Splitting Epistemic Logic Programs

Pedro Cabalar[1], Jorge Fandinno[2] and Luis Fariñas del Cerro[2]

[1] University of Corunna, Spain
[2] IRIT, Toulouse, France

LPNMR'19, June 5th, 2019
Philadelphia, PA, USA

Epistemic Logic Programs

- Epistemic Specifications [Gelfond91] (aka Epistemic Logic Programs) extend logic programs, under answer set semantics, with subjective literals.

Epistemic Logic Programs

- Epistemic Specifications [Gelfond91] (aka Epistemic Logic Programs) extend logic programs, under answer set semantics, with subjective literals.
- A subjective literal is like a query about some literal ℓ like p or $\texttt{not}\ p$

Epistemic Logic Programs

- Epistemic Specifications [Gelfond91] (aka Epistemic Logic Programs) extend logic programs, under answer set semantics, with subjective literals.

- A subjective literal is like a query about some literal ℓ like p or $\texttt{not}\ p$

  $\mathbf{K}\,\ell$ means ℓ holds in all stable models (cautions consequence)

  $\mathbf{M}\,\ell$ means ℓ holds in some stable model (brave consequence)

Epistemic Logic Programs

Example

Program Π = Hamiltonian paths

$in(X, Y)$ = edge (X, Y) in the path

$\mathbf{K}\, in(1, 2)$ = all Hamiltonian paths contain edge $(1, 2)$

$\mathbf{M}\, in(1, 2)$ = some Hamiltonian path contains edge $(1, 2)$

Epistemic Logic Programs

Example

Program Π = Hamiltonian paths
$in(X, Y)$ = edge (X, Y) in the path

$\mathbf{K}\, in(1, 2)$ = all Hamiltonian paths contain edge $(1, 2)$

$\mathbf{M}\, in(1, 2)$ = some Hamiltonian path contains edge $(1, 2)$

More than queries: we can use them to derive new information

$$critical(X, Y) \leftarrow \mathbf{K}\, in(X, Y)$$
$$irrelevant(X, Y) \leftarrow edge(X, Y), \mathtt{not}\ \mathbf{M}\, in(X, Y)$$

Epistemic Logic Programs

- The idea looks simple! So, why is it so hard to characterize?

Epistemic Logic Programs

The idea looks simple! So, why is it so hard to characterize?

- **K** and **M** may affect the answer sets they are meant to quantify. How to solve cyclic specifications?

$$a \leftarrow \texttt{not}\ \mathbf{K}\, b \qquad\qquad b \leftarrow \texttt{not}\ \mathbf{K}\, a$$

Epistemic Logic Programs

👍 The idea looks simple! So, why is it so hard to characterize?

- **K** and **M** may affect the answer sets they are meant to quantify. How to solve cyclic specifications?

$$a \leftarrow \texttt{not}\ \mathbf{K}\, b \qquad\qquad b \leftarrow \texttt{not}\ \mathbf{K}\, a$$

[Gelfond 1991]: alternative world views (sets of answer sets). Above, we get these two: $[\ \{a\}\ ]$ and $[\ \{b\}\ ]$.

Epistemic Logic Programs

👍 The idea looks simple! So, why is it so hard to characterize?

- **K** and **M** may affect the answer sets they are meant to quantify. How to solve cyclic specifications?

  $$a \leftarrow \texttt{not}\ \mathbf{K}\, b \qquad\qquad b \leftarrow \texttt{not}\ \mathbf{K}\, a$$

  [Gelfond 1991]: alternative world views (sets of answer sets). Above, we get these two: $[\ \{a\}\ ]$ and $[\ \{b\}\ ]$.

