Computing Preferred Answer Sets in Answer Set Programming

Wakaki, T., Inoue, K.,[*] Sakama, C.[**] and Nitta, K.[†]

Shibaura Institute of Technology

Kobe University,[*] Wakayama University[**]

Tokyo Institute of Technology[†]

Contents

1. Introduction

- **Prioritized Logic Programs**: **PLP (P, Φ)**

  Sakama and Inoue [JICSLP-96, AI-2000]

  ⇒ explicit representation of priorities in a logic program

  ⇒ can realize various forms of nonmonotonic reasoning

  ⇒ Semantics is given by ***preferred answer sets***.

- **Different approaches:** frameworks and implementations

  (e.g.) -ordered logic programs (Delgrande and Schaub '03)

  -preferred answer sets of ELP (Brewka and Eiter '99, '03)

- **Problems of prior works:**

  Sakama and Inoue's naïve procedure of computing preferred answer sets of PLP [AI-2000]

  ⇒ is applicable to ***a limited class*** and is turned ***unsound***.

- **We propose *a sound and complete procedure*** to compute all preferred answer set of PLP in answer set programming.
  - a generate and test algorithm
  - meta-programming

《Our method》

- **the idea:** construct ***a translated program*** T[P, Φ, S] from a PLP (P, Φ) and any answer set S of P whose answer set is preferable to S,

⇒ enables *to **generate preferences*** to decide whether any answer set of P is preferred or not.

- **We propose *a sound and complete procedure*** to compute all preferred answer set of PLP in answer set programming.
  - a generate and test algorithm
  - meta-programming

《Our method》

- **the idea:** translate a PLP (P, Φ) and any answer set S of P into a logic program $T[P, \Phi, S]$ whose answer sets are preferable to S,

⇓

enables *to generate preferences* to decide whether any answer set of P is preferred or not.

- **The soundness and completeness theorems** for the procedure are proved.
- The application to legal reasoning shows the possibility for our procedure to handle the **dynamic preferences** in addition to the original static ones of PLP,

⇓

  - widen the class of PLPs
  - further increase their expressiveness.

2. Preliminaries

- a ***general extended disjunctive program*** (GEDP) is a set of rules of the form:

$$L_1 \mid \ldots \mid L_k \mid not\ L_{k+1} \mid \ldots \mid not\ L_l \leftarrow L_{l+1}, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n \quad (1)$$

$$(\ n \geqslant m \geqslant l \geqslant k \geqslant 0\ )$$

$not\ L_i$: called a NAF literal

- A rule with variables

  ⇒ a set of its ground instances

Semantics of GEDP

Lit_P: a set of all ground literals in the language of P

Definition 1

(1) Let P be a ***not-free*** ground GEDP

The ***answer set* of** P is the smallest subset S of Lit_P satisfying the following conditions:

- for any rule such as $L_1 | \ldots | L_k \leftarrow L_{l+1}, \ldots, L_m$ in P, if $\{L_{l+1}, \ldots, L_m\} \subseteq S$, then $L_i \in S$ for some i $(1 \leq i \leq k)$.
  In particular, for any integrity constraint such as $\leftarrow L_{l+1}, \ldots, L_m$ in P, $\{L_{l+1}, \ldots, L_m\} \not\subseteq S$ holds.
- if S contains a pair of complementary literals, then $S = Lit_P$.

(2) Let P be any ground GEDP.

For any set $S \subseteq Lit_P$, let P^S (so called *reduct*) be the ***not-free*** ground GEDP obtained as follows:

- a rule: $L_1 | \ldots | L_k \leftarrow L_{l+1}, \ldots, L_m$ is in P^S

if there is a ground rule in P of the form

$$L_1 | \ldots | L_k \mid not\ L_{k+1} | \ldots | not\ L_l \leftarrow L_{l+1}, \ldots, L_m, not\ L_{m+1}, \ldots, not\ L_n$$

where

$\{L_{k+1}, \ldots, L_l\} \subseteq S$ and $\{L_{m+1}, \ldots, L_n\} \cap S = \phi$.

