Intelligent Agents:
Web-mining Agents

Probabilistic Graphical Models

Lifted Inference

Tanya Braun

UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR INFORMATIONSSYSTEME

Probabilistic Graphical Models (PGMs)

1. Recap: **Propositional** modelling
   - Factor model, Bayesian network, Markov network
   - Semantics, inference tasks + algorithms + complexity
2. **Probabilistic relational models** (PRMs)
   - Parameterised models, Markov logic networks
   - Semantics, inference tasks
3. **Lifted inference**
   - LVE, LJT, FOKC
   - Theoretical analysis
4. **Lifted learning**
   - Recap: propositional learning
   - From ground to lifted models
   - Direct lifted learning
5. **Approximate Inference: Sampling**
   - Importance sampling
   - MCMC methods
6. **Sequential models & inference**
   - Dynamic PRMs
   - Semantics, inference tasks + algorithms + complexity, learning
7. **Decision making**
   - (Dynamic) Decision PRMs
   - Semantics, inference tasks + algorithms, learning
8. **Continuous Models**
   - Probabilistic soft logic: modelling, semantics, inference tasks + algorithms

Local Symmetries and Structure

- Consider potential function as given by the table on the right
  $$\phi(Travel(X), Epid, Sick(X))$$
- Only two weighted formulas (w, ψ) necessary
  - $(\ln 2, \neg travel(X) \vee \neg epid \vee \neg sick(X))$
  - $(\ln 7, travel(X) \wedge epid \wedge sick(X))$
  - If potential of 1 instead of 2, would reduce to
    - $(\ln 7, travel(X) \wedge epid \wedge sick(X))$
    - assignments that do not make the formula true automatically get weight of $0 = \ln 1$
- If external knowledge existing, provide FOL formulas directly
  - E.g., $(\ln 2, epid \wedge sick(X) \Rightarrow \neg travel(X))$

Use for efficient inference

$Travel(X)$	$Epid$	$Sick(X)$	ϕ
$false$	$false$	$false$	2
$false$	$false$	$true$	2
$false$	$true$	$false$	2
$false$	$true$	$true$	2
$true$	$false$	$false$	2
$true$	$false$	$true$	2
$true$	$true$	$false$	2
$true$	$true$	$true$	7

MLNs: Semantics

- MLN $\Psi = \{(w_i, \psi_i)\}_{i=1}^{n}$, with $w_i \in \mathbb{R}$, induces a probability distribution over possible worlds
$$\omega \in \{true, false\}^N$$
  - N = the number of ground atoms in the grounded Ψ

$$P(\omega) = \frac{1}{Z} \prod_{i=1}^{n} \exp(w_i)^{n_i(\omega)} = \frac{1}{Z} \exp\left(\sum_{i=1}^{n} w_i n_i(\omega)\right)$$

  - $n_i(\omega)$ = number of true instances of ψ_i in ω

10	Presents(X,P,C) => Attends(X,C)

3.75	Publishes(X,C) ∧ FarAway(C) => Attends(X,C)

Outline: 3. Lifted Inference

A. Lifted variable elimination (LVE)
- Operators
- Algorithm
- Complexity (including first-order dtrees), completeness, tractability
- Variants

B. Lifted junction tree algorithm (LJT)
- First-order junction trees (FO jtrees)
- Algorithm
- Complexity, completeness
- Variants

C. First-order knowledge compilation (FOKC)
- Normal form, circuits
- Algorithm
- Complexity, completeness

D. Beyond Standard Query Answering
- Adaptive inference
- Changing and unknown domains
- Assignment queries

Weighted Model Counting

- Solve query answering problem by solving a weighted model counting problem
  - Weighted model count (WMC) given a sentence φ in propositional logic and a weight function $weight : L \to \mathbb{R}_{\geq 0}$ associating a non-negative weight to each literal in φ (set L) defined by

$$WMC(\varphi, weight) = \sum_{\omega \in \Omega_\varphi} \prod_{l \in \omega} weight(l)$$

    - where Ω_φ refers to the set of worlds of φ
  - Probability of a world ω of a sentence φ with weight function

$$P(\omega) = \frac{\prod_{l \in \omega} weight(l)}{WMC(\varphi, weight)} = \frac{WMC(\varphi \land \omega, weight)}{WMC(\varphi, weight)}$$

  - A query for literal q given evidence e is solved by computing

$$P(q|e) = \frac{WMC(\varphi \land q \land e, weight)}{WMC(\varphi \land e, weight)}$$

Vgl. $P(Q|E) = \frac{P(Q,E)}{P(E)}$

T. Sang, P. Beame, and H. Kautz: Solving Bayesian networks by weighted model counting. In: *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 2005.

Weighted Model Counting: Example

- Sentence
  - $sun \wedge rain \Rightarrow rainbow$
- Weight function:
  - $weight(sun) = 1$
  - $weight(\neg sun) = 5$
  - $weight(rain) = 2$
  - $weight(\neg rain) = 7$
  - $weight(rainbow) = 0.1$
  - $weight(\neg rainbow) = 10$

$$WMC(\varphi, weight) = \sum_{\omega \in \Omega_\varphi} \prod_{l \in \omega} weight(l)$$

Each line a world $\omega \in \Omega_\varphi$

$rain$	sun	$rainbow$	Weight	
0	0	0	$7 \cdot 5 \cdot 10$	350
0	0	1	$7 \cdot 5 \cdot 0.1$	3.5
0	1	0	$7 \cdot 1 \cdot 10$	70
0	1	1	$7 \cdot 1 \cdot 0.1$	0.7
1	0	0	$2 \cdot 5 \cdot 10$	100
1	0	1	$2 \cdot 5 \cdot 0.1$	1
~~1~~	~~1~~	~~0~~	~~$2 \cdot 1 \cdot 10$~~	~~20~~ 0
1	1	1	$2 \cdot 1 \cdot 0.1$	0.2
			+	525.4

Weighted Model Counting: Example

- Sentence
  - $sun \land rain \Rightarrow rainbow$
- Weight function:
  - $weight(sun) = 1$
  - $weight(\neg sun) = 5$
  - $weight(rain) = 2$
  - $weight(\neg rain) = 7$
  - $weight(rainbow) = 0.1$
  - $weight(\neg rainbow) = 10$
- Probability of worlds:
  - $P(sun, rain, rainbow) = \frac{0.2}{525.4} = 0.00038$

$$P(\omega) = \frac{\prod_{l \in \omega} weight(l)}{WMC(\varphi, weight)} = \frac{WMC(\varphi \land \omega, weight)}{WMC(\varphi, weight)}$$

$(sun \land rain \Rightarrow rainbow) \land sun \land rain \land rainbow$

$rain$	sun	$rainbow$	Weight	
~~0~~	~~0~~	~~0~~	~~$7 \cdot 5 \cdot 10$~~	~~350~~
~~0~~	~~0~~	~~1~~	~~$7 \cdot 5 \cdot 0.1$~~	~~3.5~~
~~0~~	~~1~~	~~0~~	~~$7 \cdot 1 \cdot 10$~~	~~70~~
~~0~~	~~1~~	~~1~~	~~$7 \cdot 1 \cdot 0.1$~~	~~0.7~~
~~1~~	~~0~~	~~0~~	~~$2 \cdot 5 \cdot 10$~~	~~100~~
~~1~~	~~0~~	~~1~~	~~$2 \cdot 5 \cdot 0.1$~~	~~1~~
~~1~~	~~1~~	~~0~~	~~$2 \cdot 1 \cdot 10$~~	~~20~~ 0
1	1	1	$2 \cdot 1 \cdot 0.1$	0.2
			+	**525.4**

$\omega = (sun, rain, rainbow) \in \Omega_\varphi$

Weighted Model Counting: Example

- Sentence
  - $sun \land rain \Rightarrow rainbow$
- Weight function:
  - $weight(sun) = 1$
  - $weight(\neg sun) = 5$
  - $weight(rain) = 2$
  - $weight(\neg rain) = 7$
  - $weight(rainbow) = 0.1$
  - $weight(\neg rainbow) = 10$
- Probability of worlds:
  - $P(rain)$
    $= \frac{100 + 1 + 0.2}{525.4} = 0.1926$

$$P(q) = \frac{WMC(\varphi \land q, weight)}{WMC(\varphi, weight)}$$

$(sun \land rain \Rightarrow rainbow) \land rain$

$rain$	sun	$rainbow$	Weight	
~~0~~	~~0~~	~~0~~	~~$7 \cdot 5 \cdot 10$~~	~~350~~
~~0~~	~~0~~	~~1~~	~~$7 \cdot 5 \cdot 0.1$~~	~~3.5~~
~~0~~	~~1~~	~~0~~	~~$7 \cdot 1 \cdot 10$~~	~~70~~
~~0~~	~~1~~	~~1~~	~~$7 \cdot 1 \cdot 0.1$~~	~~0.7~~
1	0	0	$2 \cdot 5 \cdot 10$	100
1	0	1	$2 \cdot 5 \cdot 0.1$	1
~~1~~	~~1~~	~~0~~	~~$2 \cdot 1 \cdot 10$~~	~~20~~ 0
1	1	1	$2 \cdot 1 \cdot 0.1$	0.2
			+	525.4

All $\omega \in \Omega_\varphi$ where $rain$ holds

WMC and Inference

- Solving a WMC problem for a sentence φ as introduced on previous slides is exponential in number of worlds with probability > 0 (models)
- To be more efficient, build a helper structure
  - Bring sentence into negation normal form (NNF)
    - NNF: Formulas contain only negations directly in front of variables, conjunctions, and disjunctions
  - E.g.,
    - $sun \land rain \Rightarrow rainbow$ (Apply $A \Rightarrow B \equiv \neg A \lor B$)
      $\equiv \neg(sun \land rain) \lor rainbow$ (Apply De Morgan's law)
      $\equiv \neg sun \lor \neg rain \lor rainbow$ (NNF)

Circuits

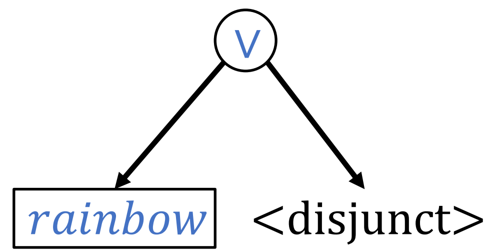
- Represent the NNF sentence as a directed, acyclic graph called circuit with leaves labelled with literals (l or $\neg l$) or $true, false$ with inner nodes being
  - *Deterministic* disjunctions
    - Only one disjunct (child node) can be true at the same time
      - I.e., their conjunction is unsatisfiable
  - *Decomposable* conjunctions
    - Each pair of conjuncts (child nodes) must be independent
      - I.e., they cannot share any variables
- Circuit is then in d-DNNF
  - deterministic Decomposable NNF
  - See later why important

Mark Chavira and Adnan Darwiche: On probabilistic inference by weighted model counting. In: *Artificial Intelligence*, 2008.

Circuits: Example

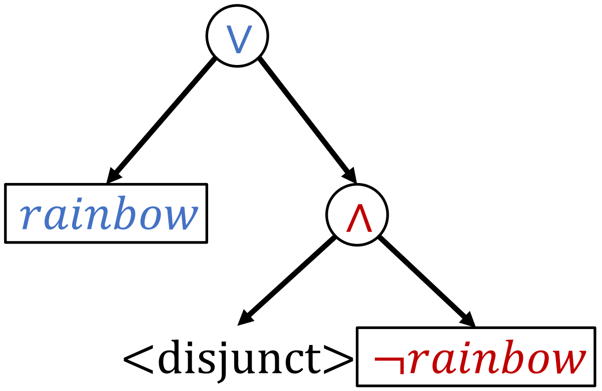
- *Deterministic* disjunctions
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable
- *Decomposable* conjunctions
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables

- E.g., $\neg sun \lor \neg rain \lor rainbow$
  - <disjunct> $\lor\ rainbow$
    - Determinism:
      <disjunct> can only be true if $rainbow$ is not
      - Add $\neg rainbow$ to disjunct:
        $\neg rainbow \land$ <disjunct>



Circuits: Example

- Deterministic disjunctions
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable
- Decomposable conjunctions
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables

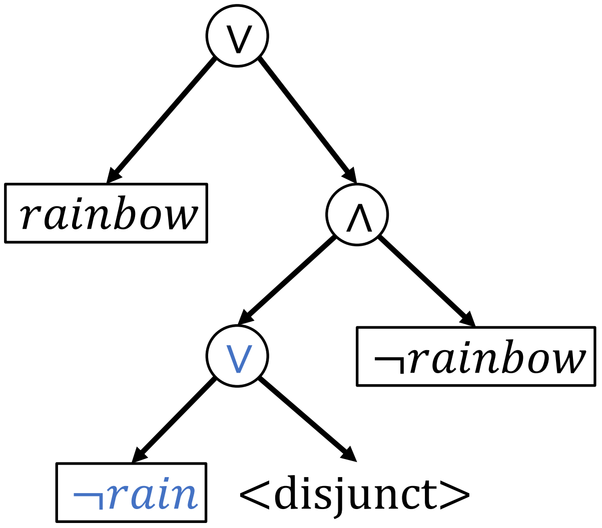


- E.g., $\neg sun \vee \neg rain \vee rainbow$
  - <disjunct> $\vee\ rainbow$
    - Determinism:
      <disjunct> can only be true if $rainbow$ is not
      - Add $\neg rainbow$ to disjunct:
        $\neg rainbow \wedge$ <disjunct>
      - <disjunct> now part of a conjunction with $\neg rainbow$
        - Decomposability: May not contain $Rainbow$

