KU LEUVEN

Stable Model Semantics in ProbLog and its Applications in Argumentation

DTAI Seminar

Pietro Totis
KU Leuven
February 21, 2022

0 Overview

This presentation discusses the following topics:

- How to use Probabilistic Logic Programming (PLP) to model Probabilistic Argumentation problems

0 Overview

This presentation discusses the following topics:

- How to use Probabilistic Logic Programming (PLP) to model Probabilistic Argumentation problems
- How to expand traditional PLP semantics to reason over such models

0 Overview

This presentation discusses the following topics:

- How to use Probabilistic Logic Programming (PLP) to model Probabilistic Argumentation problems
- How to expand traditional PLP semantics to reason over such models
- How to implement a PLP framework for the new semantics

0 Overview

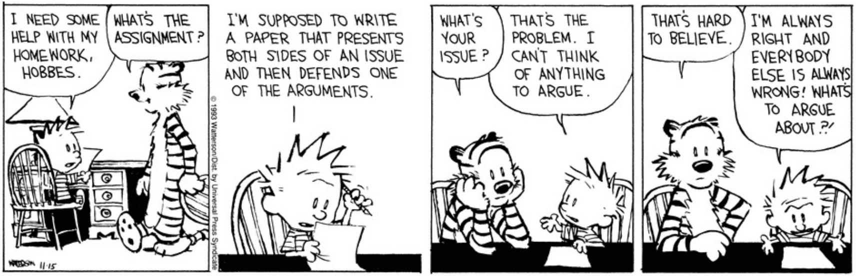
KU LEUVEN

1 Outline

1 What is argumentation?

"Humans argue."[1]

[1] Either you already believe it or you would need to argue against it. Atkinson et al., "Towards Artificial Argumentation", *AI Mag.* 38.3 (2017)



1 What is argumentation?

Argumentation has two main components:

1. Argumentation mining:
   - Find - Sources (newspapers, journals,. . .)
   - Extract - Relevant text (argumentative/non-argumentative)
   - Classify - Arguments (evidence, claims, proponent/opponent . . .)

KU LEUVEN

1 What is argumentation?

Argumentation has two main components:

1. Argumentation mining:
   - Find - Sources (newspapers, journals,. . .)
   - Extract - Relevant text (argumentative/non-argumentative)
   - Classify - Arguments (evidence, claims, proponent/opponent . . .)
2. Abstract reasoning:
   - Model - Roles, structure, strength, . . .
   - Reason - Find accepted/defeated (undecided?) arguments

1 What is argumentation?

1. Argumentation mining:
   - Source - Newspapers, journals,. . .
   - Extract - Relevant text (argumentative/non-argumentative)
   - Classify - Evidence, claims, proponent/opponent. . .
2. Abstract reasoning:
   - Model - Roles, structure, strength, . . .
   - Reason - Find accepted/defeated (undecided?) arguments

We focus on modelling and reasoning. . .

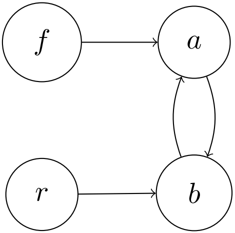
2 Outline

2 Model: Abstract Argumentation Frameworks

Definition

An Abstract Argumentation Framework (AAF) is a pair $(Args, Att)$ where $Args$ is a set of arguments and Att is a binary relation over $Args$: $Att \subseteq Args \times Args$.

We can represent an AAF as a directed graph:



f = "Hotel B is recommended by a friend"
a = "Book hotel A"
r = "Hotel A has higher review score"
b = "Book hotel B"
$Args = \{f, a, r, b\}$
$Att = \{(f, a), (a, b), (b, a), (r, b)\}$

2 Reasoning: Acceptability semantics

Given an AAF, there are different "recipies" to determine acceptable arguments[1]. A set of arguments $A \subseteq Args$ can be:

- Conflict-free (cf.) - when there are no $a, b \in A$ s.t. $(a, b) \in Att$

[1] Baroni, Caminada, and Giacomin, "An introduction to argumentation semantics", *Knowl. Eng. Rev.* 26.4 (2011).

2 Reasoning: Acceptability semantics

Given an AAF, there are different "recipies" to determine acceptable arguments[1]. A set of arguments $A \subseteq Args$ can be:

- Conflict-free (cf.) - when there are no $a, b \in A$ s.t. $(a, b) \in Att$
- Admissible - A is cf. and attacks all arguments attacking A

[1] Baroni, Caminada, and Giacomin, "An introduction to argumentation semantics", *Knowl. Eng. Rev.* 26.4 (2011).

2 Reasoning: Acceptability semantics

Given an AAF, there are different "recipies" to determine acceptable arguments[1]. A set of arguments $A \subseteq Args$ can be:

- Conflict-free (cf.) - when there are no $a, b \in A$ s.t. $(a, b) \in Att$
- Admissible - A is cf. and attacks all arguments attacking A
- Stable - A is cf. and attacks all the arguments not in A

[1] Baroni, Caminada, and Giacomin, "An introduction to argumentation semantics", *Knowl. Eng. Rev.* 26.4 (2011).

2 Reasoning: Acceptability semantics

Given an AAF, there are different "recipies" to determine acceptable arguments[1]. A set of arguments $A \subseteq Args$ can be:

- Conflict-free (cf.) - when there are no $a, b \in A$ s.t. $(a, b) \in Att$
- Admissible - A is cf. and attacks all arguments attacking A
- Stable - A is cf. and attacks all the arguments not in A
- ...

[1] Baroni, Caminada, and Giacomin, "An introduction to argumentation semantics", *Knowl. Eng. Rev.* 26.4 (2011).

2 Reasoning: more expressivity

Each extension to the original AAF requires new ad-hoc semantics to define classes of acceptable arguments in the new framework.

2 Reasoning: more expressivity

Each extension to the original AAF requires new ad-hoc semantics to define classes of acceptable arguments in the new framework.

Combining models

Integrating different formalisms becomes difficult: it requires to redefine the acceptability semantics with new concepts tailored to the specific combination of modelling strategies.