- Problem with self-supportedness [Truszczyński 2011]

  $a \leftarrow \mathbf{K}\, a$ has 2 world views,
  $[\ \emptyset\ ]$ (expected) and $[\ \{a\}\ ]$ (unfounded)

  (☞ see talk about foundedness tomorrow 🕘 16:30)

Epistemic Logic Programs: Literature

Later approaches try to overcome this original self-supportedness:

- Gelfond 2011, "New Semantics for Epistemic Specifications"
- P. T. Kahl 2014, "Refining the semantics for epistemic logic programming"
- P. Kahl et al. 2015, "The language of epistemic specifications (refined) including a prototype solver"
- Fariñas del Cerro, Herzig, and Su 2015, "Epistemic Equilibrium Logic"
- Shen and Eiter 2017, "Evaluating Epistemic Negation in Answer Set Programming (Extended Abstract)"
- Son et al. 2017, "On Computing World Views of Epistemic Logic Programs"

Methodology

- Current methodology: testing intuition on a set of example programs
  Example Π + Semantics $X \Rightarrow$ world views ⚖ intuitive?

Methodology

- Current methodology: testing intuition on a set of example programs
  Example Π + Semantics $X \Rightarrow$ world views ⚖ intuitive?
- Next step: we propose defining formal properties. Advantages:
  - they help to predict results for families of examples
  - intuitions are formalized: some intuitions may be inconsistent
  - they make comparison easier
- Knowing that semantics X does not satisfy a property is also valuable

In this paper . . .

In this paper, we introduce the property of epistemic splitting.

- Inspired by the splitting theorem for regular logic programs [Lifschitz & Turner 1994],
- Some programs Π can be splitted in two parts $\Pi_b \cup \Pi_t$:
  bottom Π_b produces world views to be queried
  top Π_t gets conclusions from queries on the bottom

In this paper . . .

In this paper, we introduce the property of epistemic splitting.

- Inspired by the splitting theorem for regular logic programs [Lifschitz & Turner 1994],
- Some programs Π can be splitted in two parts $\Pi_b \cup \Pi_t$:
  bottom Π_b produces world views to be queried
  top Π_t gets conclusions from queries on the bottom

☞ Keypoint: when this happens, the semantics of Π should be definable in terms of the semantics of Π_b and Π_t.

An example

Note $\sim p$ is treated as an atom plus $\bot \leftarrow p, \sim p$

Example (Gelfond 1991)

College rules to decide whether a student X is eligible for a scholarship:

$$\begin{array}{rcl} eligible(X) & \leftarrow & high(X) \\ eligible(X) & \leftarrow & minority(X), fair(X) \\ \sim eligible(X) & \leftarrow & \sim fair(X), \sim high(X) \end{array}$$

An example

Note $\sim p$ is treated as an atom plus $\bot \leftarrow p, \sim p$

Example (Gelfond 1991)

College rules to decide whether a student X is eligible for a scholarship:

$$\begin{aligned} \mathit{eligible}(X) &\leftarrow \mathit{high}(X) \\ \mathit{eligible}(X) &\leftarrow \mathit{minority}(X), \mathit{fair}(X) \\ \sim\mathit{eligible}(X) &\leftarrow \sim\mathit{fair}(X), \sim\mathit{high}(X) \end{aligned}$$

Additional college criterion:

> *"The students whose eligibility is not determined by the college rules should be interviewed by the scholarship committee."* □

An example

Note $\sim p$ is treated as an atom plus $\bot \leftarrow p, \sim p$

Example (Gelfond 1991)

College rules to decide whether a student X is eligible for a scholarship:

$$\begin{array}{rcl} eligible(X) & \leftarrow & high(X) \\ eligible(X) & \leftarrow & minority(X), fair(X) \\ \sim eligible(X) & \leftarrow & \sim fair(X), \sim high(X) \end{array}$$

Additional college criterion:

> *"The students whose eligibility is not determined by the college rules should be interviewed by the scholarship committee."* □

For student *mike* we just know: $high(mike) \vee fair(mike)$

An example

Note $\sim p$ is treated as an atom plus $\bot \leftarrow p, \sim p$

Example (Gelfond 1991)