Circuits: Example

- Deterministic disjunctions
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable
- Decomposable conjunctions
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables

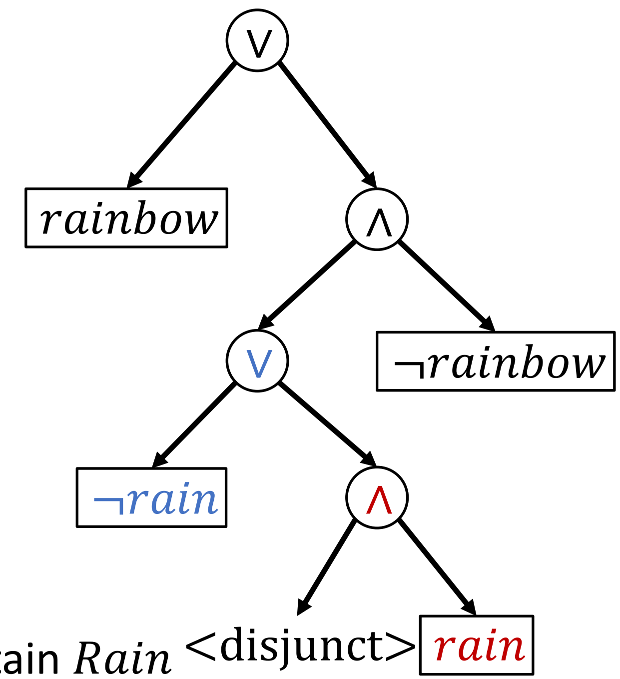
- E.g., $\neg sun \lor \neg rain \lor rainbow$
  - <disjunct> $\lor \neg rain$
    - Determinism:
      <disjunct> can only be true if $\neg rain$ is not, i.e., if $rain$ is
      - Add $rain$ to disjunct:
        $rain \land$ <disjunct>



Circuits: Example

- Deterministic disjunctions
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable
- Decomposable conjunctions
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables

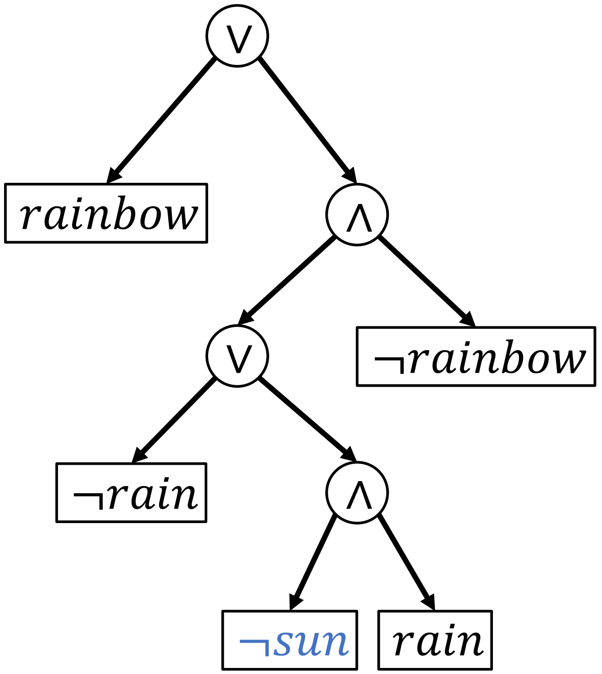
- E.g., $\neg sun \lor \neg rain \lor rainbow$
  - <disjunct> $\lor \neg rain$
    - Determinism:
      <disjunct> can only be true if $\neg rain$ is not, i.e., if $rain$ is
      - Add $rain$ to disjunct: $rain \land$ <disjunct>
      - <disjunct> now part of a conjunction with $rain$
        - Decomposability: May not contain $Rain$



Circuits: Example

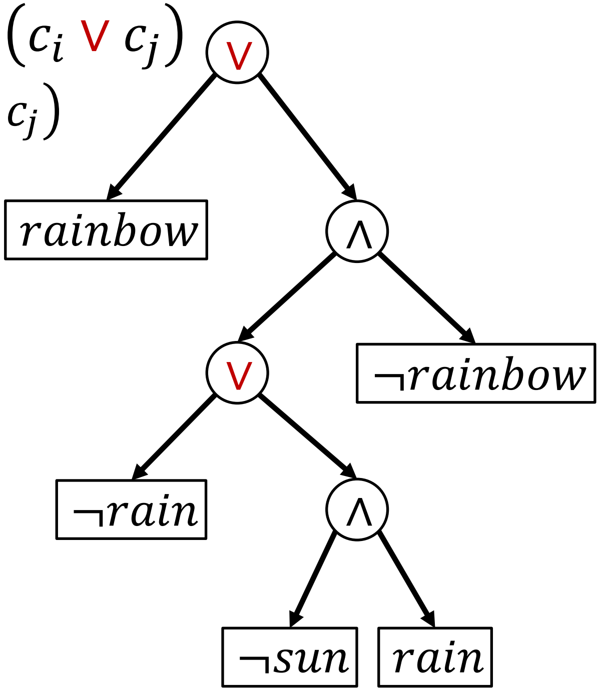
- Deterministic disjunctions
  - Only one disjunct (child node) can be true at the same time
    - I.e., their conjunction is unsatisfiable
- Decomposable conjunctions
  - Each pair of conjuncts (child nodes) must be independent
    - I.e., they cannot share any variables

- E.g., $\neg sun \lor \neg rain \lor rainbow$
  - Add as conjunct
    - Decomposability: Does not share variables with sibling node



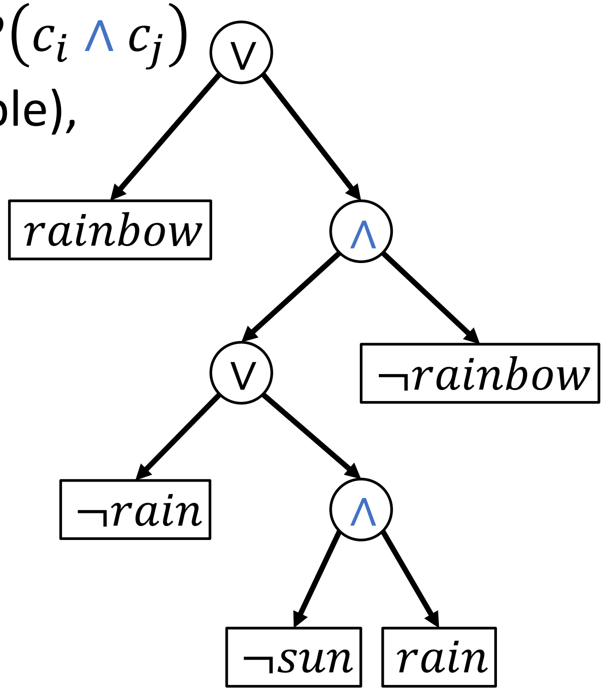
Effects of d-DNNF

- Effects of d-DNNF
  - Deterministic disjunctions
    - Only one disjunct (child node) can be true at the same time
      - I.e., their conjunction is unsatisfiable
  - Assume children c_i, c_j represent probabilities p_i, p_j
    - Node then represents probability of $P(c_i \vee c_j)$
      - $P(c_i \vee c_j) = P(c_i) + P(c_j) - P(c_i \wedge c_j)$
    - If only c_i or c_j can be true at a time, $P(c_i \wedge c_j) = 0$, i.e.,
      - $P(c_i \vee c_j) = P(c_i) + P(c_j)$
  - Can replace $\vee$ with $+$ for inference calculations



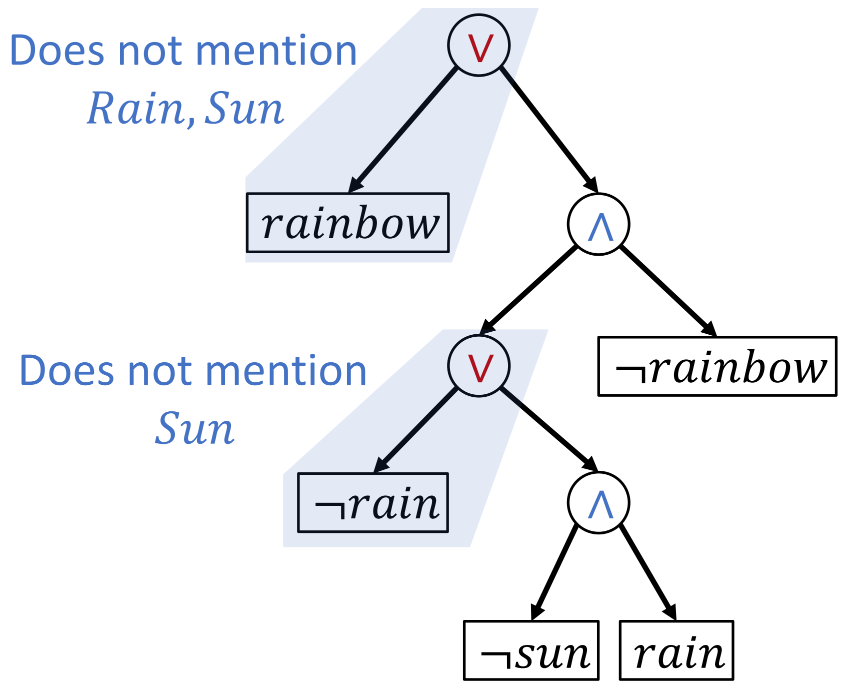
Effects of d-DNNF

- Effects of d-DNNF
  - Decomposable conjunctions
    - Each pair of conjuncts (child nodes) must be independent
      - I.e., they cannot share any variables
  - Assume children c_i, c_j represent probabilities p_i, p_j
    - Node then represents probability of $P(c_i \wedge c_j)$
    - If c_i and c_j independent (decomposable), then $P(c_i \wedge c_j) = P(c_i) \cdot P(c_j)$
  - Can replace $\wedge$ with $\cdot$ for inference calculations



Smooth d-DNNF (sd-DNNF)

- Smooth circuits: constant runtime for certain queries
  - Any pair of disjuncts mentions the same set of variables
  - E.g., $\neg sun \lor \neg rain \lor rainbow$
    - Two disjunctions that do not fulfil the smoothness property

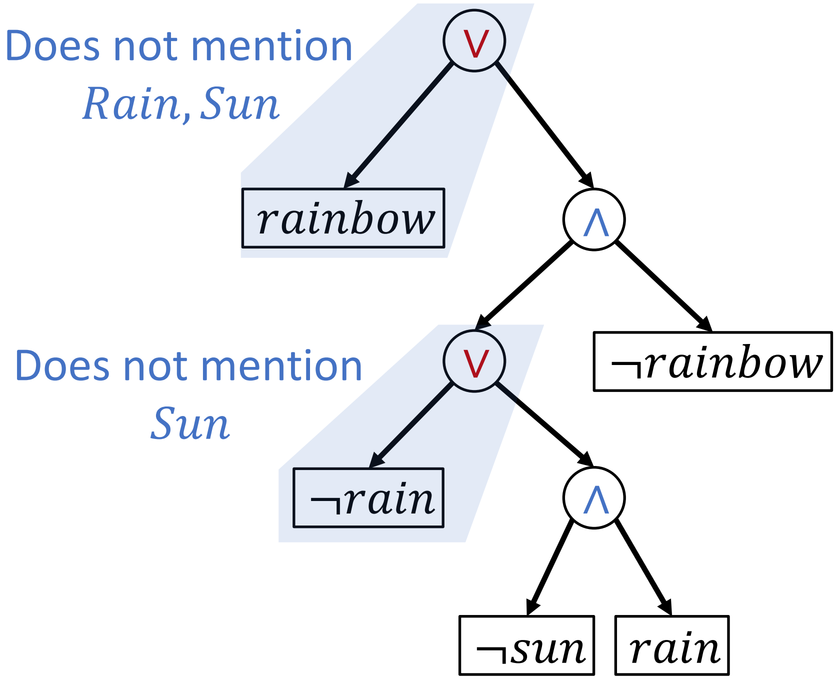


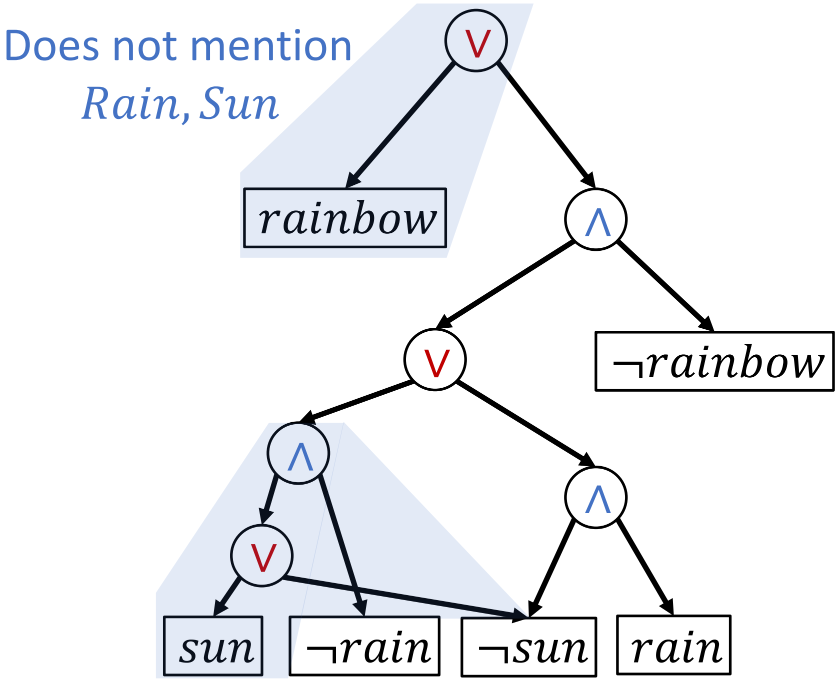
- Rules for conversion
  - For each negation of a positive literal l not appearing, replace l by
    $l \lor (\neg l \land false)$
  - For each variable A not mentioned in a disjunct <disjunct>, add $a \lor \neg a$ with a conjunction to <disjunct>:
    <disjunct> $\land (a \lor \neg a)$