2 Reasoning: more expressivity

Each extension to the original AAF requires new ad-hoc semantics to define classes of acceptable arguments in the new framework.

Combining models

Integrating different formalisms becomes difficult: it requires to redefine the acceptability semantics with new concepts tailored to the specific combination of modelling strategies.

What if we had a general-purpose framework with powerful semantics for encoding many extensions?

3 Outline

3 Modelling: the advantages of Probabilistic Logic Programming

Why Probabilistic Logic Programming

Probabilistic Logic Programming and its tools offer a *general purpose* framework for logical reasoning and uncertainty. This gives some advantages over traditional argumentation frameworks:

3 Modelling: the advantages of Probabilistic Logic Programming

Why Probabilistic Logic Programming

Probabilistic Logic Programming and its tools offer a *general purpose* framework for logical reasoning and uncertainty. This gives some advantages over traditional argumentation frameworks:

- Succinctness and expressivity - express complex interactions between random variables with (first-order) logic rules.

3 Modelling: the advantages of Probabilistic Logic Programming

Why Probabilistic Logic Programming

Probabilistic Logic Programming and its tools offer a *general purpose* framework for logical reasoning and uncertainty. This gives some advantages over traditional argumentation frameworks:

- Succinctness and expressivity - express complex interactions between random variables with (first-order) logic rules.
- Flexibility and modularity - change and evolve a model for specific domains without changing the underlying reasoning algorithms.

3 Modelling: the advantages of Probabilistic Logic Programming

Why Probabilistic Logic Programming

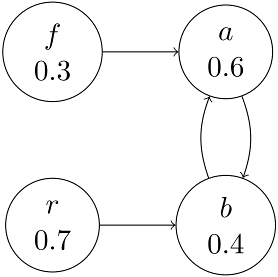
Probabilistic Logic Programming and its tools offer a *general purpose* framework for logical reasoning and uncertainty. This gives some advantages over traditional argumentation frameworks:

- Succinctness and expressivity - express complex interactions between random variables with (first-order) logic rules.
- Flexibility and modularity - change and evolve a model for specific domains without changing the underlying reasoning algorithms.
- PLP tools - bring to argumentation the general suite of PLP inference and learning algorithms.

3 Modelling: What about probabilities?

Definition

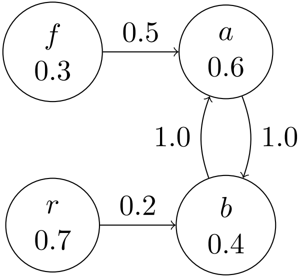
A *probabilistic argument graph* (or probabilistic AAF) is a tuple $(Args, Att, P^*)$ where $(Args, Att)$ is an AAF and P^* is a function: $P^* : Args \rightarrow [0, 1]$.



3 Modelling: What about probabilities?

Definition

A *probabilistic argument graph* (or probabilistic AAF) is a tuple $(Args, Att, P^*)$ where $(Args, Att)$ is an AAF and P^* is a function: $P^* : Args \to [0,1]$. We can also add a function over attacks: $P^\times : Args \times Args \to [0,1]$.



3 Modelling: what do probabilities represent?

Probabilities as *degrees of belief*: they represent how much the argument or relation is believed.

Epistemic interpretation

A probabilistic argument graph describes:

- arguments' prior beliefs (P^*) i.e. a bias.

3 Modelling: what do probabilities represent?

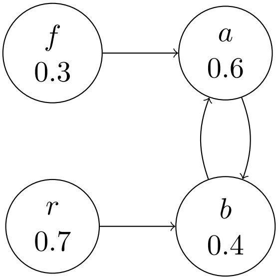
Probabilities as *degrees of belief*: they represent how much the argument or relation is believed.

Epistemic interpretation

A probabilistic argument graph describes:

- arguments' prior beliefs (P^*) i.e. a bias.
- how the initial bias is influenced by the other arguments.

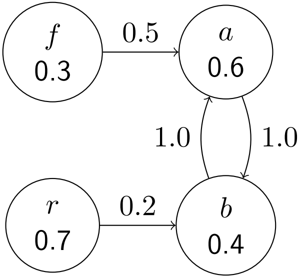
3 Modelling: from *probabilistic* AAFs to ProbLog




```
0.3::f. 0.6::a.
0.7::r. 0.4::b.
\+ a :- f.
\+ b :- r.
\+ b :- a.
\+ a :- b.
```

$$P(f) = 0.3 \ P(r) = 0.7$$
$$P(a) = 0.4 \ P(b) = 0.1$$

3 Modelling: from *gradual* AAFs to ProbLog



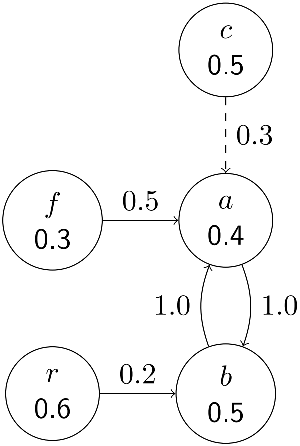

```
0.3::f. 0.6::a.
0.7::r. 0.4::b.
0.5:: \+ a :- f.
0.2:: \+ b :- r.
1.0:: \+ b :- a.
1.0:: \+ a :- b.
```

$$P(f) = 0.3 \; P(r) = 0.7$$
$$P(a) = 0.42 \; P(b) = 0.26$$

KU LEUVEN

3 Modelling: from *bipolar* AAFs to ProbLog

c: "Hotel A has a nice view"

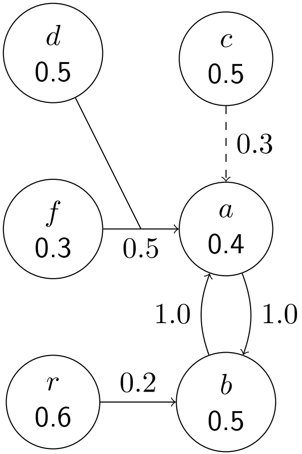



```
0.3::f. 0.6::a.
0.7::r. 0.4::b.
0.5::c.
0.5::\+ a :- f.
1.0::\+ b :- a.
1.0::\+ a :- b.
0.2::\+ b :- r.
0.3::a :- c.
```