College rules to decide whether a student X is eligible for a scholarship:

$$\begin{array}{rcl} eligible(X) & \leftarrow & high(X) \\ eligible(X) & \leftarrow & minority(X), fair(X) \\ \sim eligible(X) & \leftarrow & \sim fair(X), \sim high(X) \end{array}$$

Additional college criterion:

> *"The students whose eligibility is not determined by the college rules should be interviewed by the scholarship committee."* □

For student *mike* we just know: $high(mike) \vee fair(mike)$

Two stable models $\{high(mike), eligible(mike)\}$, $\{fair(mike)\}$

An example

Two stable models $\{high(mike), eligible(mike)\}$, $\{fair(mike)\}$

- Can we determine $eligible(mike)$? (cautious consequence).

An example

Two stable models $\{high(mike), eligible(mike)\}$, $\{fair(mike)\}$

- Can we determine $eligible(mike)$? (cautious consequence).
  Adding constraint $\bot \leftarrow eligible(mike)$ tells us "no"

An example

Two stable models $\{high(mike), eligible(mike)\}$, $\{fair(mike)\}$

- Can we determine $eligible(mike)$? (cautious consequence). Adding constraint $\bot \leftarrow eligible(mike)$ tells us "no"
- The same for $\sim eligible(mike)$, so an interview should follow. We can represent this using an epistemic rule:

$$interview(X) \leftarrow \mathtt{not}\, \mathbf{K}\, eligible(X),\ \mathtt{not}\, \mathbf{K}\, {\sim} eligible(X)$$

An example

Two stable models $\{high(mike), eligible(mike)\}$, $\{fair(mike)\}$

- Can we determine $eligible(mike)$? (cautious consequence). Adding constraint $\bot \leftarrow eligible(mike)$ tells us "no"
- The same for $\sim eligible(mike)$, so an interview should follow. We can represent this using an epistemic rule:

$$interview(X) \leftarrow \mathtt{not}\, \mathbf{K}\, eligible(X),\ \mathtt{not}\, \mathbf{K} \sim eligible(X)$$

The rule for $interview(X)$ uses a query on the rest of the program

Syntax

- Given regular literal ℓ, a subjective literal L can be:
  $L ::= \ell \mid \mathbf{K}\,\ell \mid \mathbf{M}\,\ell \mid \mathtt{not}\ \mathbf{K}\,\ell \mid \mathtt{not}\ \mathbf{M}\,\ell$

Syntax

- Given regular literal ℓ, a subjective literal L can be:
  $L ::= \ell \mid \mathbf{K}\,\ell \mid \mathbf{M}\,\ell \mid \texttt{not}\ \mathbf{K}\,\ell \mid \texttt{not}\ \mathbf{M}\,\ell$
- A rule has the form:

$$\underbrace{p_1 \vee \cdots \vee p_n}_{\text{head: atoms}} \leftarrow \underbrace{L_1\ ,\ \ldots\ ,\ L_m}_{\text{body: subjective lits.}}$$

- When $n = 0$, the head becomes $\bot$ (a constraint).

Syntax

- Given regular literal ℓ, a subjective literal L can be:
  $L ::= \ell \mid \mathbf{K}\,\ell \mid \mathbf{M}\,\ell \mid \mathtt{not}\ \mathbf{K}\,\ell \mid \mathtt{not}\ \mathbf{M}\,\ell$
- A rule has the form:

$$\underbrace{p_1 \vee \cdots \vee p_n}_{\text{head: atoms}} \leftarrow \underbrace{L_1\ ,\ \ldots\ ,\ L_m}_{\text{body: subjective lits.}}$$

- When $n = 0$, the head becomes $\bot$ (a constraint).
  The constraint is subjective if all L_i are subjective

(Generic) semantics: some properties

Definition (Semantics)

A semantics is a function that maps each program (set of rules) to a set of world views.

(Generic) semantics: some properties

Definition (Semantics)

A semantics is a function that maps each program (set of rules) to a set of world views.