Smooth d-DNNF (sd-DNNF)

- Add $sun \vee \neg sun$ to $\neg rain$, replacing $\neg rain$ with

$$\neg rain \wedge (sun \vee \neg sun)$$

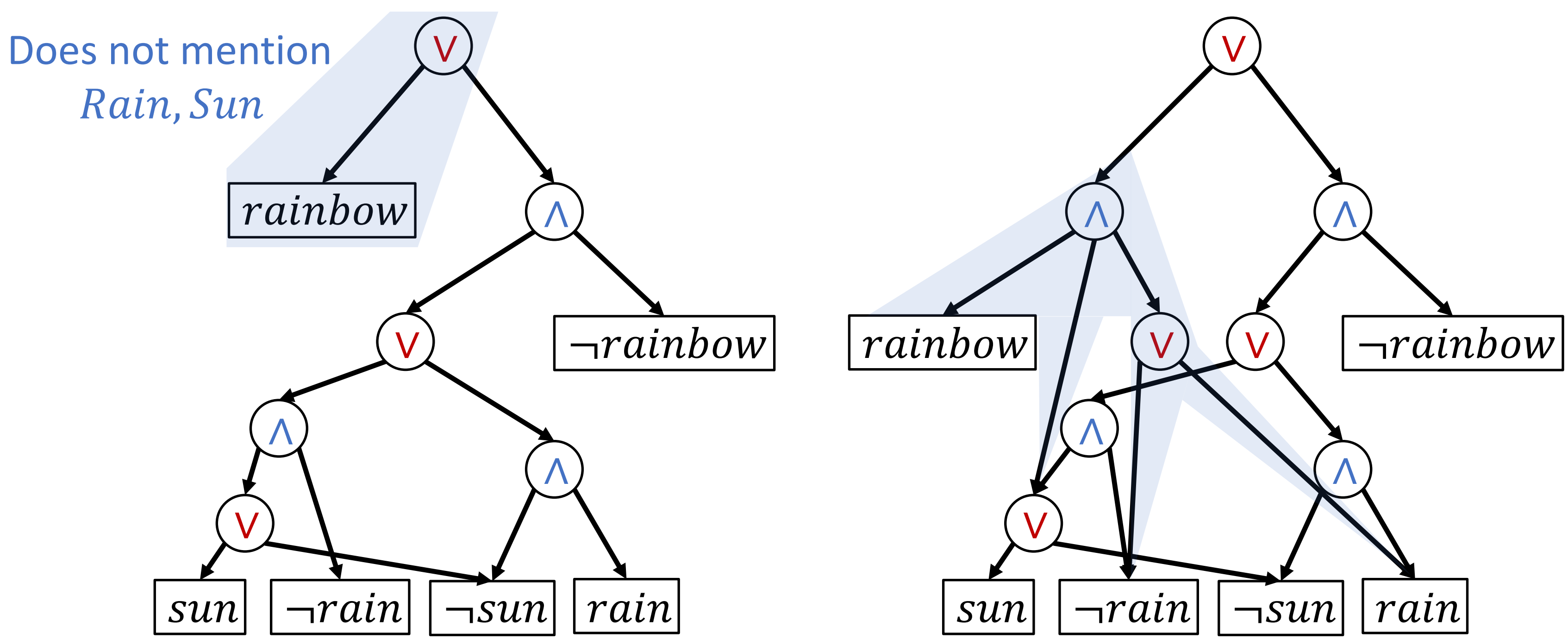




Smooth d-DNNF (sd-DNNF)

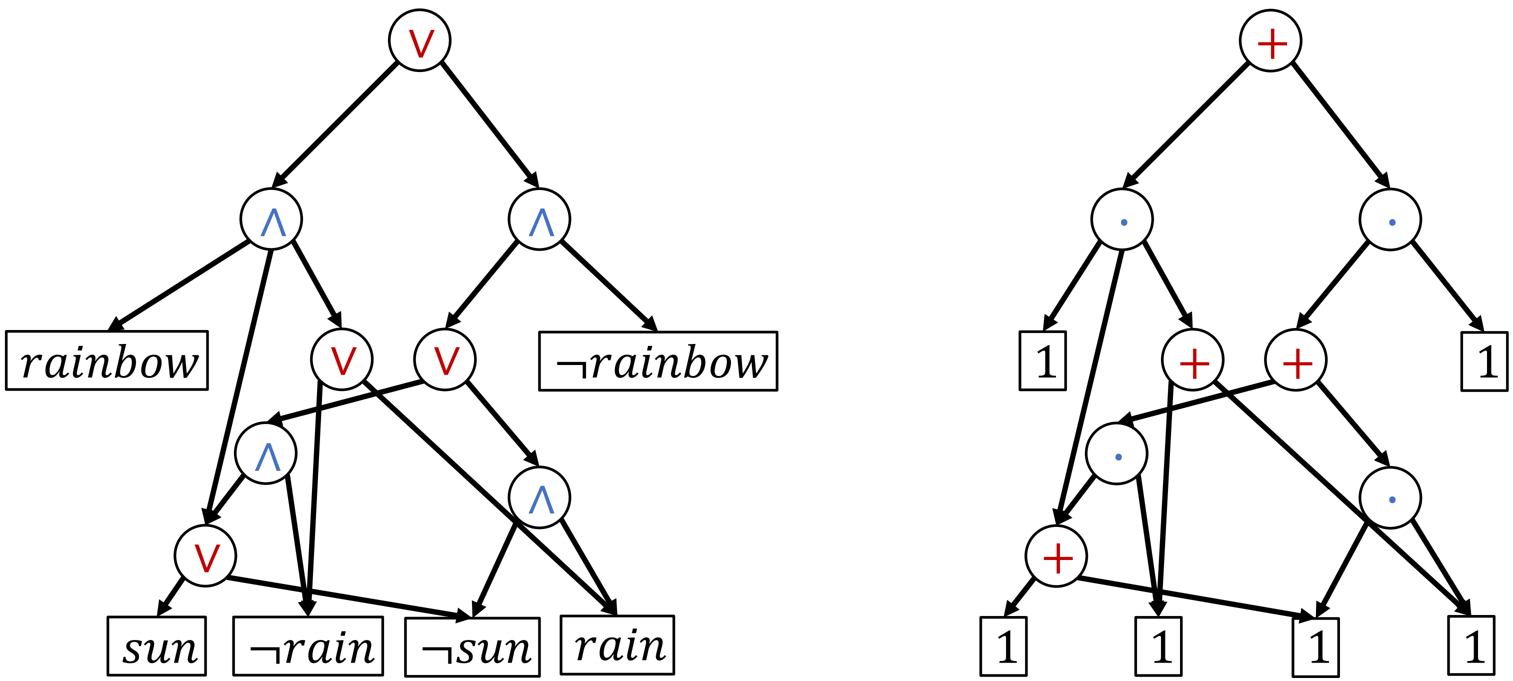
- Add $sun \lor \neg sun$ and $rain \lor \neg rain$, replacing $rainbow$ with

$$rainbow \land (sun \lor \neg sun) \land (rain \lor \neg rain)$$



Circuit for Model Counting

- Model counting problem: Count how many models fulfil a sentence
- Model counting arithmetic circuit
  - Replace ∧ with ·
  - Replace ∨ with +
  - Replace leaves with 1’s

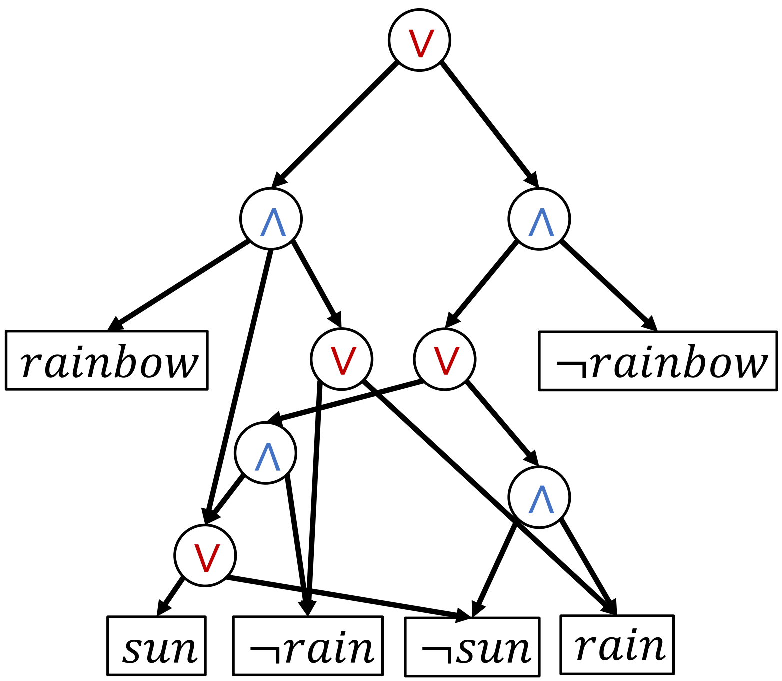


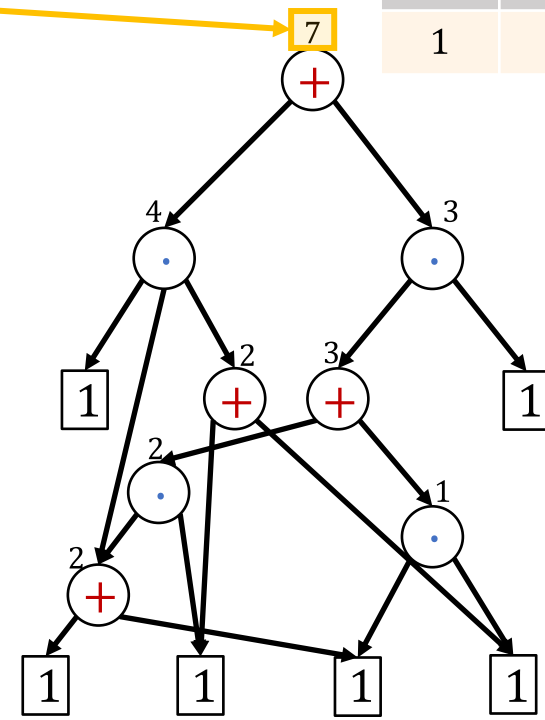
$sun \land rain \Rightarrow rainbow$

$rain$	sun	$rainbow$
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
~~1~~	~~1~~	~~0~~
1	1	1

Circuit for Model Counting

- Propagate 1's upwards (from leaves to root), using arithmetic operations in inner nodes to combine incoming numbers
    - Result at root: Model count