$P(f) = 0.3$ $P(r) = 0.7$
$P(a) = 0.47$ $P(b) = 0.25$

3 Modelling: from *set-based* AAFs to ProbLog

d: "Same taste in hotels"




```
0.3::f. 0.6::a.
0.7::r. 0.4::b.
0.5::c. 0.5::d.
0.5::\+ a :- f, d.
1.0::\+ b :- a.
1.0::\+ a :- b.
0.2::\+ b :- r.
0.3::a :- c.
```

$P(f) = 0.3$ $P(r) = 0.7$
$P(a) = 0.5$ $P(b) = 0.24$

4 Outline

4 Probabilistic Logic Programs

Distribution semantics

Each possible choice of all probabilistic facts (total choice) defines a *possible world*: a deterministic program of the chosen facts plus logic rules

4 Probabilistic Logic Programs

Distribution semantics

Each possible choice of all probabilistic facts (total choice) defines a *possible world*: a deterministic program of the chosen facts plus logic rules

Assumption of traditional PLP semantics

All choices are modelled by probabilistic facts: the total choice uniquely determines the truth value of all atoms.

4 Probabilistic Logic Programs

Distribution semantics

Each possible choice of all probabilistic facts (total choice) defines a *possible world*: a deterministic program of the chosen facts plus logic rules

Assumption of traditional PLP semantics

All choices are modelled by probabilistic facts: the total choice uniquely determines the truth value of all atoms.

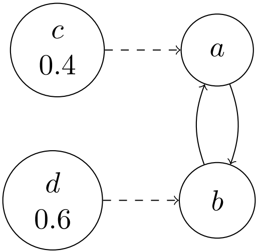
Example

In our hotel example for the *probabilistic* choice where neither the friend nor the reviews influence the choice of the hotel, logic still prescribes a choice between the two options.

4 Probabilistic Logic Programs

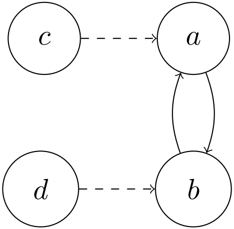
Distribution semantics + Well-founded model semantics

Traditional PLP frameworks cannot reason over programs with cyclic dependencies through negation




```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

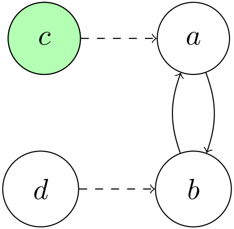
4 Probabilistic Logic Programs




```
c.
d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_1 = \{c, d\}$: $P(\omega_1) = 0.4 \cdot 0.6 = 0.24$

4 Probabilistic Logic Programs



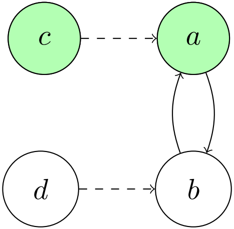

```
c.
d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_1 = \{c, d\}$: $P(\omega_1) = 0.4 \cdot 0.6 = 0.24$

$\textsc{mod}(\omega_1) = \{a, b, c, d\}$

KU LEUVEN

4 Probabilistic Logic Programs

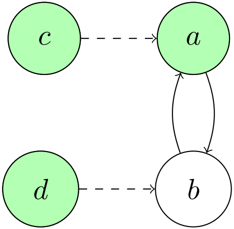



```
c.
d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_1 = \{c, d\}$: $P(\omega_1) = 0.4 \cdot 0.6 = 0.24$

$\text{MOD}(\omega_1) = \{a, b, c, d\}$

4 Probabilistic Logic Programs

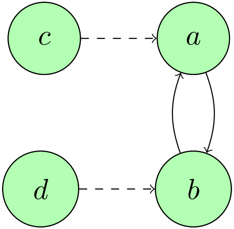



```
c.
d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_1 = \{c, d\}$: $P(\omega_1) = 0.4 \cdot 0.6 = 0.24$

$\text{MOD}(\omega_1) = \{a, b, c, d\}$

4 Probabilistic Logic Programs

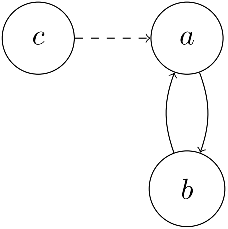



```
c.
d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_1 = \{c, d\}$: $P(\omega_1) = 0.4 \cdot 0.6 = 0.24$

$\text{MOD}(\omega_1) = \{a, b, c, d\}$

4 Probabilistic Logic Programs

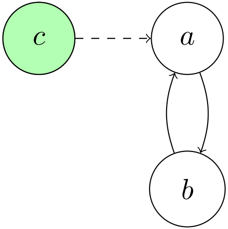



```
c.

a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_2 = \{c\}$: $P(\omega_2) = 0.4 \cdot (1 - 0.6) = 0.16$

4 Probabilistic Logic Programs



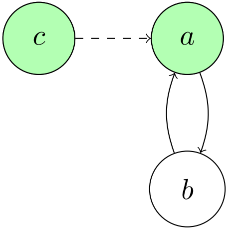

```
c.

a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_2 = \{c\}$: $P(\omega_2) = 0.4 \cdot (1 - 0.6) = 0.16$

$\text{MOD}(\omega_2) = \{a, c\}$

4 Probabilistic Logic Programs



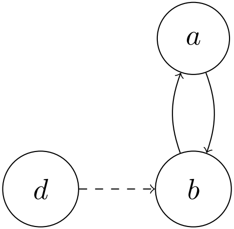

```
c.

a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_2 = \{c\}$: $P(\omega_2) = 0.4 \cdot (1 - 0.6) = 0.16$

$\text{MOD}(\omega_2) = \{a, c\}$

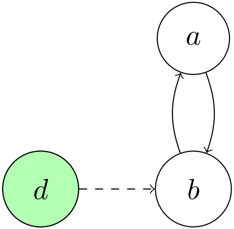
4 Probabilistic Logic Programs