For P^S, its answer sets have been defined in (1).

Then, S is an ***answer set*** of P if S is an answer set of P^S.

- An answer set is *consistent* if it is not Lit_P.
- The answer set Lit_P is said to be *contradictory*.

- If a GEDP *P* has a consistent answer set,
  then, it is *consistent* ;
  otherwise, it is *inconsistent*.

2.2 Prioritized Logic Programs

$\mathcal{L}_P^* \stackrel{\text{def}}{=} Lit_P \cup \{ not\ L \mid L \in Lit_P \}$

$\preceq$: a priority relation (pre-order on $\mathcal{L}_P^*$)

Definition 2 (priorities between literals)

$e_1, e_2 \in \mathcal{L}_P^*$

$e_1 \preceq e_2$: e_2 has a higher priority than e_1

$e_1 \prec e_2$: $e_1 \preceq e_2$ and $e_2 \not\preceq e_1$

Definition 3 (prioritized logic programs:PLP)

A prioritized logic program (PLP) is defined as a pair (P, Φ) where P is a GEDP and Φ is a set of priorities over $\mathcal{L}_P^*$.

Semantics of PLP

(P, Φ): a PLP

$\sqsubseteq$: a preference relation

defined over the answer sets of P

according to priorities in Φ

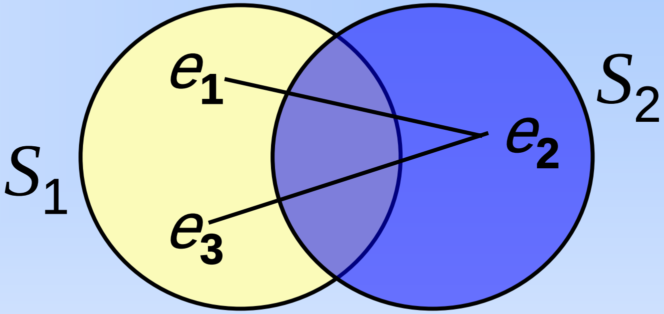
$S_1 \sqsubseteq S_2$: preference

S_2 is ***preferable*** to S_1 w.r.t Φ

for answer sets S_1 and S_2 of P.

Definition 4 (preference between answer sets)

For any answer sets S_1, S_2 and S_3 of P,



(i) $S_1 \sqsubseteq S_1$

(ii) $S_1 \sqsubseteq S_2$ if

$$\exists e_2 \in S_2 - S_1 \, [\, \exists e_1 \in S_1 - S_2 \text{ such that } (e_1 \leq e_2) \in \Phi^* \wedge \neg \exists e_3 \in S_1 - S_2 \text{ such that } (e_2 < e_3) \in \Phi^* \,]$$

(iii) if $S_1 \sqsubseteq S_2$ and $S_2 \sqsubseteq S_3$, *then* $S_1 \sqsubseteq S_3$

Semantics of PLP

Definition 5 (preferred answer sets)

An answer set S of P is called **a *preferred answer set*** of PLP (P, Φ)
if $\boldsymbol{S} \sqsubseteq \boldsymbol{S'}$ implies $\boldsymbol{S'} \sqsubseteq \boldsymbol{S}$ with respect to Φ
for any answer set $\boldsymbol{S'}$ of P.

In other words,

An answer set S of P is called **a *preferred answer set*** of PLP (P, Φ) if $\boldsymbol{S} \not\sqsubset \boldsymbol{S'}$
with respect to Φ for any any answer set $\boldsymbol{S'}$ of P.

Example 1

PLP (P, Φ)

P : $p \leftarrow not\ q,$ $\Phi = \{\ p \preceq \mathbf{not\ p}\ \}$

$q \leftarrow not\ p,$

preferred answer sets of (P, Φ) : $\{\ q\ \}$

PLP (P', Φ')

P' : $p \leftarrow not\ q,$ $\Phi' = \{\ p \preceq p'\ \}$

$q \leftarrow not\ p,$

$p' \leftarrow not\ p,$

preferred answer sets of (P', Φ') : $\{\ p', q\ \}$

3. Computing Preferred Answer Sets

Let (P, Φ) be a PLP s.t. Φ contains no NAF formulas.