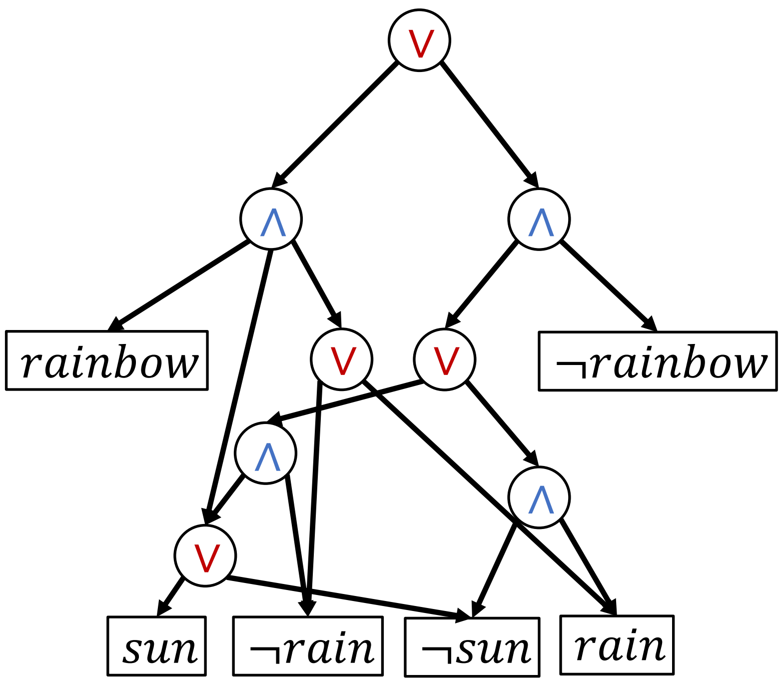


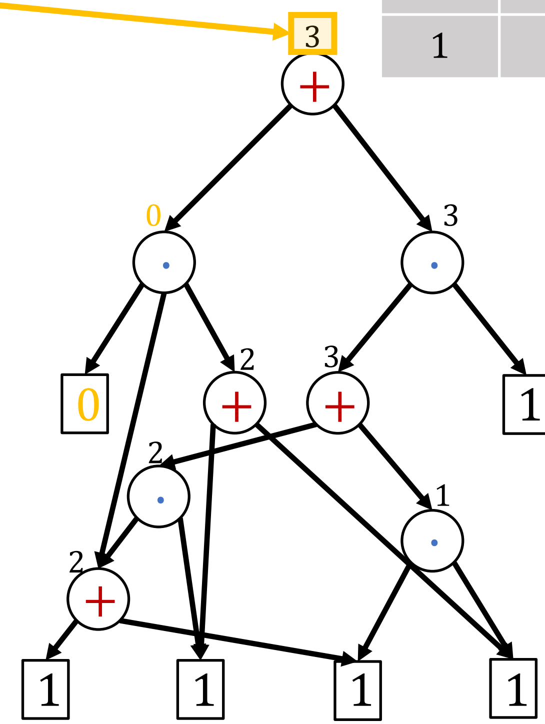
$sun \wedge rain \Rightarrow rainbow$

Conditioning

- To get model count of models fulfilling certain truth values
  - Replace 1's with zeros where literal contradicts truth values
    - Could minimise circuit
  - E.g., condition on $\neg rainbow$

$rain$	sun	$rainbow$
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
~~1~~	~~1~~	~~0~~
1	1	1

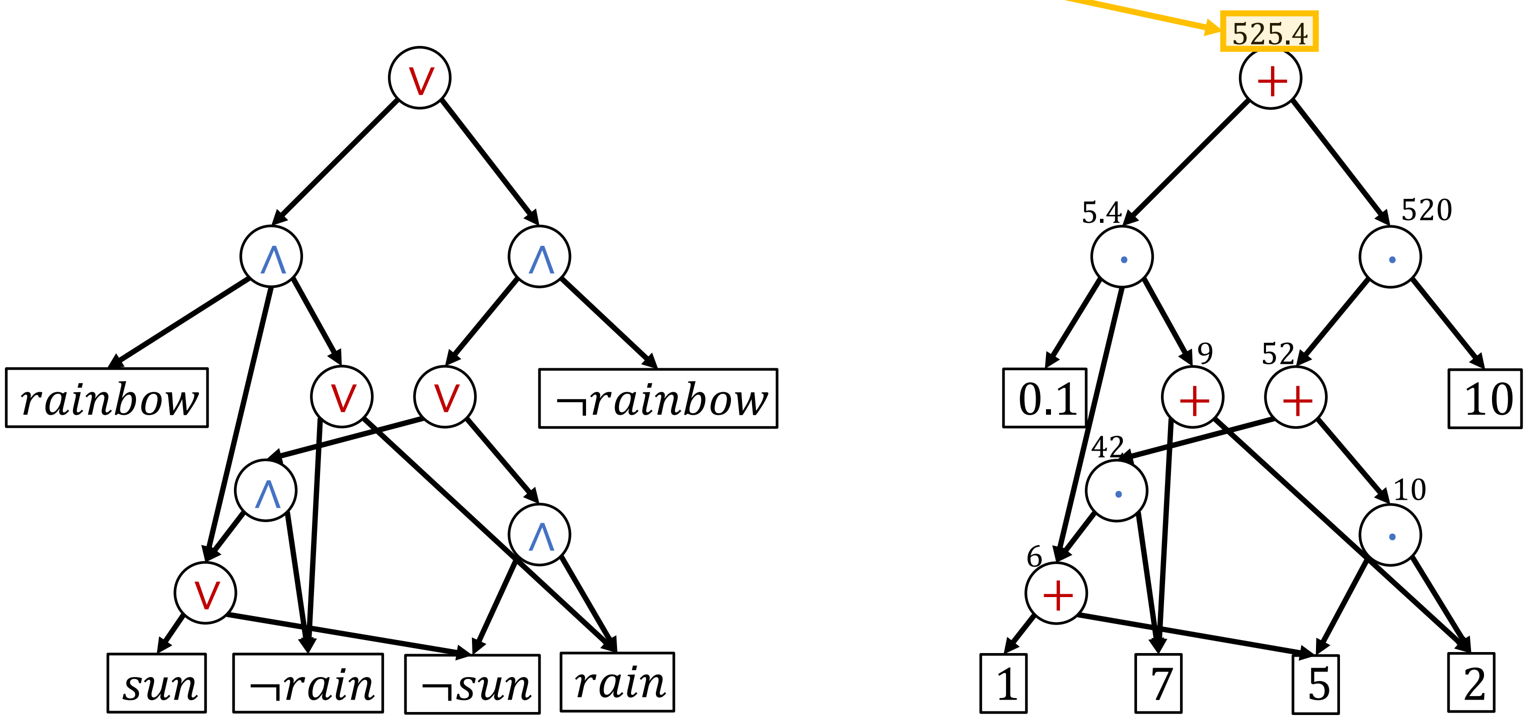




Circuit for Weighted Model Counting

- Replace literals with weights in leaves and propagate weights upwards
  - Computes $WMC(\varphi, weight)$

$weight(sun) = 1$
$weight(\neg sun) = 5$
$weight(rain) = 2$
$weight(\neg rain) = 7$
$weight(rainbow) = 0.1$
$weight(\neg rainbow) = 10$

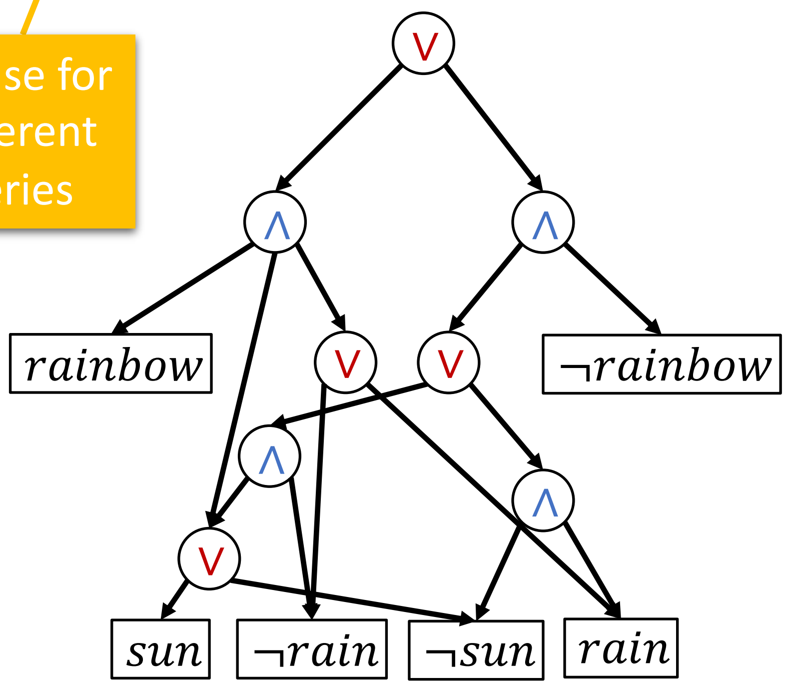


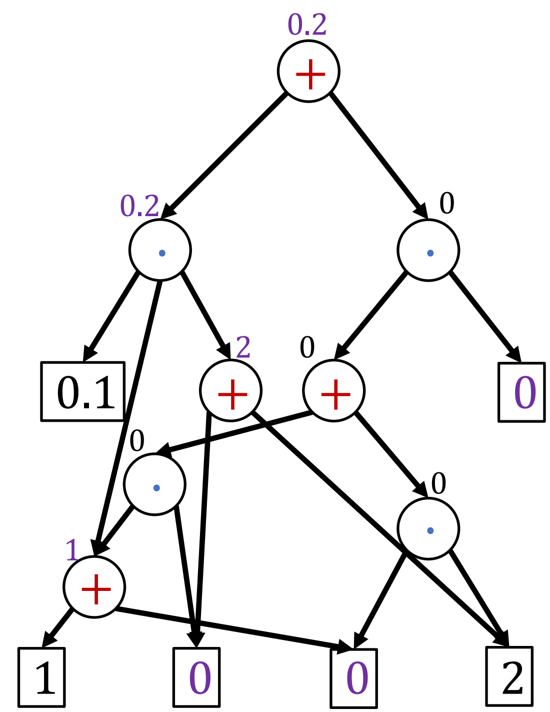
Circuit for Weighted Model Counting

- For probabilities of worlds or query terms ω, condition on truth values
  1. Compute $WMC(\varphi, weight)$
  2. Compute $WMC(\varphi \wedge \omega, weight)$
  3. Divide the two counts

$$P(\omega = \{sun, rain, rainbow\}) = \frac{WMC(\varphi \wedge \omega, weight)}{WMC(\varphi, weight)} = \frac{0.2}{525.4} = 0.00038$$

Reuse for different queries





Knowledge Compilation

- Solve the weighted model counting problem by knowledge compilation
- Given a theory Δ and a set of queries $\{P(q_i|e)\}_{i=1}^{m}$
  - Build a circuit for theory Δ (a conjunction of sentences)
  - Make the circuit a WMC circuit
    - Replace inner nodes with arithmetic operations
    - Replace leaves with weights
  - Condition on given evidence e
    - Replace weights with 0 where literals contradict e
  - Calculate $WMC(\Delta \wedge e, weight)$ in the circuit
    - By propagating the weights upwards
  - For each query $P(q_i|e)$ in the circuit
    - Compute $WMC(\Delta \wedge e \wedge q_i, weight)$
    - Return or store $P(q_i|e) = \frac{WMC(\Delta \wedge e \wedge q_i, weight)}{WMC(\Delta \wedge e, weight)}$

Knowledge Compilation

Propositional → First-order

- If input theory is in FOL-DC ((function-free) first-order logic with domain constraints), one could ground the theory given domains and build a circuit for the grounded theory
  - FOL-DS includes intensional conjunctions and disjunctions (∀, ∃)
  - Leads to repeated structures in circuit
- Combine repeated structures using new inner node types for intensional conjunctions and disjunctions (∀, ∃)
- We are not going into every detail of FOKC;
  - For complete description, analysis, and discussion, see the PhD thesis by Guy Van den Broeck

Weighted First-order Model Counting

- Define a weighted first-order model counting problem using a weighted first-order model count (WFOMC)

$$WFOMC(\Delta, w_T, w_F) = \sum_{\substack{\omega=\omega_T\cup\omega_F \\ \omega\in\Omega_\Delta}} \prod_{l\in\omega_T} w_T\big(pred(l)\big) \prod_{l\in\omega_F} w_F\big(pred(l)\big)$$

  - Δ a theory in FOL-DC
  - w_T a weight function for predicates being positive
  - w_F a weight function for predicates being negative
  - Ω_Δ the set of worlds (i.e., models in logics) of Δ
  - $pred(l)$ a function mapping a literal l to its predicate
- Query can be answered by computing

$$P(q_i|e) = \frac{WFOMC(\Delta \wedge e \wedge q_i, w_T, w_F)}{WFOMC(\Delta \wedge e, w_T, w_F)}$$

Guy Van den Broeck, Nima Taghipour, Wannes Meert, Jesse Davis, and Luc De Raedt: Lifted Probabilistic Inference by First-order Knowledge Compilation. In: IJCAI-11 Proceedings of the 22nd International Joint Conference on Artificial Intelligence, 2011.