```
d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_3 = \{d\}$: $P(\omega_3) = (1 - 0.4) \cdot 0.6 = 0.36$

4 Probabilistic Logic Programs

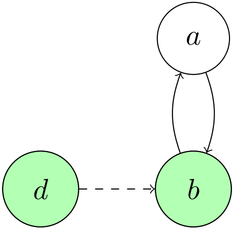



```
d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_3 = \{d\}$: $P(\omega_3) = (1 - 0.4) \cdot 0.6 = 0.36$

$\text{MOD}(\omega_3) = \{b, d\}$

4 Probabilistic Logic Programs

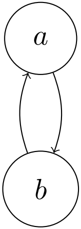



```
d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_3 = \{d\}$: $P(\omega_3) = (1 - 0.4) \cdot 0.6 = 0.36$

$\text{MOD}(\omega_3) = \{b, d\}$

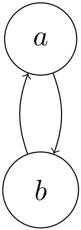
4 Probabilistic Logic Programs




```
  a :- c.
  b :- d.
a :- \+ b.
b :- \+ a.
```

Possible world $\omega_4 = \{\}$: $P(\omega_4) = (1 - 0.4) \cdot (1 - 0.6) = 0.24$

4 Probabilistic Logic Programs



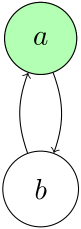

```
  a :- c.
  b :- d.
→ a :- \+ b.
→ b :- \+ a.
```

Possible world $\omega_4 = \{\}$: $P(\omega_4) = (1 - 0.4) \cdot (1 - 0.6) = 0.24$

$\text{MOD}(\omega_4) = ?$

KU LEUVEN

4 Probabilistic Logic Programs




```
    a :- c.
    b :- d.
a :- \+ b.
b :- \+ a.
```

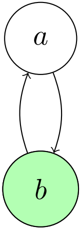
$\top \leftarrow \top$

$\bot \leftarrow \bot$

Possible world $\omega_4 = \{\}$: $P(\omega_4) = (1 - 0.4) \cdot (1 - 0.6) = 0.24$

$\text{MOD}(\omega_4) = \{a\}$

4 Probabilistic Logic Programs




```
      a :- c.
      b :- d.
a :- \+ b.
b :- \+ a.
```

$\bot \leftarrow \bot$

$\top \leftarrow \top$

Possible world $\omega_4 = \{\}$: $P(\omega_4) = (1 - 0.4) \cdot (1 - 0.6) = 0.24$

$\text{MOD}(\omega_4) = \{a\}$ *or* $\{b\}$

4 Probabilistic Logic Programs: well-founded vs stable model semantics

- Valid ProbLog programs are programs where each total choice corresponds to *exactly one* well-founded model.

[2] Totis, Kimmig, and De Raedt, "SMProbLog: Stable Model Semantics in ProbLog and its Applications in Argumentation", *StarAI* abs/2110.01990 (2021).

4 Probabilistic Logic Programs: well-founded vs stable model semantics

- Valid ProbLog programs are programs where each total choice corresponds to *exactly one* well-founded model.
- ω_4 has no well-founded model.

[2] Totis, Kimmig, and De Raedt, "SMProbLog: Stable Model Semantics in ProbLog and its Applications in Argumentation", *StarAI* abs/2110.01990 (2021).

4 Probabilistic Logic Programs: well-founded vs stable model semantics

- Valid ProbLog programs are programs where each total choice corresponds to *exactly one* well-founded model.
- ω_4 has no well-founded model.

SMProbLog programs

SMProbLog[2] reasons over the (zero or more) *stable models* of each total choice.

[2] Totis, Kimmig, and De Raedt, "SMProbLog: Stable Model Semantics in ProbLog and its Applications in Argumentation", *StarAI* abs/2110.01990 (2021).

4 Probabilistic Logic Programs: well-founded vs stable model semantics

- Valid ProbLog programs are programs where each total choice corresponds to *exactly one* well-founded model.
- ω_4 has no well-founded model.

SMProbLog programs

SMProbLog[2] reasons over the (zero or more) *stable models* of each total choice.

If a program has one well-founded model then it is its unique stable model

[2] Totis, Kimmig, and De Raedt, "SMProbLog: Stable Model Semantics in ProbLog and its Applications in Argumentation", *StarAI* abs/2110.01990 (2021).

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$
ω_2	$\{c\}$	0.16	$\{c, a\}$
ω_3	$\{d\}$	0.36	$\{d, b\}$
ω_4	$\{\}$	0.24	$\{a\}, \{b\}$

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$
ω_2	$\{c\}$	0.16	$\{c, a\}$
ω_3	$\{d\}$	0.36	$\{d, b\}$
ω_4	$\{\}$	0.24	$\{a\}, \{b\}$

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

What about the probability of atoms?

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$
ω_2	$\{c\}$	0.16	$\{c, a\}$
ω_3	$\{d\}$	0.36	$\{d, b\}$
ω_4	$\{\}$	0.24	$\{a\}, \{b\}$

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

What about the probability of atoms?

$$P(a) = \sum_{M \models a, M = \text{MOD}(\omega)} P(M)$$

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$
ω_2	$\{c\}$	0.16	$\{c, a\}$
ω_3	$\{d\}$	0.36	$\{d, b\}$
ω_4	$\{\}$	0.24	$\{a\}, \{b\}$

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

What about the probability of atoms?

$$P(a) = \sum_{M \models a, M = \textsc{mod}(\omega)} P(M)$$

We need to define the probability distribution of the models...

4 Reasoning over non-probabilistic choices

- With stable model semantics we have choices that are prescribed by logical consistency rather than probabilistic facts.

4 Reasoning over non-probabilistic choices

- With stable model semantics we have choices that are prescribed by logical consistency rather than probabilistic facts.
- We thus assume that for a fixed possible world all further (logical) choices are equally possible.

KU LEUVEN

4 Reasoning over non-probabilistic choices

- With stable model semantics we have choices that are prescribed by logical consistency rather than probabilistic facts.
- We thus assume that for a fixed possible world all further (logical) choices are equally possible.
- The probability of a model is thus the probability of the possible world normalized w.r.t. the number of non-probabilistic choices.