The following two properties are satisfied by all semantics in the literature:

Property (Supra-ASP)

A semantics satisfies supra-ASP iff any program Π *with no subjective literal has a unique world view collecting the answer sets of* Π. □

(Generic) semantics: some properties

Definition (Semantics)

A semantics is a function that maps each program (set of rules) to a set of world views.

The following two properties are satisfied by all semantics in the literature:

Property (Supra-ASP)

A semantics satisfies supra-ASP iff any program Π with no subjective literal has a unique world view collecting the answer sets of Π. □

Property (Supra-S5)

A semantics satisfies supra-S5 iff every world view of Π is also a model of Π in the modal logic S5. □

Epistemic Splitting

Property (Epistemic splitting, Informally)

A semantics satisfies epistemic splitting if, for every program that can be divided in two parts, bottom and top, such that

- *the bottom does not refer to atoms in the top, and*
- *the top may only refer bottom atoms through subjective literals.*

its world views can be computed as a combination of the world views of the bottom and the top. □

Epistemic Splitting

— bottom part —

$$eligible(X) \leftarrow high(X) \quad (1)$$

$$eligible(X) \leftarrow minority(X), fair(X) \quad (2)$$

$$\sim eligible(X) \leftarrow \sim fair(X), \sim high(X) \quad (3)$$

$$fair(mike) \vee high(mike) \quad (4)$$

— top part —

$$interview(X) \leftarrow \texttt{not}\,\mathbf{K}\,eligible(X),\ \texttt{not}\,\mathbf{K}\sim eligible(X) \quad (5)$$

(1)-(4) do not refer to *interview*(X)
(5) only refers to atoms in (1)-(4) through subjective literals.

Epistemic Splitting

— bottom part ——————————————

$$
\begin{aligned}
eligible(X) &\leftarrow high(X) \\
eligible(X) &\leftarrow minority(X), fair(X) \\
\sim eligible(X) &\leftarrow \sim fair(X), \sim high(X) \\
fair(mike) &\vee high(mike)
\end{aligned}
$$

No subjective literals: we get two answer sets

$$M_1 = \{fair(mike)\} \qquad M_2 = \{high(mike), eligible(mike)\}$$

Epistemic Splitting

— bottom part

$$eligible(X) \leftarrow high(X)$$
$$eligible(X) \leftarrow minority(X), fair(X)$$
$$\sim eligible(X) \leftarrow \sim fair(X), \sim high(X)$$
$$fair(mike) \vee high(mike)$$

No subjective literals: we get two answer sets

$$M_1 = \{fair(mike)\} \qquad M_2 = \{high(mike), eligible(mike)\}$$

Supra-ASP implies unique world view $W = [\,M_1, M_2\,]$

Epistemic Splitting

— bottom part ——————————

$$\begin{array}{rcl} \mathit{eligible}(X) & \leftarrow & \mathit{high}(X) \\ \mathit{eligible}(X) & \leftarrow & \mathit{minority}(X), \mathit{fair}(X) \\ \sim\mathit{eligible}(X) & \leftarrow & \sim\mathit{fair}(X), \sim\mathit{high}(X) \end{array}$$

$$\mathit{fair}(\mathit{mike}) \vee \mathit{high}(\mathit{mike})$$

No subjective literals: we get two answer sets

$$M_1 = \{\mathit{fair}(\mathit{mike})\} \qquad M_2 = \{\mathit{high}(\mathit{mike}), \mathit{eligible}(\mathit{mike})\}$$

Supra-ASP implies unique world view $W = [\, M_1, M_2 \,]$

In W, $\mathbf{K}\, \mathit{eligible}(\mathit{mike})$ and $\mathbf{K} \sim\mathit{eligible}(\mathit{mike})$ are false!