- Theory: one sentence

$$\forall X \in \text{People} : smokes(X) \Rightarrow cancer(X)$$

  - People $= \{x_1, x_2\}$
- Weight functions
  - $w_T\big(smokes(X)\big) = 3$
  - $w_F\big(\neg smokes(X)\big) = 1$
  - $w_T\big(cancer(X)\big) = 6$
  - $w_F\big(\neg cancer(X)\big) = 2$
- Model count: 9
  - Worlds that fulfil the theory

$$WFOMC(\Delta, w_T, w_F) = \sum_{\substack{\omega = \omega_T \cup \omega_F \\ \omega \in \Omega_\Delta}} \prod_{l \in \omega_T} w_T(pred(l)) \prod_{l \in \omega_F} w_F(pred(l))$$

$s(x_1)$	$c(x_1)$	$s(x_2)$	$c(x_2)$	Weight	
0	0	0	0	$1 \cdot 2 \cdot 1 \cdot 2$	4
0	0	0	1	$1 \cdot 2 \cdot 1 \cdot 6$	12
~~0~~	~~0~~	~~1~~	~~0~~	~~$1 \cdot 2 \cdot 3 \cdot 2$~~	~~12~~
0	0	1	1	$1 \cdot 2 \cdot 3 \cdot 6$	36
0	1	0	0	$1 \cdot 6 \cdot 1 \cdot 2$	12
0	1	0	1	$1 \cdot 6 \cdot 1 \cdot 6$	36
~~0~~	~~1~~	~~1~~	~~0~~	~~$1 \cdot 6 \cdot 3 \cdot 2$~~	36
0	1	1	1	$1 \cdot 6 \cdot 3 \cdot 6$	108
~~1~~	~~0~~	~~0~~	~~0~~	~~$3 \cdot 2 \cdot 1 \cdot 2$~~	~~12~~
~~1~~	~~0~~	~~0~~	~~1~~	~~$3 \cdot 2 \cdot 1 \cdot 6$~~	36
~~1~~	~~0~~	~~1~~	~~0~~	~~$3 \cdot 2 \cdot 3 \cdot 2$~~	36
~~1~~	~~0~~	~~1~~	~~1~~	~~$3 \cdot 2 \cdot 3 \cdot 6$~~	~~108~~
1	1	0	0	$3 \cdot 6 \cdot 1 \cdot 2$	36
1	1	0	1	$3 \cdot 6 \cdot 1 \cdot 6$	108
~~1~~	~~1~~	~~1~~	~~0~~	~~$3 \cdot 6 \cdot 3 \cdot 2$~~	~~108~~
1	1	1	1	$3 \cdot 6 \cdot 3 \cdot 6$	324
				+	**676**

- Theory: one sentence

$$\forall X \in \text{People} : smokes(X) \Rightarrow cancer(X)$$

  - People $= \{x_1, x_2\}$

- Weight functions
  - $w_T(smokes(X)) = 3$
  - $w_F(\neg smokes(X)) = 1$
  - $w_T(cancer(X)) = 6$
  - $w_F(\neg cancer(X)) = 2$

$$P(s(x_1)) = \frac{WFOMC(\Delta \wedge s(x_1), w_T, w_F)}{WFOMC(\Delta, w_T, w_F)} = \frac{36 + 108 + 324}{676} = \frac{468}{676} = 0.692$$

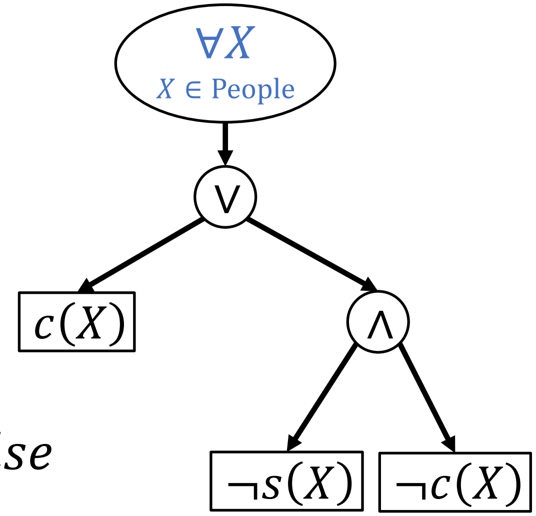
$s(x_1)$	$c(x_1)$	$s(x_2)$	$c(x_2)$	Weight	
0	0	0	0	$1 \cdot 2 \cdot 1 \cdot 2$	4
0	0	0	1	$1 \cdot 2 \cdot 1 \cdot 6$	12
0	0	1	0	$1 \cdot 2 \cdot 3 \cdot 2$	12
0	0	1	1	$1 \cdot 2 \cdot 3 \cdot 6$	36
0	1	0	0	$1 \cdot 6 \cdot 1 \cdot 2$	12
0	1	0	1	$1 \cdot 6 \cdot 1 \cdot 6$	36
0	1	1	0	$1 \cdot 6 \cdot 3 \cdot 2$	36
0	1	1	1	$1 \cdot 6 \cdot 3 \cdot 6$	108
1	0	0	0	$3 \cdot 2 \cdot 1 \cdot 2$	12
1	0	0	1	$3 \cdot 2 \cdot 1 \cdot 6$	36
1	0	1	0	$3 \cdot 2 \cdot 3 \cdot 2$	36
1	0	1	1	$3 \cdot 2 \cdot 3 \cdot 6$	108
1	1	0	0	$3 \cdot 6 \cdot 1 \cdot 2$	36
1	1	0	1	$3 \cdot 6 \cdot 1 \cdot 6$	108
1	1	1	0	$3 \cdot 6 \cdot 3 \cdot 2$	108
1	1	1	1	$3 \cdot 6 \cdot 3 \cdot 6$	324
				+	676

First-order (FO) Circuits

- Assume theory in Skolem normal form + CNF
  - Sequence of intensional conjunctions in CNF
  - E.g., with $s = smokes, c = cancer$

$$\forall X \in \text{People} \; : s(X) \Rightarrow c(X)$$
$$\equiv \forall X \in \text{People} \; : \neg s(X) \vee c(X)$$

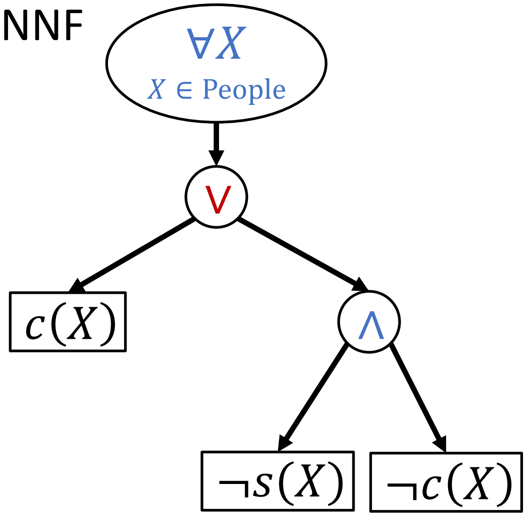
- FO circuit (extract)
  - Inner nodes:
    - Extensional conjunctions/disjunctions (as before)
    - Set conjunctions
  - Leaf nodes
    - Positive and negative predicates, $true$, $false$
  - Full + construction:
    see PhD thesis by Guy Van den Broeck



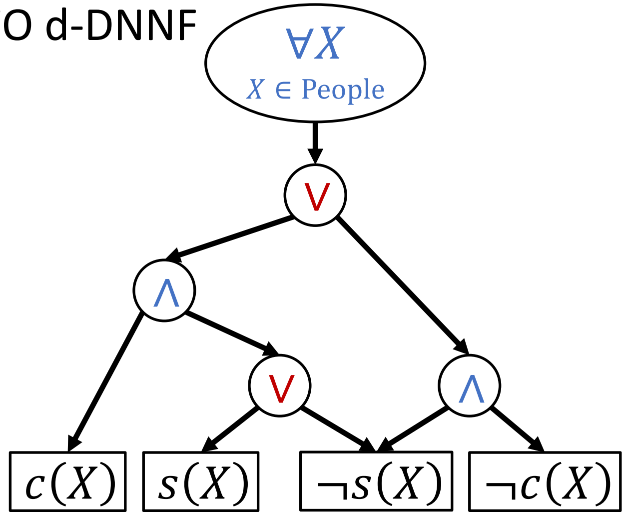
Smooth FO d-DNNF Circuits

- Properties
  - Deterministic disjunctions
    - Only one disjunct (child node) can be true at the same time
  - Decomposable conjunctions
    - Each pair of conjuncts (child nodes) must be independent
  - Smoothness
    - Each disjunct contains the same variables

FO d-DNNF

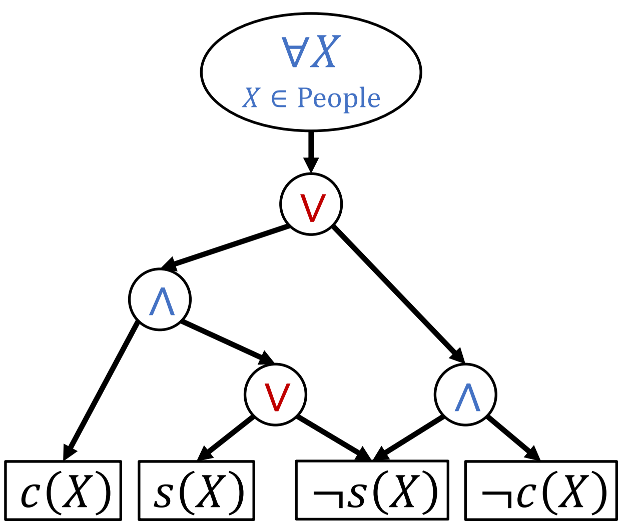


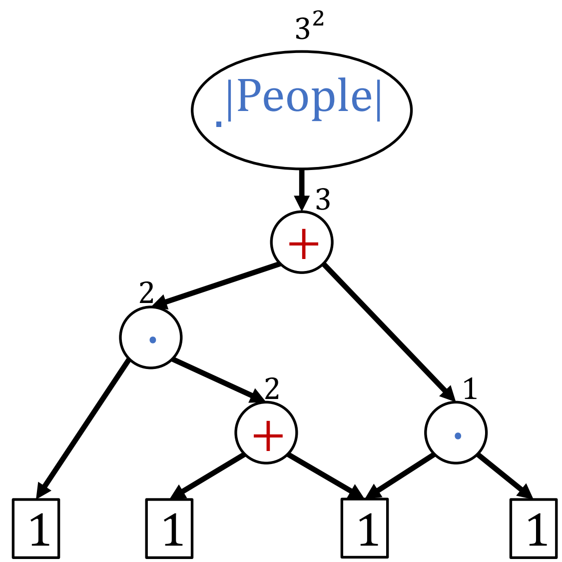
Smooth FO d-DNNF



Arithmetic FO d-DNNF Circuits

- Replace
  - Replace $\wedge$ with $\cdot$
  - Replace $\vee$ with $+$
  - Replace $\forall$ with exponentiation for $|\mathrm{Domain}|$
  - Replace leaves with 1's
  - E.g., with $|\mathrm{People}| = |\{x_1, x_2\}| = 2$



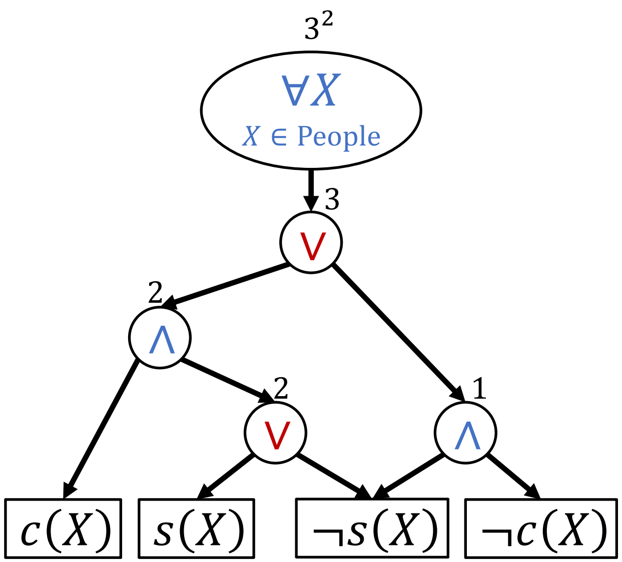


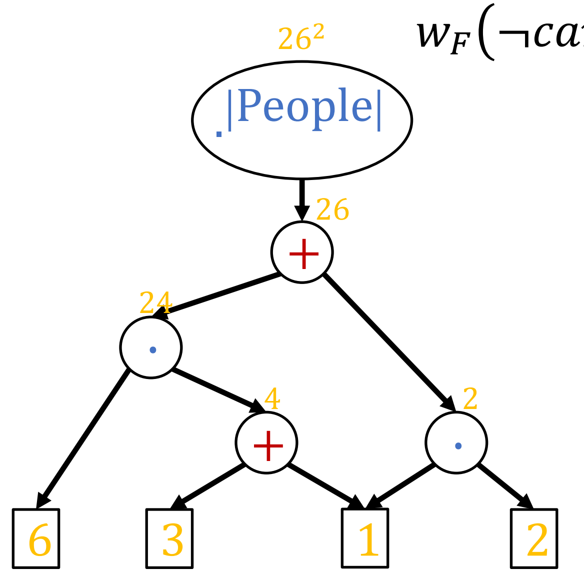
WFOMC Circuits

- Replace
  - Replace $\wedge$ with $\cdot$
  - Replace $\vee$ with $+$
  - Replace $\forall$ with exponentiation for $|\text{Domain}|$
  - Replace leaves with weights
  - E.g., with $|\text{People}| = |\{x_1, x_2\}| = 2$

$$WFOMC(\Delta, w_T, w_F) = \sum_{\substack{\omega = \omega_T \cup \omega_F \\ \omega \in \Omega_\Delta}} \prod_{l \in \omega_T} w_T(pred(l)) \prod_{l \in \omega_F} w_F(pred(l))$$