4 Reasoning over non-probabilistic choices

Let $\mathcal{L}$ be a probabilistic logic program, $\Omega_{\mathcal{L}}$ its set of possible worlds, and $\textsc{mod}(\omega)$ the set of models of a possible world $\omega \in \Omega_{\mathcal{L}}$, then:

4 Reasoning over non-probabilistic choices

Let $\mathcal{L}$ be a probabilistic logic program, $\Omega_{\mathcal{L}}$ its set of possible worlds, and $\textsc{mod}(\omega)$ the set of models of a possible world $\omega \in \Omega_{\mathcal{L}}$, then:

$$P(M) = \frac{P(\omega)}{|\textsc{mod}(\omega)|} \text{ for each } M \in \textsc{mod}(\omega)$$

4 Reasoning over non-probabilistic choices

Let $\mathcal{L}$ be a probabilistic logic program, $\Omega_{\mathcal{L}}$ its set of possible worlds, and $\textsc{mod}(\omega)$ the set of models of a possible world $\omega \in \Omega_{\mathcal{L}}$, then:

$$P(M) = \frac{P(\omega)}{|\textsc{mod}(\omega)|} \text{ for each } M \in \textsc{mod}(\omega)$$

The probability of an atom is then:

4 Reasoning over non-probabilistic choices

Let $\mathcal{L}$ be a probabilistic logic program, $\Omega_{\mathcal{L}}$ its set of possible worlds, and $\textsc{mod}(\omega)$ the set of models of a possible world $\omega \in \Omega_{\mathcal{L}}$, then:

$$P(M) = \frac{P(\omega)}{|\textsc{mod}(\omega)|} \text{ for each } M \in \textsc{mod}(\omega)$$

The probability of an atom is then:

$$P(a) = \sum_{a \in M, M \in \textsc{mod}(\omega), \omega \in \Omega_{\mathcal{L}}} P(M)$$

4 Reasoning over non-probabilistic choices

Let $\mathcal{L}$ be a probabilistic logic program, $\Omega_{\mathcal{L}}$ its set of possible worlds, and $\text{MOD}(\omega)$ the set of models of a possible world $\omega \in \Omega_{\mathcal{L}}$, then:

$$P(M) = \frac{P(\omega)}{|\text{MOD}(\omega)|} \text{ for each } M \in \text{MOD}(\omega)$$

The probability of an atom is then:

$$P(a) = \sum_{a \in M, M \in \text{MOD}(\omega), \omega \in \Omega_{\mathcal{L}}} P(M)$$

We thus solve for a query φ:

4 Reasoning over non-probabilistic choices

Let $\mathcal{L}$ be a probabilistic logic program, $\Omega_{\mathcal{L}}$ its set of possible worlds, and $\textsc{mod}(\omega)$ the set of models of a possible world $\omega \in \Omega_{\mathcal{L}}$, then:

$$P(M) = \frac{P(\omega)}{|\textsc{mod}(\omega)|} \text{ for each } M \in \textsc{mod}(\omega)$$

The probability of an atom is then:

$$P(a) = \sum_{a \in M, M \in \textsc{mod}(\omega), \omega \in \Omega_{\mathcal{L}}} P(M)$$

We thus solve for a query φ:

$$\widehat{WMC}_{\mathcal{L}}(\varphi) = \sum_{M \models \varphi, M \in \textsc{mod}(\omega), \omega \in \Omega_{\mathcal{L}}} \frac{1}{|\textsc{mod}(\omega)|} \cdot \prod_{l \in M} w(l)$$

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model	$P(M)$
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$	0.24
ω_2	$\{c\}$	0.16	$\{c, a\}$	0.16
ω_3	$\{d\}$	0.36	$\{d, b\}$	0.36
ω_4	$\{\}$	0.24	$\{a\}$, $\{b\}$	0.12, 0.12

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model	$P(M)$
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$	0.24
ω_2	$\{c\}$	0.16	$\{c, a\}$	0.16
ω_3	$\{d\}$	0.36	$\{d, b\}$	0.36
ω_4	$\{\}$	0.24	$\{a\}$, $\{b\}$	0.12, 0.12

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

$$P(c) = 0.24 + 0.16 = 0.4$$

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model	$P(M)$
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$	0.24
ω_2	$\{c\}$	0.16	$\{c, a\}$	0.16
ω_3	$\{d\}$	0.36	$\{d, b\}$	0.36
ω_4	$\{\}$	0.24	$\{a\}$, $\{b\}$	0.12, 0.12

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

$P(c) = 0.24 + 0.16 = 0.4$
$P(d) = 0.24 + 0.36 = 0.6$

KU LEUVEN

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model	$P(M)$
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$	0.24
ω_2	$\{c\}$	0.16	$\{c, a\}$	0.16
ω_3	$\{d\}$	0.36	$\{d, b\}$	0.36
ω_4	$\{\}$	0.24	$\{a\}$, $\{b\}$	0.12, 0.12

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

$P(c) = 0.24 + 0.16 = 0.4$
$P(d) = 0.24 + 0.36 = 0.6$
$P(a) = 0.24 + 0.16 + 0.12 = 0.52$

4 Probabilistic Logic Programs: success probability

P.W.	Facts	Probability	Model	$P(M)$
ω_1	$\{c, d\}$	0.24	$\{a, b, c, d\}$	0.24
ω_2	$\{c\}$	0.16	$\{c, a\}$	0.16
ω_3	$\{d\}$	0.36	$\{d, b\}$	0.36
ω_4	$\{\}$	0.24	$\{a\}$, $\{b\}$	0.12, 0.12

```
0.4::c.
0.6::d.
a :- c.
b :- d.
a :- \+ b.
b :- \+ a.
```

$$P(c) = 0.24 + 0.16 = 0.4$$
$$P(d) = 0.24 + 0.36 = 0.6$$
$$P(a) = 0.24 + 0.16 + 0.12 = 0.52$$
$$P(b) = 0.24 + 0.36 + 0.12 = 0.72$$

4 Probabilistic Logic Programs: inconsistencies

An even (odd, respectively) cycle is a simple cycle with a non-zero even (odd, respectively) number of negative edges.

