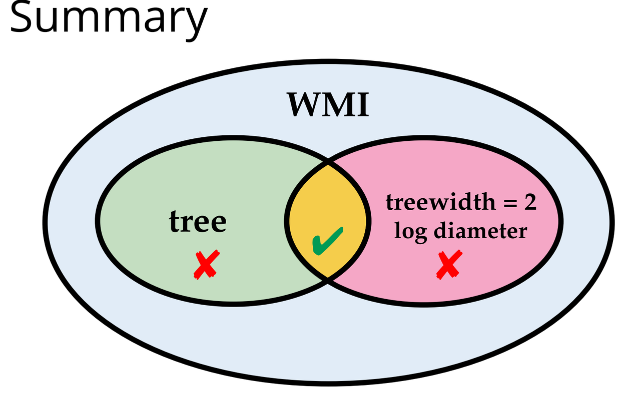



=> *WMI Inference on tree-shaped primal graphs with unbounded-diameter is #P-hard!*

- Hardness from loopy structures:

=> *WMI inference on primal graphs with bounded-diameter but treewidth two is #P-hard!*

- Summary

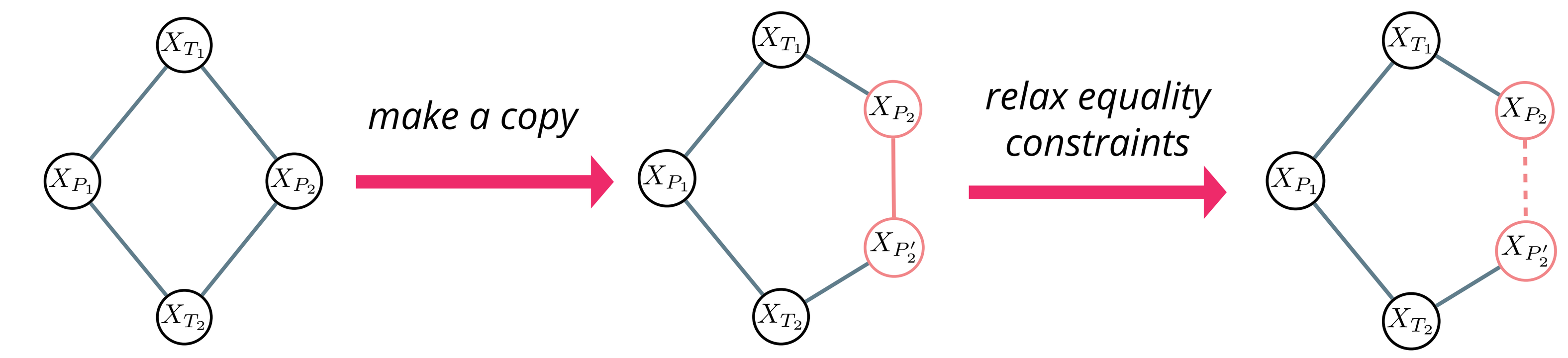



- *#P-hard in general*
- *Tree problem class: intractable*
- *Logarithmic diameter and treewidth two: intractable*
- *Intersection: tractable* [1]

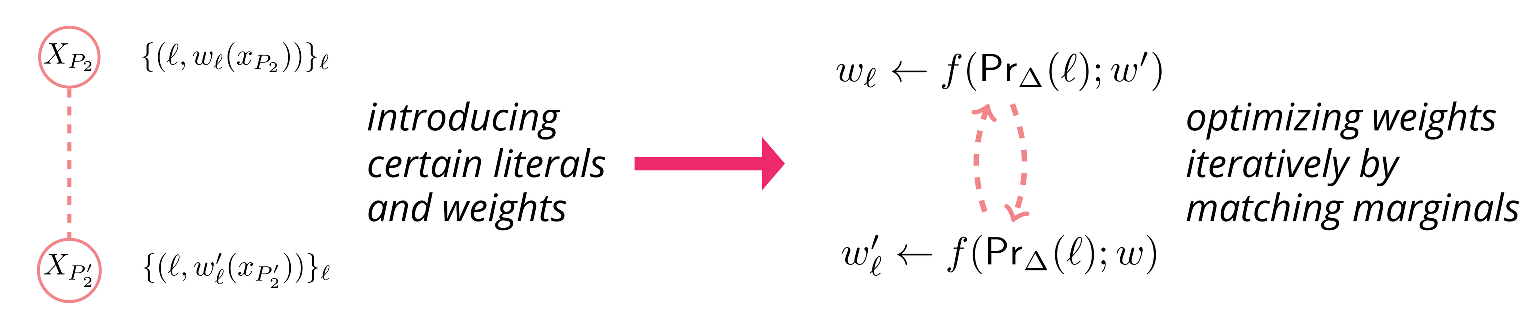
## How to perform *approximations?*

***ReCoIn:*** ***Re****lax,* ***Co****mpensate, and* ***In****tegrate*

- Given a WMI problem with loopy primal graph, ReCoIn breaks loops by:




- Further, ReCoIn compensate for the removed dependencies by:

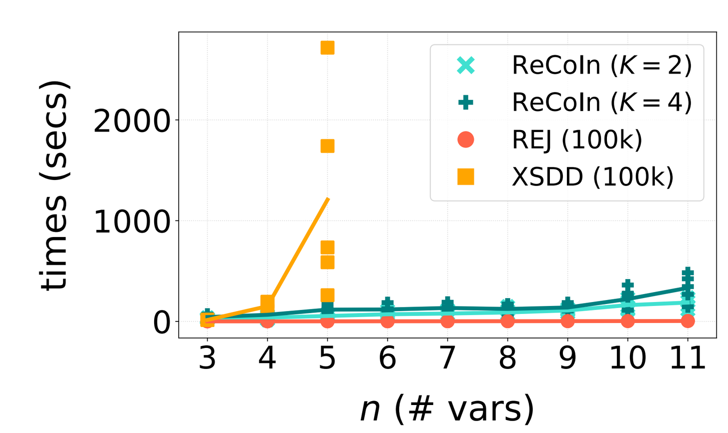


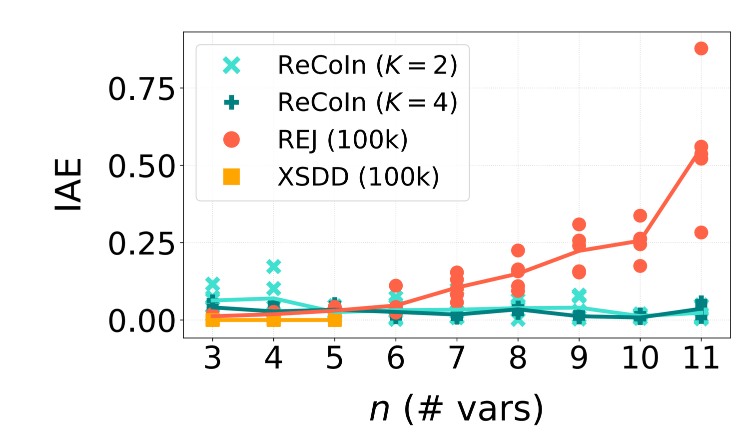

**Algorithm 1** ReCoIn $(\Delta, \mathcal{W}, K)$

**Input:** a WMI model $(\Delta, \mathcal{W})$, $K$ number of compensating literals
**Output:** $(\Delta^{\mathsf{rel}}, \mathcal{W}^{\mathsf{rel}})$: a relaxed and compensated WMI model

```
E_d ← initStrategy(Δ, W)                                   ▹ Select edges to remove
Δ^aug, W^aug, L ← augmentModel(Δ, W, E_d)
(Δ^rel, W^rel), (Δ^rem, W^rem) ← relaxModel(Δ^aug, W^aug, L)
Δ^rel, W^rel ← addingCompensations(Δ^rel, W^rel, L, K)
while not converged do
    for X_i ∈ copiedNodes(Δ^rel) do
        for k = 1, ..., K do
            r^k ← WMI(Δ^rem, W^rem) / WMI(Δ^rem ∧ ⋀_{c=0}^{C_i} ℓ^c_{k,i}, W^rem) − 1
            for c = 0, 1, ..., C_i do
                θ^{c,(t+1)}_{k,i} ← log(r^k α_{k,σ(c)}) − log(1 − α_{k,σ(c)}) − Σ_{c'≠c} θ^{c,(t)}_{k,i}
Return (Δ^rel, W^rel)
```

## *Experiments*




**Code Available at:** github.com/UCLA-StarAI/recoin

**Reference:** [1] Zeng, Z., Morettin, P., Yan, F., Vergari, A., Broeck, G.V.d.: Scaling up hybridprobabilistic inference with logical and arithmetic constraints via message passing.In: International Conference of Machine Learning (2020)