《Our method of computing *preferred answer sets*》

generate-and-test algorithms

① **generate** all answer sets of P

② **check** whether each answer set of P is a preferred answer set of (***P***, ***Φ***) using ***preferences generated by the translated program T*[*P*, *Φ*, *S*]** constructed from PLP **(*P*, *Φ*)** and any answer set ***S*** of ***P***.

3.1 Translation for Preference Generation

- **meta-programming**:
  The priorities of Φ as well as a GEDP P are represented in the same translated program $T[P, \Phi, S]$ s.t.
  a ***priority***: $c \preceq d \in \Phi$ ⟺ a ***literal***: $\preceq(c_t, d_t)$
  where c_t, d_t are terms representing literals c, d.

- $L \in S$ is **renamed** by a newly introduced L^*
  in order to encode a given answer set S and another answer set S' in a same answer set of $T[P, \Phi, S]$.

- For a term c_t representing a literal c as well as its renamed c^*,
  $m_1(c_t)$ means $c \in S$, and $m_2(c_t)$ means $c \in S'$,
  where m_1, m_2 are predicate symbols.

Definition 6

$Lit_P{}^* = \{ L^* \mid L \in Lit_P \}$, $C = \{ L_t \mid L \in Lit_P \}$

where Lit_P : a finite set,

L_t : a term containing *no function symbols*.

Definition 7

Given PLP (P, Φ) and an answer set S of P

$$T[P, \Phi, S] \overset{\text{def}}{=} P \cup \Gamma \cup \Pi$$

- Γ is a set of ***domain dependent*** rules constructed from (P, Φ) and S,
- Π is a set of ***domain independent*** rules.

$$T[P, \Phi, S] \stackrel{\text{def}}{=} P \cup \Gamma \cup \Pi$$

Γ :

1. $L^* \leftarrow,$ for any $L \in S$

2. $\preceq(a_t, b_t) \leftarrow,$ for any $a \preceq b \in \Phi$ where $a_t, b_t \in C$

3. $m_1(L_t) \leftarrow L^*,$ $m_2(L_t) \leftarrow L,$ for any $L \in Lit_P$, $L^* \in Lit_P{}^*$ where $L_t \in C$

Π :

4. $\leq(x, x) \leftarrow$

5. $\leq(x, z) \leftarrow \leq(x, y), \leq(y, z).$

6. $<(x, y) \leftarrow \leq(x, y),\ not \leq(y, x).$

7. $gr_1(x,y) \leftarrow m_1(x), \leq(x,y),\ m_2(y), not\ m_2(x), not\ m_1(y).$

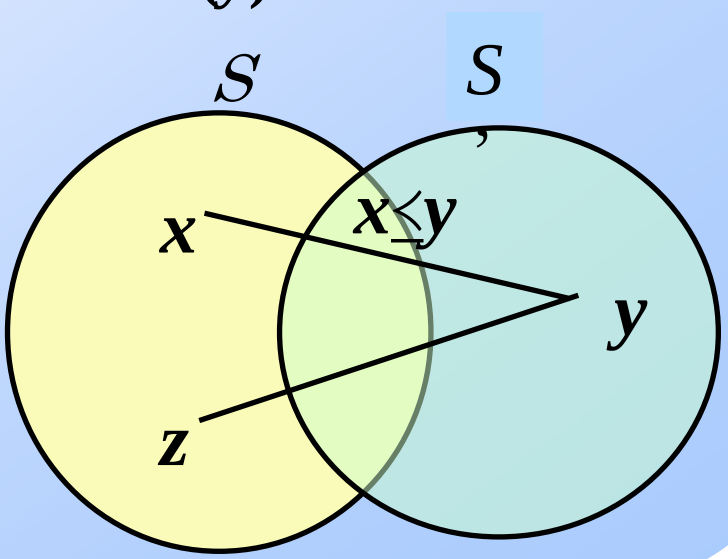
8. $gr_2(y,z) \leftarrow m_2(y), <(y,z),\ m_1(z),\ not\ m_1(y), not\ m_2(z).$