Even/odd cycles through negations

4 Probabilistic Logic Programs: inconsistencies

An even (odd, respectively) cycle is a simple cycle with a non-zero even (odd, respectively) number of negative edges.

Even/odd cycles through negations

- if a program has no even-length cycle, then it has at most one stable model

4 Probabilistic Logic Programs: inconsistencies

An even (odd, respectively) cycle is a simple cycle with a non-zero even (odd, respectively) number of negative edges.

Even/odd cycles through negations

- if a program has no even-length cycle, then it has at most one stable model
- if a program has no odd-length cycle (*call-consistent*), then it has at least one stable model

4 Probabilistic Logic Programs: inconsistencies

An even (odd, respectively) cycle is a simple cycle with a non-zero even (odd, respectively) number of negative edges.

Even/odd cycles through negations

- if a program has no even-length cycle, then it has at most one stable model
- if a program has no odd-length cycle (*call-consistent*), then it has at least one stable model

  Lin and Zhao, "On Odd and Even Cycles in Normal Logic Programs" (2004)

4 Probabilistic Logic Programs: inconsistencies

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

4 Probabilistic Logic Programs: inconsistencies

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

- If `c` or `d` are chosen, `a` is true and the possible world has *0 stable models*

4 Probabilistic Logic Programs: inconsistencies

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

- If `c` or `d` are chosen, `a` is true and the possible world has *0 stable models*
- Therefore logic states that some total choices are *inconsistent* with the domain knowledge.

4 Probabilistic Logic Programs: inconsistencies

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

- If `c` or `d` are chosen, `a` is true and the possible world has *0 stable models*
- Therefore logic states that some total choices are *inconsistent* with the domain knowledge.

We use a *three-valued interpretation* to keep track of the probability of inconsistent choices.

4 Probabilistic Logic Programs: inconsistencies

Three-valued interpretaiton

A three-valued interpretation is a pair (T, F) where T is the set of atoms interpreted *true* and F is the set of atoms interpreted *false*. The atoms outside $T \cup F$ are interpreted as *inconsistent*.

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

4 Probabilistic Logic Programs: inconsistencies

Three-valued interpretaiton

A three-valued interpretation is a pair (T, F) where T is the set of atoms interpreted *true* and F is the set of atoms interpreted *false*. The atoms outside $T \cup F$ are interpreted as *inconsistent*.

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

P.W.	Facts	Probability	Model
ω_1	$\{c, d\}$	0.24	$\{(\emptyset, \emptyset)\}$
ω_2	$\{c\}$	0.16	$\{(\emptyset, \emptyset)\}$
ω_3	$\{d\}$	0.36	$\{(\emptyset, \emptyset)\}$
ω_4	$\{\}$	0.24	$\{(\{e\}, \{c, d, a, b\})\}$

4 Probabilistic Logic Programs: inconsistencies

Three-valued interpretaiton

A three-valued interpretation is a pair (T, F) where T is the set of atoms interpreted *true* and F is the set of atoms interpreted *false*. The atoms outside $T \cup F$ are interpreted as *inconsistent*.

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

P.W.	Facts	Probability	Model
ω_1	$\{c, d\}$	0.24	$\{(\emptyset, \emptyset)\}$
ω_2	$\{c\}$	0.16	$\{(\emptyset, \emptyset)\}$
ω_3	$\{d\}$	0.36	$\{(\emptyset, \emptyset)\}$
ω_4	$\{\}$	0.24	$\{(\{e\}, \{c, d, a, b\})\}$

$$P(\mathcal{L} \models \bot) = 0.24 + 0.16 + 0.36 = 0.76$$

4 Probabilistic Logic Programs: inconsistencies

Three-valued interpretaiton

A three-valued interpretation is a pair (T, F) where T is the set of atoms interpreted *true* and F is the set of atoms interpreted *false*. The atoms outside $T \cup F$ are interpreted as *inconsistent*.

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

P.W.	Facts	Probability	Model
ω_1	$\{c, d\}$	0.24	$\{(\emptyset, \emptyset)\}$
ω_2	$\{c\}$	0.16	$\{(\emptyset, \emptyset)\}$
ω_3	$\{d\}$	0.36	$\{(\emptyset, \emptyset)\}$
ω_4	$\{\}$	0.24	$\{(\{e\}, \{c, d, a, b\})\}$

$$P(\mathcal{L} \models \bot) = 0.24 + 0.16 + 0.36 = 0.76$$
$$P(e) = P(\neg b) = 0.24$$

4 Probabilistic Logic Programs: inconsistencies

Three-valued interpretaiton

A three-valued interpretation is a pair (T, F) where T is the set of atoms interpreted *true* and F is the set of atoms interpreted *false*. The atoms outside $T \cup F$ are interpreted as *inconsistent*.

```
0.4::c.
0.6::d.
a :- c.
a :- d.
b :- a, \+b.
e :- \+a.
```

P.W.	Facts	Probability	Model
ω_1	$\{c, d\}$	0.24	$\{(\emptyset, \emptyset)\}$
ω_2	$\{c\}$	0.16	$\{(\emptyset, \emptyset)\}$
ω_3	$\{d\}$	0.36	$\{(\emptyset, \emptyset)\}$
ω_4	$\{\}$	0.24	$\{(\{e\}, \{c, d, a, b\})\}$

$$P(\mathcal{L} \models \bot) = 0.24 + 0.16 + 0.36 = 0.76$$
$$P(e) = P(\neg b) = 0.24$$
$$P(\neg e) = P(b) = 0$$

KU LEUVEN

5 Outline

KU LEUVEN

5 Pipelines

ProbLog

Program → Grounding → [Cycle breaking → CNF] → Knowledge compilation → Circuit → WMC

SMProbLog

Program → Grounding → Knowledge compilation → Circuit → [Stable models → $\widehat{WMC}$]

5 Pipelines: differences

- Cycle breaking and CNF conversion do not preserve stable models

[3] Aziz et al., "Stable Model Counting and Its Application in Probabilistic Logic Programming" (2015).