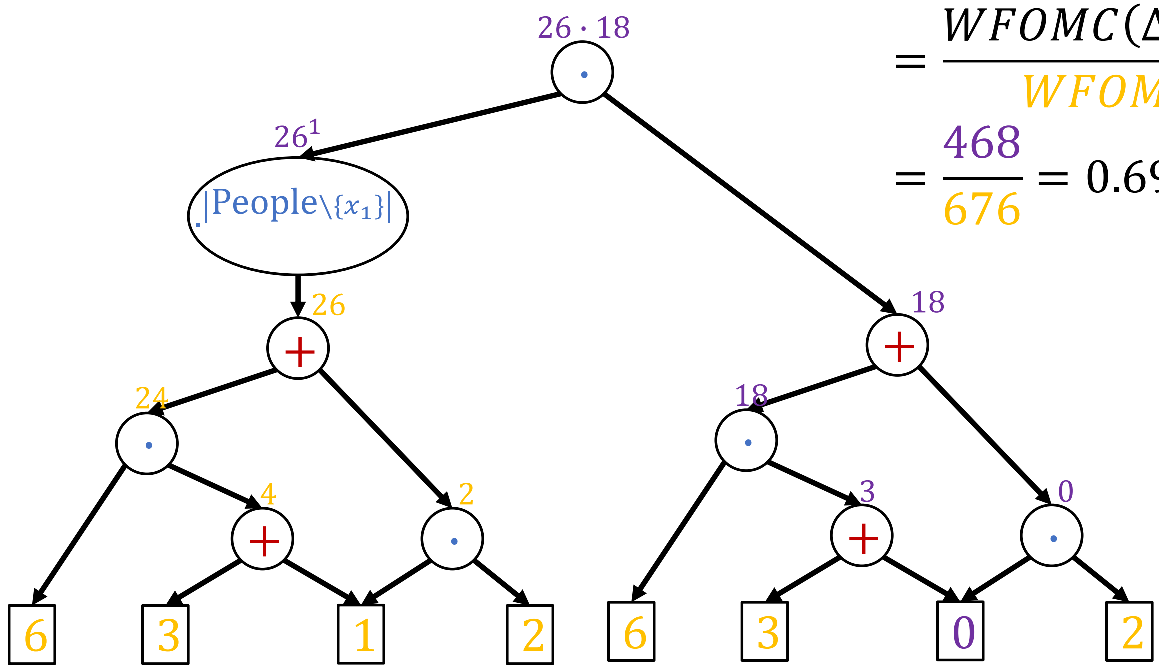
$$w_T(smokes(X)) = 3$$
$$w_F(\neg smokes(X)) = 1$$
$$w_T(cancer(X)) = 6$$
$$w_F(\neg cancer(X)) = 2$$





WFOMC Circuits

- Given $P(q_i|e)$
  - Basically, compile a circuit for $\Delta \wedge e \wedge q_i$ reusing components from the circuit of $\Delta \wedge e$
  - E.g., $P(s(x_1))$ with $|\text{People}| = |\{x_1, x_2\}| = 2$



$$P(s(x_1)) = \frac{WFOMC(\Delta \wedge s(x_1), w_T, w_F)}{WFOMC(\Delta, w_T, w_F)} = \frac{468}{676} = 0.692$$

Conditioning in FO Circuits

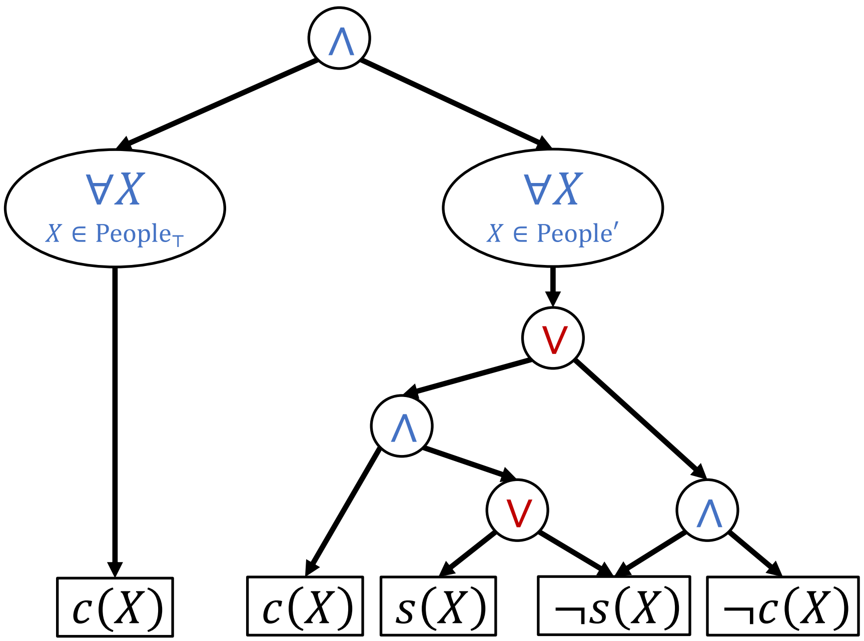
- Evidence on
  - Propositional variables L
    - Replace leaf values with 0 where literal contradicts observation
      - As in propositional circuits
  - Unary variable $L(X)$
    - For *each* variable $L(X)$ that one wants to condition on,
      - Replace FOL-DC formula with three copies with additional domain constraints, possibly simplify formula based on observation
        1. $X \in D_\top$ for observations $l(x)$
        2. $X \in D_\bot$ for observations $\neg l(x)$
        3. $X \notin D_\top \land X \notin D_\bot$ no observations
    - Compile a circuit for the extended theory
    - Given specific evidence, domains for $D_\top, D_\bot$ are determined
      - Might be empty
  - Binary variable $L(X, Y)$
    - Can compile a circuit, no longer polynomial in time (reduction of #2SAT problem)

Guy Van den Broeck and Jesse Davis: Conditioning in First-Order Knowledge Compilation and Lifted Probabilistic Inference. In: AAAI-12 Proceedings of the 26th AAAI Conference on Artificial Intelligence, 2012.

Conditioning in FO Circuits

- E.g., $\forall X \in \text{People} \; : s(X) \Rightarrow c(X)$ and $S(X)$
  1. $\forall X \in \text{People}_\top \; : s(X) \Rightarrow c(X) \overset{s(X)}{\equiv} \forall X \in \text{People}_\top \; : c(X)$
  2. $\forall X \in \text{People}_\bot \; : s(X) \Rightarrow c(X) \overset{\neg s(X)}{\equiv} \forall X \in \text{People}_\bot \; : true$
  3. $\forall X \in \text{People}, X \notin \text{People}_\top, X \notin \text{People}_\bot : s(X) \Rightarrow c(X)$

  - Delete Formula 2 as it is always true
  - If one also wants to condition on $C(X)$, theory becomes larger again:
    - Formulas (1) and (3) contain $C(X)$ and therefore need to be replaced by three formulas, then simplify



First-order Knowledge Compilation (FOKC)

- Solve the weighted first-order model counting problem by knowledge compilation
- Given
  - a theory Δ in FOL-DC in Skolem NNF
  - a weight function w_T for predicates being positive
  - a weight function w_F for predicates being negative
  - and a set of queries $\{P(q_i|e)\}_{i=1}^{m}$ with evidence for variables $\boldsymbol{E}$
- Do
  - Build a WFOMC circuit $\mathcal{C}_\Delta$ for Δ, also preparing for evidence on $\boldsymbol{E}$
  - Condition on e
  - Calculate $WFOMC(\Delta \wedge e, w_T, w_F)$ in $\mathcal{C}_\Delta$
  - For each query $P(q_i|e)$
    - Build a WFOMC circuit $\mathcal{C}_{\Delta,q_i}$ for $\Delta \wedge q_i$ conditioned on e
    - Compute $WFOMC(\Delta \wedge e \wedge q_i, w_T, w_F)$ in $\mathcal{C}_{\Delta,q_i}$
    - Return or store $P(q_i|e) = \frac{WFOMC(\Delta \wedge e \wedge q_i, w_T, w_F)}{WFOMC(\Delta \wedge e, w_T, w_F)}$

FOKC

MLNs for WFOMCs

- Weights in MLNs specified for formulas instead of single predicates
  - E.g., example from the beginning
    - $(\ln 7, travel(X) \land epid \land sick(X))$
    - $(\ln 2, \neg travel(X) \lor \neg epid \lor \neg sick(X))$
- Trick:
  - Introduce a new predicate θ_i containing all free variables of ψ_i as equivalent to ψ_i
    - E.g.,
      - $\forall X \in \text{People} : \theta_1(X) \Leftrightarrow (travel(X) \land epid \land sick(X))$
      - $\forall X \in \text{People} : \theta_2(X) \Leftrightarrow (\neg travel(X) \lor \neg epid \lor \neg sick(X))$
  - Specify weight functions such that θ_i takes the weight of ψ_i
    - $w_T(\theta_1(X)) = \exp(\ln 7) = 7$
    - $w_T(\theta_2(X)) = \exp(\ln 2) = 2$
    - All other predicates and $\neg\theta_1, \neg\theta_2$ are mapped to 1 by both w_T, w_F

WFOMC Reduction

- Formally, given an MLN $\Psi = \{(w_i, \psi_i)\}_{i=1}^{n}$
  - Transform each weighted formula (w_i, ψ_i) into an FOL-DC formula

$$\forall \boldsymbol{X}_i, cs_i : \theta_i(\boldsymbol{X}_i) \Leftrightarrow \psi_i$$

  - where
    - $\boldsymbol{X}_i$ are the free variables in ψ_i
    - cs_i is the constraint set that enforces the domain constraints as given by the MLN
    - $\theta_i(\boldsymbol{X}_i)$ is a new predicate containing all free variables of ψ_i
  - Specify weight functions w_T, w_F such that for each
    - $w_T\big(\theta_i(\boldsymbol{X}_i)\big) = \exp(w_i)$
    - $w_T(p_i) = 1$ for all predicates p_i occurring in Ψ
    - $w_F\big(\theta_i(\boldsymbol{X}_i)\big) = w_F(p_i) = 1$
- Continue with knowledge compilation

Example

- Given
  - $\left(\ln 7, travel(X) \wedge epid \wedge sick(X)\right)$
  - $\left(\ln 2, \neg travel(X) \vee \neg epid \vee \neg sick(X)\right)$
- Resulting theory
  - with $t = travel, e = epid, s = sick$
    - $\forall X \in \text{People} : \theta_1(X) \Leftrightarrow \left(t(X) \wedge e \wedge s(X)\right)$
    - $\forall X \in \text{People} : \theta_2(X) \Leftrightarrow \left(\neg t(X) \vee \neg e \vee \neg s(X)\right)$
  - with weight functions
    - $w_T\left(\theta_1(X)\right) = 7$
    - $w_T\left(\theta_2(X)\right) = 2$
    - Rest mapped to 1 by both w_T, w_F
- Transform formulas into CNF

Example: Normal Form

- Transform formulas into CNF
  - $\forall X \in \text{People} \;:\; \theta_1(X) \Leftrightarrow \big(t(X) \wedge e \wedge s(X)\big)$

$$
\begin{aligned}
& \theta_1(X) \Leftrightarrow \big(t(X) \wedge e \wedge s(X)\big) && \text{(resolve } \Leftrightarrow\text{)} \\
\equiv\ & \Big(\theta_1(X) \Rightarrow \big(t(X) \wedge e \wedge s(X)\big)\Big) \wedge \Big(\theta_1(X) \Leftarrow \big(t(X) \wedge e \wedge s(X)\big)\Big) && \text{(De Morgan on } \Rightarrow\text{)} \\
\equiv\ & \Big(\neg\theta_1(X) \vee \big(t(X) \wedge e \wedge s(X)\big)\Big) \wedge \Big(\theta_1(X) \vee \neg\big(t(X) \wedge e \wedge s(X)\big)\Big) && \text{(move } \neg \text{ inward)} \\
\equiv\ & \Big(\neg\theta_1(X) \vee \big(t(X) \wedge e \wedge s(X)\big)\Big) \wedge \big(\theta_1(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)\big) && \text{(distribute } \vee\text{)} \\
\equiv\ & \big(\neg\theta_1(X) \vee t(X)\big) \wedge (\neg\theta_1(X) \vee e) \wedge \big(\neg\theta_1(X) \vee s(X)\big) \\
& \qquad \wedge \big(\theta_1(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)\big) && \text{(CNF)}
\end{aligned}
$$

- Result (each conjunct as own formula):
  - $\forall X \in \text{People} \;:\neg\theta_1(X) \vee t(X)$
  - $\forall X \in \text{People} \;:\neg\theta_1(X) \vee e$
  - $\forall X \in \text{People} \;:\neg\theta_1(X) \vee s(X)$
  - $\forall X \in \text{People} \;:\theta_1(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)$