Epistemic Splitting

We can simplify the top with respect to W

— top part

$$interview(mike) \leftarrow \texttt{not}\,\mathbf{K}\,eligible(mike),\ \texttt{not}\,\mathbf{K}\,{\sim}eligible(mike)$$

Epistemic Splitting

We can simplify the top with respect to W

— top part —

$interview(mike) \leftarrow \texttt{not}\,\bot \quad , \texttt{not}\,\bot$

Epistemic Splitting

We can simplify the top with respect to W

— top part ——————————————

$$interview(mike) \leftarrow \texttt{not}\,\bot \qquad\qquad , \texttt{not}\,\bot$$

which is now a regular program with a unique answer set

$$M_3 = \{interview(mike)\}$$

and world view $W' = [M_3]$

Epistemic Splitting

— bottom part —

$$eligible(X) \leftarrow high(X)$$
$$eligible(X) \leftarrow minority(X), fair(X)$$
$$\sim eligible(X) \leftarrow \sim fair(X), \sim high(X)$$
$$fair(mike) \vee high(mike)$$

— top part —

$$interview(X) \leftarrow \texttt{not}\,\mathbf{K}\,eligible(X),\ \texttt{not}\,\mathbf{K}\,\sim eligible(X)$$

The world view of the whole program is $\{M_1 \cup M_3, M_2 \cup M_3\}$ where

$$M_1 \cup M_3 = \{fair(mike), interview(mike)\}$$
$$M_2 \cup M_3 = \{high(mike), eligible(mike), interview(mike)\}$$

Epistemic Splitting: Formally

Definition (Epistemic splitting set)

A set of atoms U is an epistemic splitting set of program Π if for each rule $r \in \Pi$

1. all atoms in r belong to U, or
2. no atom from U occurs outside a subjective literal in r

Epistemic Splitting: Formally

Definition (Epistemic splitting set)

A set of atoms U is an epistemic splitting set of program Π if for each rule $r \in \Pi$

1. all atoms in r belong to U, or
2. no atom from U occurs outside a subjective literal in r

U defines a disjoint splitting on Π where bottom Π_b contains rules that (1) and top Π_t rules that (2). □

Epistemic Splitting: Formally

Definition (Epistemic splitting set)

A set of atoms U is an epistemic splitting set of program Π if
for each rule $r \in \Pi$

1. all atoms in r belong to U, or
2. no atom from U occurs outside a subjective literal in r

U defines a disjoint splitting on Π where bottom Π_b contains rules that (1)
and top Π_t rules that (2). □

- In our running example $U = At \setminus \{interview(mike)\}$

Epistemic Splitting: Formally

Definition (Epistemic splitting set)

A set of atoms U is an epistemic splitting set of program Π if for each rule $r \in \Pi$

1. all atoms in r belong to U, or
2. no atom from U occurs outside a subjective literal in r

U defines a disjoint splitting on Π where bottom Π_b contains rules that 1 and top Π_t rules that 2. □

- In our running example $U = At \setminus \{interview(mike)\}$
- Subjective constraints with all atoms in U can be included in Π_b or Π_t

Epistemic Splitting: Formally

Definition (Subjective reduct)

Π^W_U = replace in Π each subjective literal L with atoms in U by: $\top$ if $W \models L$ or by $\bot$ otherwise. □

Property (Epistemic splitting, Formally)

A semantics satisfies epistemic splitting *iff, for any epistemic splitting set of any program* Π*, the following two conditions are equivalent:*

- W *is a world view of* Π
- $W = \{ I_b \cup I_t \mid I_b \in W_b \text{ and } I_t \in W_t \}$ *where* W_b *is a world view of* Π_b *and* W_t *is a world view of* $(\Pi_t)^{W_b}_U$. □

Epistemic splitting in Gelfond's original semantics

Theorem

[Gelfond 1991] satisfies epistemic splitting.

Epistemic splitting in Gelfond's original semantics

Theorem

[Gelfond 1991] satisfies epistemic splitting.

☞ [Watson 2000] proved a different splitting result for [Gelfond 1991]

- It is syntactically less restrictive: it allows dependences involving atoms,
- The price to pay is that an additional semantic condition needs to be checked for all possible world views.