9. $attacked(y) \leftarrow gr_2(y,z)$

10. $defeated(x) \leftarrow gr_1(x,y), not\ attacked(y).$

11. $better \leftarrow defeated(x).$

12. $\leftarrow not\ better.$



tie-preferred and *strictly preferred* answer set

Definition 8

A preferred answer set S of a PLP (P, Φ) is called a ***tie-preferred*** if there is another preferred answer set S' of (P, Φ) such that $S \sqsubseteq S'$ and $S' \sqsubseteq S$.

S is called a ***strictly preferred*** if $S \not\sqsubseteq S'$ for any preferred answer set S' of P.

Theorem 1 (Soundness/Completeness)

Let $\boldsymbol{T[P, \Phi, S]}$ *be a GEDP constructed from a PLP* $\boldsymbol{(P, \Phi)}$ *and an answer set* $\boldsymbol{S}$ *of* $\boldsymbol{P}$.

- *Then, if* $\boldsymbol{T[P, \Phi, S]}$ *is consistent,* $\boldsymbol{S' \stackrel{\text{def}}{=} E \cap Lit_P}$ *is another answer set of P such that* $\boldsymbol{S \sqsubseteq S'}$ *for any answer set* $\boldsymbol{E}$ *of* $\boldsymbol{T[P, \Phi, S]}$.
- *Conversely, if there is another answer set* $\boldsymbol{S'}$ *of* $\boldsymbol{P}$ such that $\boldsymbol{S \sqsubseteq S'}$, then $\boldsymbol{T[P, \Phi, S]}$ is consistent.

Theorem 2

Let ***T***[***P*, *Φ*,*S***] *be a GEDP constructed from a PLP* (***P***, *Φ*) *and an answer set* ***S*** *of* ***P***.

- *Then,* ***T***[***P*, *Φ*,*S***] *is inconsistent if and only if* ***S*** *is a strictly preferred answer set of* (*P*, *Φ*).

Example 2

PLP (P, Φ)

$$P: \quad p \mid q \leftarrow \qquad\qquad \Phi: \{ p \leq q , q \leq r \}$$
$$q \mid r \leftarrow$$

$$Lit_{\mathbf{p}} = \{ p, \quad q, \quad r, \quad \neg p, \quad \neg q, \quad \neg r \}$$

$$Lit^*_{\mathbf{p}} = \{ p^*, \; q^*, \; r^*, \neg p^*, \neg q^*, \neg r^* \}$$
$$\boldsymbol{C} = \{ p_t , \quad q_t , \quad r_t , \quad np_t , \quad nq_t , \quad nr_t \}$$

⬇

Rule 2: $\leq(p_t, q_t) \leftarrow \;, \quad \leq(q_t, r_t) \leftarrow$

Rule 3 :
$m_1(p_t) \leftarrow p^*, \qquad m_1(np_t) \leftarrow \neg p^*,$
$m_2(p_t) \leftarrow p, \qquad m_2(np_t) \leftarrow \neg p, \quad$ *etc.*

PLP (P, Φ)

$P:\ p \mid q \leftarrow$ $\qquad$ $\Phi:\ \{\, p \leq q\ ,\ q \leq r\ \}$

$q \mid r \leftarrow$

- answer sets of P: $S_1 = \{\, p\ ,\ r\ \}$, $S_2 = \{\, q\ \}$

Rule 1: $p^* \leftarrow,\quad r^* \leftarrow,\quad$ *for* S_1

- $T[P, \Phi, S_1] = P \cup \Gamma_1 \cup \Pi$

Γ_1:

$p^* \leftarrow,\quad r^* \leftarrow,\quad \leq (p_t, q_t) \leftarrow\ ,\quad \leq (q_t, r_t) \leftarrow,$

$m_1(p_t) \leftarrow p^*,\quad m_1(q_t) \leftarrow q^*,\quad m_1(r_t) \leftarrow r^*,$