5 Pipelines: differences

- Cycle breaking and CNF conversion do not preserve stable models
- We need a knowledge compiler for stable models: DSHARP[3]

[3] Aziz et al., "Stable Model Counting and Its Application in Probabilistic Logic Programming" (2015).

5 Pipelines: differences

- Cycle breaking and CNF conversion do not preserve stable models
- We need a knowledge compiler for stable models: DSHARP[3]
- We need then to count the stable models to solve the $\widehat{WMC}$ problem

[3] Aziz et al., "Stable Model Counting and Its Application in Probabilistic Logic Programming" (2015).

KU LEUVEN

5 Pipelines: WMC

Definition

A *NNF* is a rooted directed acyclic graph in which each leaf node is labeled with a literal and each internal node is labeled with a disjunction or conjunction. A smooth *d-DNNF* is an NNF with the following properties:

- Deterministic: for all disjunctive nodes the children represent formulas pairwise inconsistent.
- Decomposable: the subtrees rooted in two children of a conjunction node do not have atoms in common.
- Smooth: all children of a disjunction node use the same set of atoms.

5 Pipelines: WMC

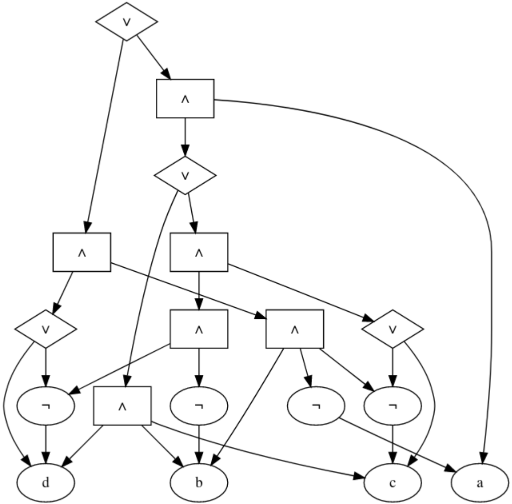
Definition

A *NNF* is a rooted directed acyclic graph in which each leaf node is labeled with a literal and each internal node is labeled with a disjunction or conjunction. A smooth *d-DNNF* is an NNF with the following properties:

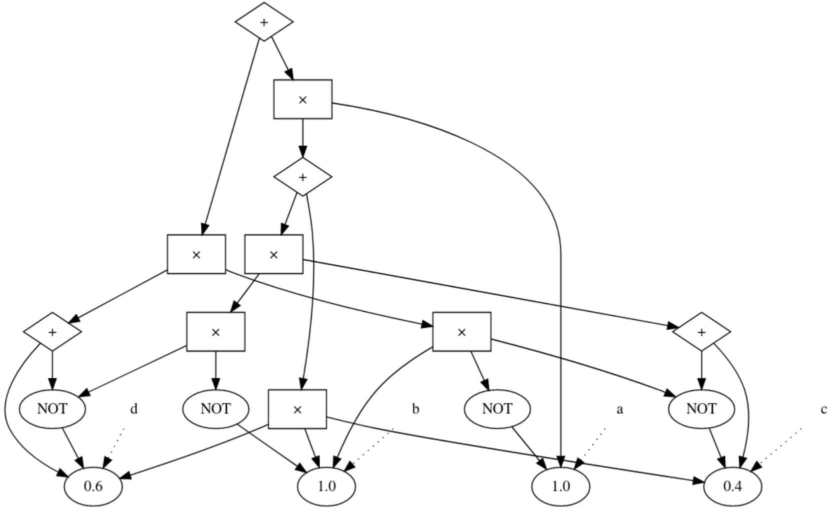
- Deterministic: for all disjunctive nodes the children represent formulas pairwise inconsistent.
- Decomposable: the subtrees rooted in two children of a conjunction node do not have atoms in common.
- Smooth: all children of a disjunction node use the same set of atoms.