Epistemic Splitting: Consequences

Epistemically stratified program = no cycles through subjective literals

Theorem

Epistemic splitting + supra-ASP + Π epistemically stratified
$\Rightarrow$ Π has (at most) one world view W.

W can be computed iteratively applying the splitting property. □

Epistemic Splitting: Consequences

Epistemically stratified program = no cycles through subjective literals

Theorem

Epistemic splitting + supra-ASP + Π epistemically stratified
$\Rightarrow \Pi$ has (at most) one world view W. □

Example

— layer 1 —

$$\begin{aligned} eligible(X) &\leftarrow high(X) \\ eligible(X) &\leftarrow minority(X), fair(X) \\ \sim eligible(X) &\leftarrow \sim fair(X), \sim high(X) \end{aligned}$$

$$fair(mike) \vee high(mike)$$

— layer 2 —

$$interview(X) \leftarrow \texttt{not}\, \mathbf{K}\, eligible(X),\ \texttt{not}\, \mathbf{K} \sim eligible(X)$$

— layer 3 —

$$appointment(X) \leftarrow \mathbf{K}\, interview(X)$$

Epistemic Splitting: Consequences

A property introduced by [Leclerc & Kahl 2018]

Property (Subjective constraint monotonicity)

A semantics satisfies subjective constraint monotonicity if, for every program Π and subjective constraint c, the world views of $\Pi \cup \{c\}$ are precisely the world views of Π such that $W \models c$ (in $S5$). □

In standard ASP, we have that

$$a \leftarrow \mathtt{not}\ b \qquad b \leftarrow \mathtt{not}\ a \qquad \bot \leftarrow b$$

has a unique stable model $\{a\}$.

Epistemic Splitting: Consequences

A property introduced by [Leclerc & Kahl 2018]

Property (Subjective constraint monotonicity)

A semantics satisfies subjective constraint monotonicity if, for every program Π and subjective constraint c, the world views of $\Pi \cup \{c\}$ are precisely the world views of Π such that $W \models c$ (in $S5$). □

Similarly, we may expect that

$$a \leftarrow \texttt{not}\ \mathbf{K}\, b \qquad\qquad b \leftarrow \texttt{not}\ \mathbf{K}\, a \qquad\qquad \bot \leftarrow \mathbf{K}\, b$$

has a unique world view $[\{a\}]$. The program containing the two first rules has two world views: $[\{a\}]$ and $[\{b\}]$

Epistemic Splitting: Consequences

A property introduced by [Leclerc & Kahl 2018]

Property (Subjective constraint monotonicity)

A semantics satisfies subjective constraint monotonicity if, for every program Π *and subjective constraint* c*, the world views of* $\Pi \cup \{c\}$ *are precisely the world views of* Π *such that* $W \models c$ *(in* $S5$*).* □

Similarly, we may expect that

$$a \leftarrow \texttt{not}\ \mathbf{K}\, b \qquad b \leftarrow \texttt{not}\ \mathbf{K}\, a \qquad \bot \leftarrow \mathbf{K}\, b$$

has a unique world view $[\{a\}]$. The program containing the two first rules has two world views: $[\{a\}]$ and $[\{b\}]$

Theorem

Epistemic splitting implies subjective constraint monotonicity. □

Outline

Epistemic splitting: application to conformant planning

Example

To turn on a *light*, we can toggle one of two lamps ℓ_1 or ℓ_2.
Initially, ℓ_1 is *plugged* but we ignore the state of ℓ_2.
Get a plan to guarantee *light* in one step.

Ignoring inertia for simplicity, given $L \in \{\ell_1, \ell_2\}$ we can use the rules

$$\begin{array}{cc} plugged(\ell_1) & light \leftarrow toggle(L), plugged(L) \\ plugged(\ell_2) \vee {\sim} plugged(\ell_2) & \bot \leftarrow toggle(\ell_1), toggle(\ell_2) \end{array}$$

Epistemic splitting: application to conformant planning

Example

To turn on a *light*, we can toggle one of two lamps ℓ_1 or ℓ_2.
Initially, ℓ_1 is *plugged* but we ignore the state of ℓ_2.
Get a plan to guarantee *light* in one step.