Example: Normal Form

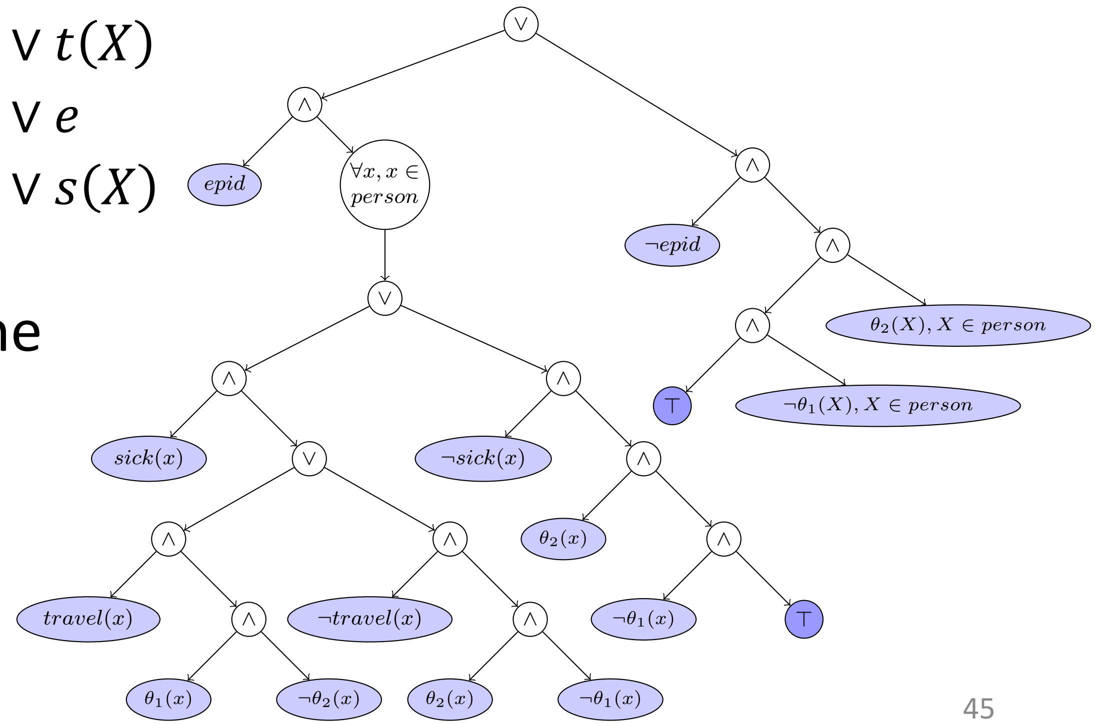
- Transform formulas into CNF
  - $\forall X \in \text{People} \ : \ \theta_2(X) \Leftrightarrow \big(\neg t(X) \vee \neg e \vee \neg s(X)\big)$

$$
\begin{aligned}
&\theta_2(X) \Leftrightarrow \big(\neg t(X) \vee \neg e \vee \neg s(X)\big) \\
\equiv\ &\Big(\theta_2(X) \Rightarrow \big(\neg t(X) \vee \neg e \vee \neg s(X)\big)\Big) \wedge \Big(\theta_2(X) \Leftarrow \big(\neg t(X) \vee \neg e \vee \neg s(X)\big)\Big) \\
\equiv\ &\big(\neg\theta_2(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)\big) \wedge \Big(\theta_2(X) \vee \neg\big(\neg t(X) \vee \neg e \vee \neg s(X)\big)\Big) \\
\equiv\ &\big(\neg\theta_2(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)\big) \wedge \Big(\theta_2(X) \vee \big(t(X) \wedge e \wedge s(X)\big)\Big) \\
\equiv\ &\big(\neg\theta_2(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)\big) \wedge \big(\theta_2(X) \vee t(X)\big) \wedge (\theta_2(X) \vee e) \wedge \big(\theta_2(X) \vee s(X)\big)
\end{aligned}
$$

- Result (each conjunct as own formula):
  - $\forall X \in \text{People} \ : \neg\theta_2(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)$
  - $\forall X \in \text{People} \ : \theta_2(X) \vee t(X)$
  - $\forall X \in \text{People} \ : \theta_2(X) \vee e$
  - $\forall X \in \text{People} \ : \theta_2(X) \vee s(X)$

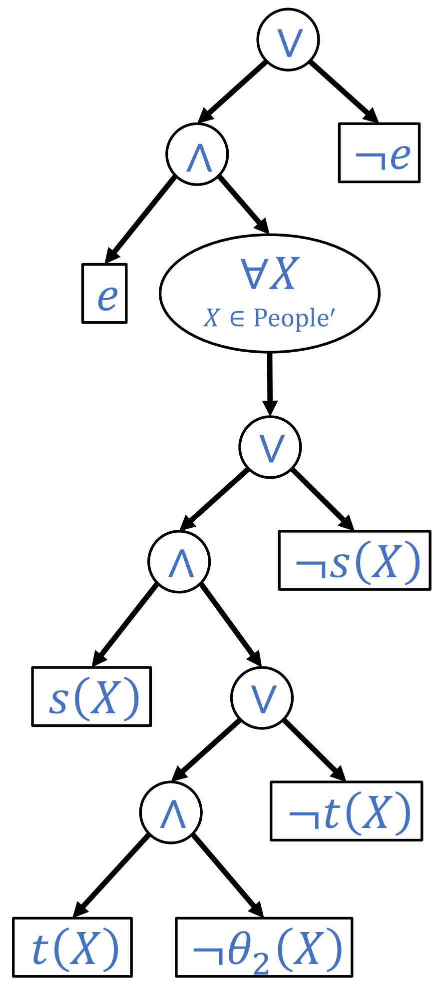
Example: FO d-DNNF Circuit

- Given theory in CNF
  - $\forall X \in \text{People} : \neg\theta_1(X) \vee t(X)$
  - $\forall X \in \text{People} : \neg\theta_1(X) \vee e$
  - $\forall X \in \text{People} : \neg\theta_1(X) \vee s(X)$
  - $\forall X \in \text{People} : \theta_1(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)$
  - $\forall X \in \text{People} : \neg\theta_2(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)$
  - $\forall X \in \text{People} : \theta_2(X) \vee t(X)$
  - $\forall X \in \text{People} : \theta_2(X) \vee e$
  - $\forall X \in \text{People} : \theta_2(X) \vee s(X)$
- Resulting FO d-DNNF circuit generated by the FOKC implementation
  - Some leaves repeated for readability



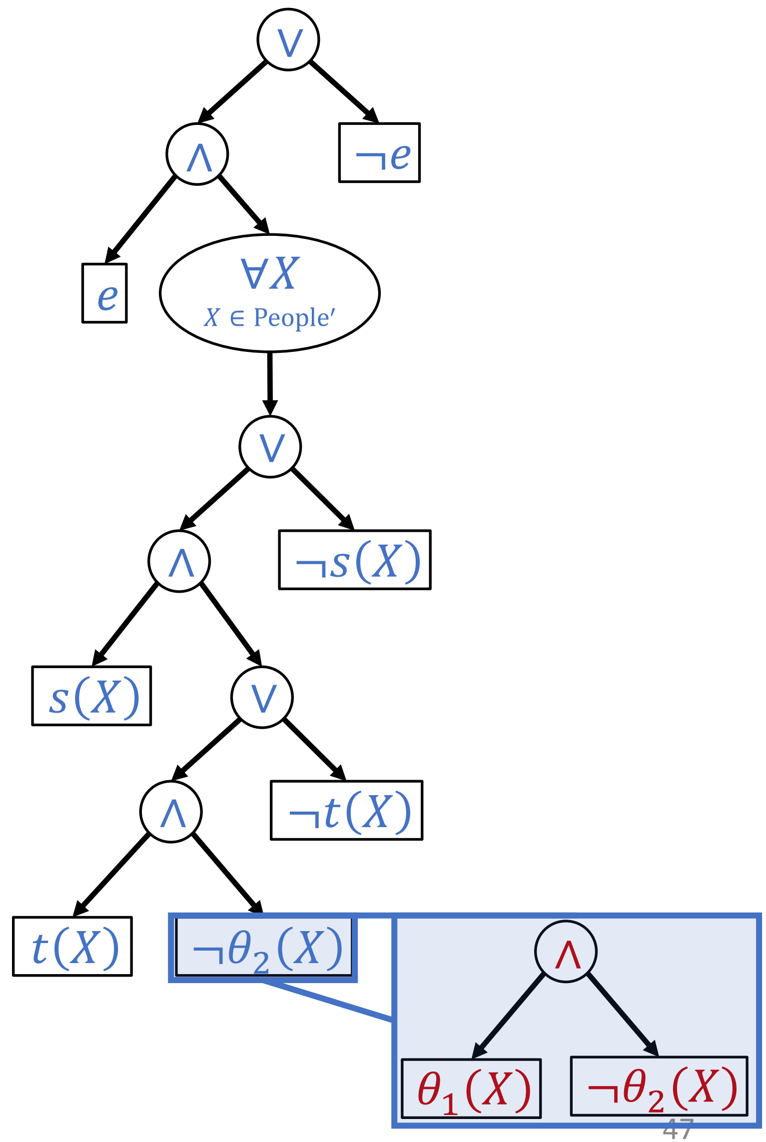
Example: FO d-DNNF Circuit

- Given theory in CNF
  1. $\forall X \in \text{People} :$ $\neg\theta_2(X) \vee \neg t(X) \vee \neg s(X) \vee \neg e$
  2. $\forall X \in \text{People} :$ $\theta_1(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)$
  3. $\forall X \in \text{People} : \neg\theta_1(X) \vee t(X)$
  4. $\forall X \in \text{People} : \neg\theta_1(X) \vee e$
  5. $\forall X \in \text{People} : \neg\theta_1(X) \vee s(X)$
  6. $\forall X \in \text{People} : \theta_2(X) \vee t(X)$
  7. $\forall X \in \text{People} : \theta_2(X) \vee e$
  8. $\forall X \in \text{People} : \theta_2(X) \vee s(X)$



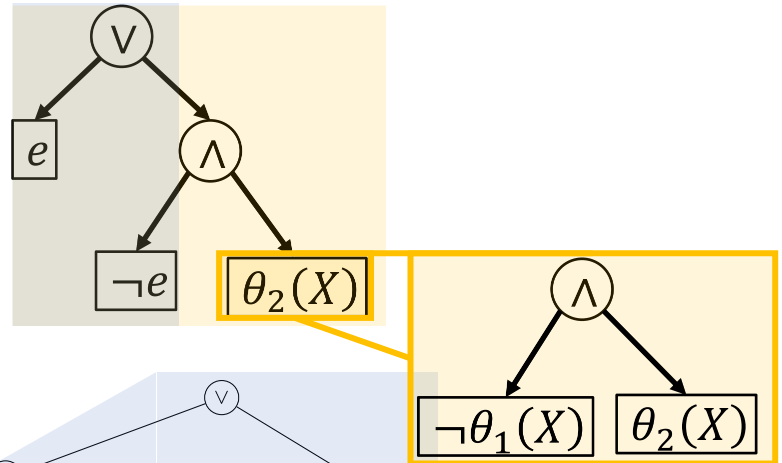
Example: FO d-DNNF Circuit

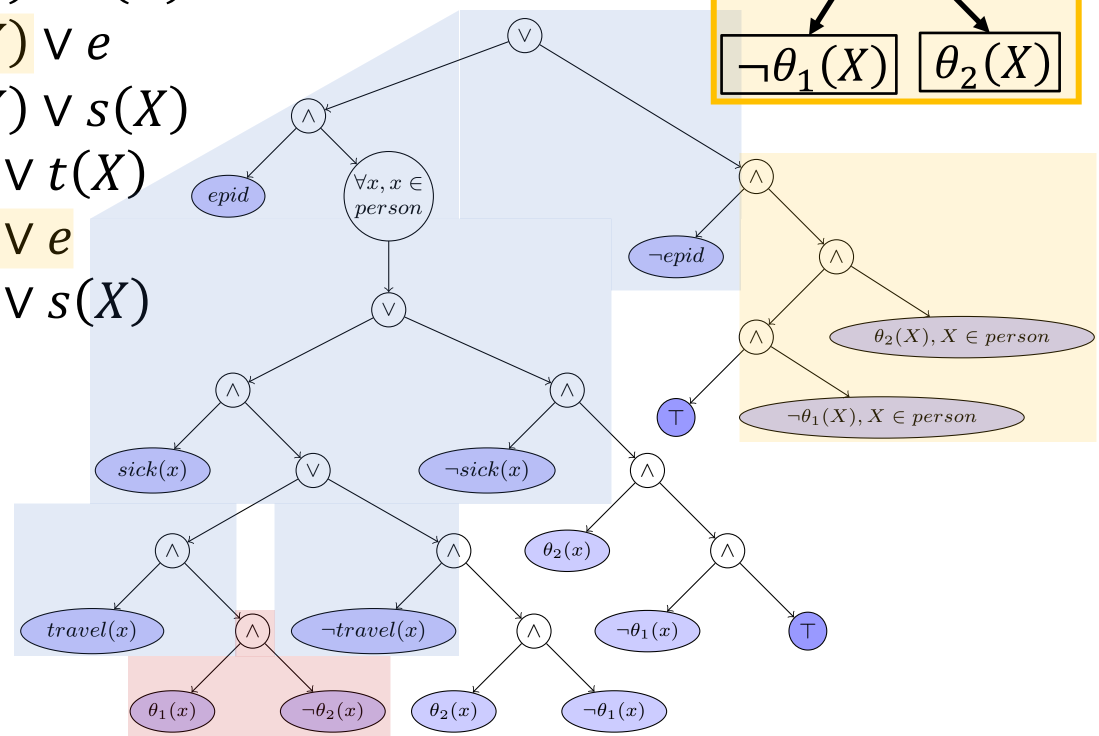
- Given theory in CNF
  1. $\forall X \in \text{People} :$ $\neg\theta_2(X) \vee \neg t(X) \vee \neg s(X) \vee \neg e$
  2. $\forall X \in \text{People} :$ $\theta_1(X) \vee \neg t(X) \vee \neg e \vee \neg s(X)$
  3. $\forall X \in \text{People} : \neg\theta_1(X) \vee t(X)$
  4. $\forall X \in \text{People} : \neg\theta_1(X) \vee e$
  5. $\forall X \in \text{People} : \neg\theta_1(X) \vee s(X)$
  6. $\forall X \in \text{People} : \theta_2(X) \vee t(X)$
  7. $\forall X \in \text{People} : \theta_2(X) \vee e$
  8. $\forall X \in \text{People} : \theta_2(X) \vee s(X)$