WMC can be solved in polynomial time on d-DNNFs

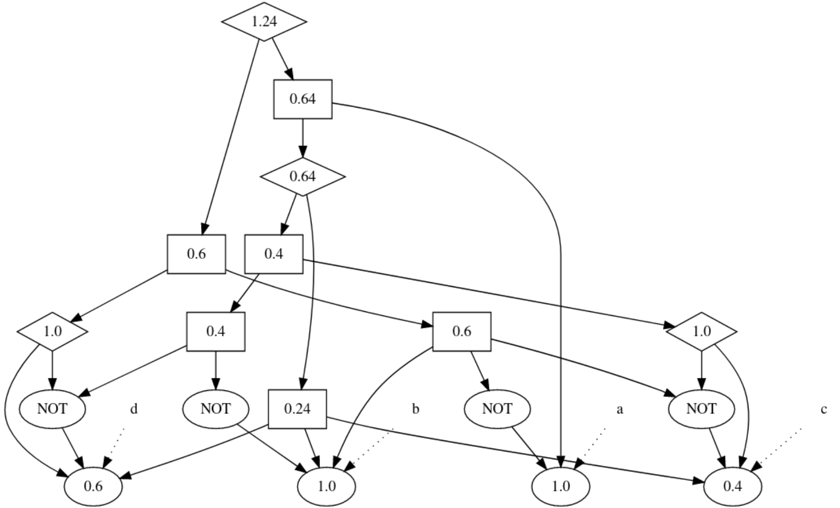
5 Pipelines: WMC



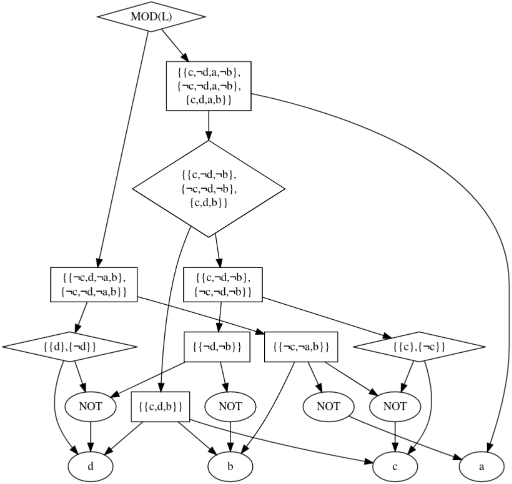
5 Pipelines: WMC



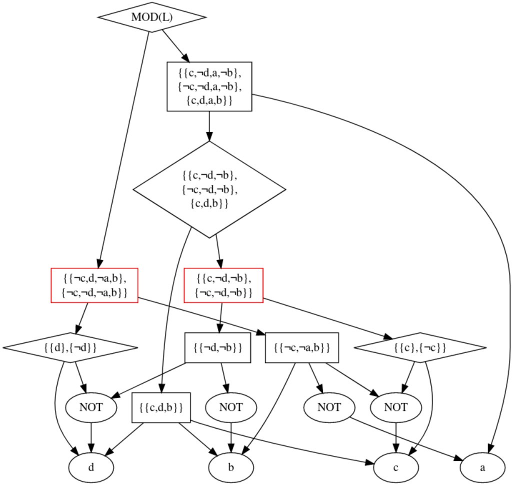
5 Pipelines: WMC



5 Pipelines: WMC



5 Pipelines: WMC



5 Pipelines: from WMC to $\widehat{WMC}$

We are solving two tasks in one circuit[4]:

[4]Kiesel, Totis, and Kimmig, "Efficient Knowledge Compilation Beyond Weighted Model Counting", *(Under review)* (2022).

KU LEUVEN

5 Pipelines: from WMC to $\widehat{WMC}$

We are solving two tasks in one circuit[4]:

- Computing the weight of the total choices

[4] Kiesel, Totis, and Kimmig, "Efficient Knowledge Compilation Beyond Weighted Model Counting", *(Under review)* (2022).

5 Pipelines: from WMC to $\widehat{WMC}$

We are solving two tasks in one circuit[4]:

- Computing the weight of the total choices
- Computing the number of stable models for each total choice

[4] Kiesel, Totis, and Kimmig, "Efficient Knowledge Compilation Beyond Weighted Model Counting", *(Under review)* (2022).

5 Pipelines: from WMC to $\widehat{WMC}$

We are solving two tasks in one circuit[4]:

- Computing the weight of the total choices
- Computing the number of stable models for each total choice

Constrained KC

Constrained Knowledge Compilation allows us to solve the two problems in polynomial time by constraining the order in which the variables are decided.

[4] Kiesel, Totis, and Kimmig, "Efficient Knowledge Compilation Beyond Weighted Model Counting", *(Under review)* (2022).

KU LEUVEN

5 Pipelines: from WMC to $\widehat{WMC}$

We are solving two tasks in one circuit[4]:

- Computing the weight of the total choices
- Computing the number of stable models for each total choice

Constrained KC

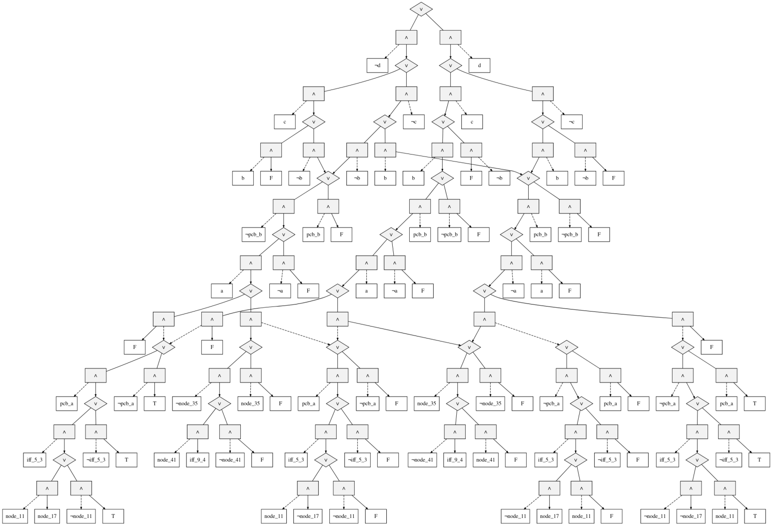
Constrained Knowledge Compilation allows us to solve the two problems in polynomial time by constraining the order in which the variables are decided.

Trade-off

Constrained variable orders typically lead to an increase of the size of the circuit.

[4] Kiesel, Totis, and Kimmig, "Efficient Knowledge Compilation Beyond Weighted Model Counting", *(Under review)* (2022).

5 Pipelines: constrained compilation



KU LEUVEN

6 Outline

KU LEUVEN

6 Summary

- Argumentation frameworks come in many flavors and interpretations: we combine with PLP the most important ones, unlocking additional resources for inference and learning in argumentation.

6 Summary

- Argumentation frameworks come in many flavors and interpretations: we combine with PLP the most important ones, unlocking additional resources for inference and learning in argumentation.
- Traditional PLP semantics do not support common logical patterns in argument graphs: we introduce more expressive semantics allowing us reason on such programs.

6 Summary

- Argumentation frameworks come in many flavors and interpretations: we combine with PLP the most important ones, unlocking additional resources for inference and learning in argumentation.
- Traditional PLP semantics do not support common logical patterns in argument graphs: we introduce more expressive semantics allowing us reason on such programs.
- New semantics require new inference methods: we resort to pipelines/compilers for stable model semantics.

6 Summary

- Argumentation frameworks come in many flavors and interpretations: we combine with PLP the most important ones, unlocking additional resources for inference and learning in argumentation.
- Traditional PLP semantics do not support common logical patterns in argument graphs: we introduce more expressive semantics allowing us reason on such programs.
- New semantics require new inference methods: we resort to pipelines/compilers for stable model semantics.
- Not all circuits that allow efficient inference for WMC are suitable for $\widehat{WMC}$: we constrain the variable order to be able to solve efficiently two tasks in one circuit.

Questions?