Ignoring inertia for simplicity, given $L \in \{\ell_1, \ell_2\}$ we can use the rules

$$plugged(\ell_1) \qquad light \leftarrow toggle(L), plugged(L)$$

$$plugged(\ell_2) \vee \sim plugged(\ell_2) \qquad \bot \leftarrow toggle(\ell_1), toggle(\ell_2)$$

- $\{toggle(\ell_1)\}$ conformant plan: we get one world view by adding

$$toggle(\ell_1) \qquad \bot \leftarrow \texttt{not}\ \mathbf{K}\ light$$

Epistemic splitting: application to conformant planning

Example

To turn on a *light*, we can toggle one of two lamps ℓ_1 or ℓ_2.
Initially, ℓ_1 is *plugged* but we ignore the state of ℓ_2.
Get a plan to guarantee *light* in one step.

Ignoring inertia for simplicity, given $L \in \{\ell_1, \ell_2\}$ we can use the rules

$$\begin{array}{cc} plugged(\ell_1) & light \leftarrow toggle(L), plugged(L) \\ plugged(\ell_2) \vee \sim plugged(\ell_2) & \bot \leftarrow toggle(\ell_1), toggle(\ell_2) \end{array}$$

- $\{toggle(\ell_1)\}$ conformant plan: we get one world view by adding

$$toggle(\ell_1) \qquad\qquad \bot \leftarrow \texttt{not}\ \mathbf{K}\ light$$

- $\{toggle(\ell_2)\}$ is not a conformant plan: we get no world view by replacing $toggle(\ell_1)$ by $toggle(\ell_2)$ above

Epistemic splitting: application to conformant planning

Let us assume a semantics has a set of rules $\mathit{Choice}(a)$ which has two world views $W_1 = [\ \{a\}\ ]$ and $W_2 = [\emptyset]$.

For instance, in [Gelfond 1991] we can define

$$\mathit{Choice}(a) \stackrel{\text{def}}{=} \{a \leftarrow \mathtt{not}\ \mathbf{K}\, \mathtt{not}\ a\}$$

Epistemic splitting: application to conformant planning

Let us assume a semantics has a set of rules $Choice(a)$ which has two world views $W_1 = [\ \{a\}\ ]$ and $W_2 = [\emptyset]$.

For instance, in [Gelfond 1991] we can define

$$Choice(a) \stackrel{\text{def}}{=} \{a \leftarrow \texttt{not}\ \mathbf{K}\, \texttt{not}\ a\}$$

Under epistemic splitting and supra-ASP, we can obtain conformant plans following the usual ASP methodology:

1. GENERATE: $Choice(a_t)$ for every action a_t and time $1 \leq t \leq n$
2. DEFINE: an ASP-program describing effects of actions and inertia
3. TEST: a constraint $\bot \leftarrow \texttt{not}\ \mathbf{K}\, goal_n$

Epistemic splitting: application to conformant planning

Let us assume a semantics has a set of rules $\mathit{Choice}(a)$ which has two world views $W_1 = [\ \{a\}\ ]$ and $W_2 = [\emptyset]$.

For instance, in [Gelfond 1991] we can define

$$\mathit{Choice}(a) \stackrel{\text{def}}{=} \{a \leftarrow \mathtt{not}\ \mathbf{K}\, \mathtt{not}\ a\}$$

Under epistemic splitting and supra-ASP, we can obtain conformant plans following the usual ASP methodology:

1. GENERATE: $\mathit{Choice}(a_t)$ for every action a_t and time $1 \leq t \leq n$
2. DEFINE: an ASP-program describing effects of actions and inertia
3. TEST: a constraint $\bot \leftarrow \mathtt{not}\ \mathbf{K}\, \mathit{goal}_n$

The conformant plan is the sequence of actions $\langle a_1, \ldots, a_n \rangle$ such that $W \models \mathbf{K}\, a_t$ for the unique world view W

Outline

Epistemic splitting in existing semantics

- The rest of semantics [Gelfond 11, Fariñas et al. 15, Kahl et al. 15, Shen & Eiter 17, Son et al. 17] do not satisfy epistemic splitting.