Example: FO d-DNNF Circuit

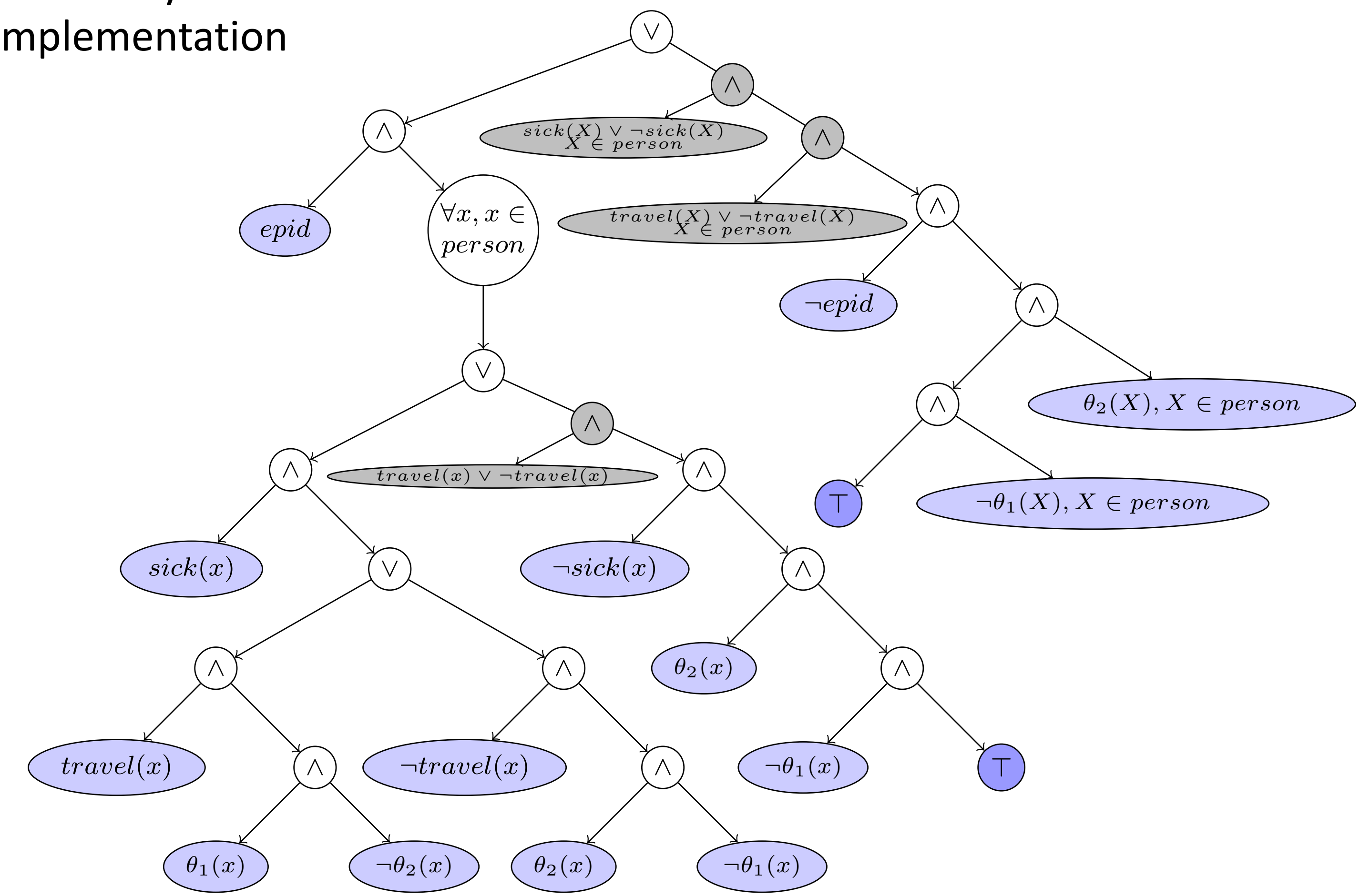
- Given theory in CNF
  1. $\forall X \in \text{People} :$ $\neg\theta_2(X) \lor \neg t(X) \lor \neg s(X) \lor \neg e$
  2. $\forall X \in \text{People} :$ $\theta_1(X) \lor \neg t(X) \lor \neg e \lor \neg s(X)$
  3. $\forall X \in \text{People} : \neg\theta_1(X) \lor t(X)$
  4. $\forall X \in \text{People} : \neg\theta_1(X) \lor e$
  5. $\forall X \in \text{People} : \neg\theta_1(X) \lor s(X)$
  6. $\forall X \in \text{People} : \theta_2(X) \lor t(X)$
  7. $\forall X \in \text{People} : \theta_2(X) \lor e$
  8. $\forall X \in \text{People} : \theta_2(X) \lor s(X)$
- Not smooth since
  - Right branch of root ∨ misses $s(X), t(X)$
  - Right branch of ∨ after set conjunction misses $t(X)$





Example: Smoothed FO d-DNNF Circuit

As generated by the FOKC implementation



Theoretical Results

- Compilation independent of domain sizes
  - Just like construction of FO jtree is also independent of domain sizes
- Inference
  - Polynomial in domain sizes
    - Based on the computations that are computed at different node types
- Completeness as before
  - $\mathcal{M}^{2lv}$
    - Two-logvar theories with max. two logical variables per formula
  - $\mathcal{M}^{1prv}$
    - One logvar per variable

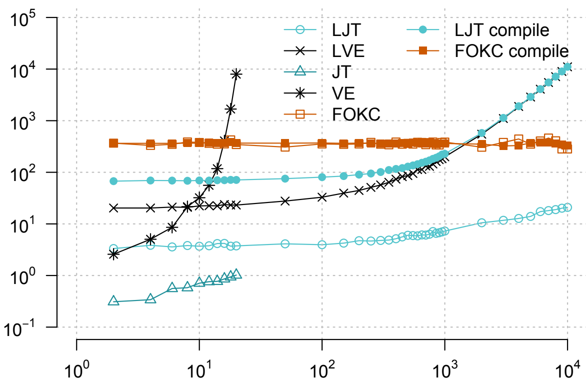
Implementation

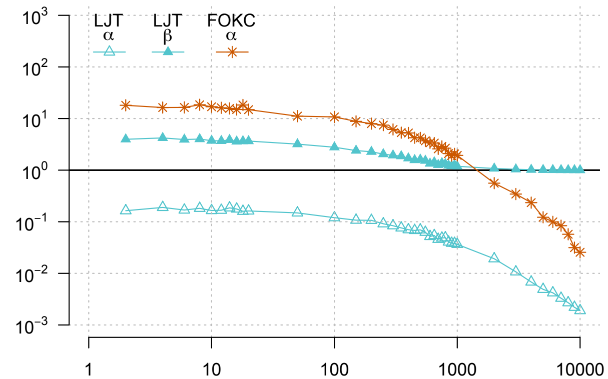
- Available at
  - https://github.com/UCLA-StarAI/Forclift
    - May no longer work according to Guy so you may have to try
      - https://github.com/tanyabraun/wfomc
  - Officially three input formats
    - Based on the normal form required (.wmc)
    - Early version of parfactor graphs (.fg)
    - MLN version (.mln)

    → MLN file format only one I got the compiled version to parse

Implementation

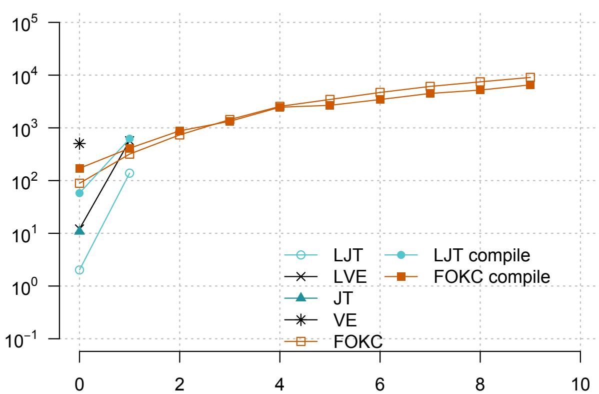
- Query answering times, trade-off criteria
- Increasing domain size

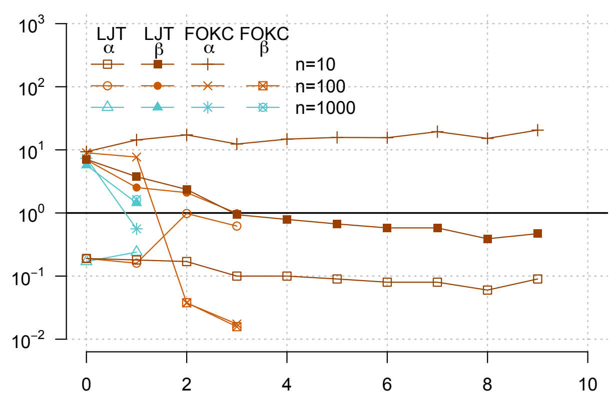




FOKC almost invariant w.r.t. domain sizes

- Increasing counting width





FOKC does not build histograms, which blow up the representation

Runtimes in milliseconds

Probabilistic Theorem Proving (PTP)

- Based on theorem proving in logics
- Solves lifted weighted model counting problem
  - Similar to the weighted first-order model counting problem by Guy Van den Broeck
  - MLNs as input
- Implementation available: Alchemy
  - http://alchemy.cs.washington.edu
  - Input format: MLNs

Vibbhav Gogate and Pedro Domingos: Probabilistic Theorem Proving. In: *UAI-11 Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, 2011.

LJT as a Framework

- Remember: LJT only specifies a helper structure and steps
  - I.e., no specific inference algorithm as a subroutine for its calculations
- Requirements for subroutine
  - Lifted evidence handling
  - Lifted message calculation
    - Message = conj. param'd query
  - Lifted query answering

Calculated lifted?	LVE	FOKC
Evidence	✓	✓
Messages	✓	✗ *
Queries	✓	✓

* Not obvious how parameterised queries are handled in circuits

- LJTKC: LJT with LVE & FOKC
  - LVE for evidence entering and message passing
  - FOKC for query answering
    - Only for Boolean PRVs

Tanya B and Ralf Möller. Fusing First-order Knowledge Compilation and the Lifted Junction Tree Algorithm. In *Proceedings of KI 2018: Advances in Artificial Intelligence*, 2018.

LJTKC: Algorithm

LJTKC$(G, \{Q_i\}_{i=1}^{n}, \{g_e\}_{e=1}^{m})$

Construct an FO jtree J for G

Enter evidence $\{g_e\}_{e=1}^{m}$ into J

Pass message in J

for each parcluster $\boldsymbol{C}_j$ in J **do**

Transform local model G_j into an MLN Ψ_j

Transform Ψ_j into a theory Δ_j in CNF with weight functions w_T, w_F

Build a circuit $\mathcal{C}_j$ for Δ_j

Compute $c_j = WFOMC(\Delta_j, w_T, w_F)$ in $\mathcal{C}_j$

for each query terms Q_i **do**

Build a circuit $\mathcal{C}_{j,q}$ for $\Delta_j \wedge q_i$

Compute $c_q = WFOMC(\Delta_j \wedge q_i, w_T, w_F)$ in $\mathcal{C}_{j,q}$

Return or store $\frac{c_q}{c_j}$ (and possibly $1 - \frac{c_q}{c_j}$)

Summary

- Propositional (weighted) model counting
  - WMC definition
  - Circuits:
    - Inner nodes: conjunctions/disjunctions
    - Leaves: literals, $true$, $false$
    - Properties: d-DNNF, smooth
    - Model counts, WMC by propagation
  - Knowledge compilation
    - Inference in circuits:
      Query answering by weighted model counting in circuits
- Lifted (weighted) model counting
  - WFOMC definition
  - FO circuits
    - Inner nodes can also be set conjunctions/disjunctions
  - First-order knowledge compilation
    - Inference in FO circuits
- Further uses
  - WFOMC in PTP
  - FOKC for query answering in LJT

Outline: 3. Lifted Inference

A. *Lifted variable elimination (LVE)*
- Operators
- Algorithm
- Complexity (including first-order dtrees), completeness, tractability
- Variants

B. *Lifted junction tree algorithm (LJT)*
- First-order junction trees (FO jtrees)
- Algorithm
- Complexity, completeness
- Variants

C. *First-order knowledge compilation (FOKC)*
- Normal form, circuits
- Algorithm
- Complexity, completeness

D. *Beyond Standard Query Answering*
- Adaptive inference
- Changing and unknown domains
- Assignment queries