Epistemic splitting in existing semantics

- The rest of semantics [Gelfond 11, Fariñas et al. 15, Kahl et al. 15, Shen & Eiter 17, Son et al. 17] do not satisfy epistemic splitting.
- Take this program:

$$a \leftarrow \texttt{not}\ b \tag{6}$$
$$b \leftarrow \texttt{not}\ a \tag{7}$$

$$c \leftarrow \mathbf{K}\, a \tag{8}$$
$$\bot \leftarrow \texttt{not}\ c \tag{9}$$

- Bottom $\{(6)-(7)\}$ has world view $[\{a\},\{b\}] \not\models \mathbf{K}\, a$ and we get

$$c \leftarrow \bot \tag{10}$$
$$\bot \leftarrow \texttt{not}\ c \tag{11}$$

with no world view.

Epistemic splitting in existing semantics

However, according to [Gelfond 11] or [Kahl et al. 15], the reduct w.r.t. $[\{a, c\}]$ is computed as follows:

$$a \leftarrow \texttt{not}\ b \tag{6}$$

$$b \leftarrow \texttt{not}\ a \tag{7}$$

$$c \leftarrow \mathbf{K}\, a \tag{8}$$

$$\bot \leftarrow \texttt{not}\ c \tag{9}$$

Epistemic splitting in existing semantics

However, according to [Gelfond 11] or [Kahl et al. 15], the reduct w.r.t. $[\{a, c\}]$ is computed as follows:

$$a \leftarrow \texttt{not}\ b \tag{6}$$

$$b \leftarrow \texttt{not}\ a \tag{7}$$

$$c \leftarrow a \tag{8}$$

$$\bot \leftarrow \texttt{not}\ c \tag{9}$$

Epistemic splitting in existing semantics

However, according to [Gelfond 11] or [Kahl et al. 15], the reduct w.r.t. $[\{a, c\}]$ is computed as follows:

$$a \leftarrow \texttt{not}\ b \tag{6}$$

$$b \leftarrow \texttt{not}\ a \tag{7}$$

$$c \leftarrow a \tag{8}$$

$$\bot \leftarrow \texttt{not}\ c \tag{9}$$

This program has a unique answer set $\{a, c\}$ and, thus, $[\{a, c\}]$ is a world view.

Epistemic splitting in existing semantics

However, according to [Gelfond 11] or [Kahl et al. 15], the reduct w.r.t. $[\{a, c\}]$ is computed as follows:

$$a \leftarrow \texttt{not}\ b \tag{6}$$

$$b \leftarrow \texttt{not}\ a \tag{7}$$

$$c \leftarrow a \tag{8}$$

$$\bot \leftarrow \texttt{not}\ c \tag{9}$$

This program has a unique answer set $\{a, c\}$ and, thus, $[\{a, c\}]$ is a world view.

[Fariñas et al. 15, Shen & Eiter 17, Son et al. 17] agree on this world view

Conclusions

1. We have introduced some formal properties to be satisfied for any semantics for epistemic specifications.
2. In particular, we have studied some of the consequences of satisfying epistemic splitting
   - Constraint monotonicity
   - Application of the ASP methodology to conformant planning
3. Unfortunately, only [Gelfond 1991] semantics satisfies epistemic splitting while it suffers from self-supportedness.

☞ See talk about founded world views (tomorrow 🕘 16:30)

Splitting Epistemic Logic Programs

Pedro Cabalar[1], Jorge Fandinno[2] and Luis Fariñas del Cerro[2]

[1] University of Corunna, Spain
[2] IRIT, Toulouse, France

Thanks for your attention!

LPNMR'19, June 5th, 2019
Philadelphia, PA, USA