$m_1(np_t) \leftarrow \neg p^*,\quad m_1(nq_t) \leftarrow \neg q^*,\quad m_1(nr_t) \leftarrow \neg r^*,$

$m_2(p_t) \leftarrow p,\quad m_2(q_t) \leftarrow q,\quad m_2(r_t) \leftarrow r,$

$m_2(np_t) \leftarrow \neg p,\quad m_2(nq_t) \leftarrow \neg\, q,\quad m_2(nr_t) \leftarrow \neg r.$

3.2 A Procedure of Computing Preferred Answer Sets

Procedure ***CompPAS*** (P, Φ, Δ)

Input: a PLP (P, Φ)

Output: the set Δ of all preferred answer sets of (P, Φ)

Step1: Compute the set AS of all answer sets of P.

Step2: If Φ is empty,

(a) then $\Delta := AS$, return Δ.

(b) otherwise,

i. $\Omega := \{s_i \mid 1 \leqq i \leqq |AS|\}$ //s_i : the individual constant

ii. To each answer set $S \in AS$, assign the respective $s_i \in \Omega$ called answer set ID.

Step3: if $T[P, \Phi, S]$ is consistent for any answer set $S \in AS$ whose answer set ID is s, do from (a) to (b) for its each answer set E,

(a) put $S' := E \cap Lit_p$ and find the answer set ID s' for S'

(b) put $\Sigma := \Sigma \cup \{ \sqsubseteq (s, s') \leftarrow \}$ //initially Σ is empty.

Step4: Compute an answer set ***U*** of the logic program as follows;

$$\Psi \cup \Sigma \cup \{\mathbf{as(s)} \leftarrow \mid \mathbf{s} \in \Omega\}$$

Step5: Return Δ which is given by

$$\Delta \stackrel{\text{def}}{=} \{\boldsymbol{S} \in \boldsymbol{AS} \mid \boldsymbol{S} \text{ is an answer set whose answer set ID } \boldsymbol{s} \text{ satisfies } \mathbf{p\text{-}as}(\boldsymbol{s}) \in \boldsymbol{U}\}$$

Table 1. A set Ψ of rules

Ψ **:**	$\sqsubseteq(x, x) \leftarrow \mathbf{as}(x).$
	$\sqsubseteq(x, z) \leftarrow \sqsubseteq(x, y),\ \sqsubseteq(y, z).$
	$\sqsubset(x, y) \leftarrow \sqsubseteq(x, y),\ \textit{not} \sqsubseteq(y, x).$
	$\mathbf{worse}(x) \leftarrow \sqsubset(x, y).$
	$\mathbf{p\text{-}as}(x) \leftarrow \mathbf{as}(x),\ \textit{not}\ \mathbf{worse}(x).$

Example 2

PLP (P, Φ)

$P: \; p \mid q \leftarrow$

$\quad\;\; q \mid r \leftarrow$

$\Phi: \; \{ p \preceq q \, , \, q \preceq r \}$

Step 1:

- answer sets of P:

  $S_1 = \{ p \, , r \}$, $S_2 = \{ q \}$

Step 3:

- $T[P, \Phi, S_1] = P \cup \Gamma_1 \cup \Pi \Rightarrow$ ***inconsistent***

Γ_1:

$p^* \leftarrow, \quad r^* \leftarrow, \quad \preceq(p_t, q_t) \leftarrow, \quad \preceq(q_t, r_t) \leftarrow,$

$m_1(p_t) \leftarrow p^*, \quad m_1(q_t) \leftarrow q^*, \quad m_1(r_t) \leftarrow r^*,$

$m_1(np_t) \leftarrow \neg p^*, \quad m_1(nq_t) \leftarrow \neg q^*, \quad m_1(nr_t) \leftarrow \neg r^*,$

$m_2(p_t) \leftarrow p, \quad m_2(q_t) \leftarrow q, \quad m_2(r_t) \leftarrow r,$

$m_2(np_t) \leftarrow \neg p, \quad m_2(nq_t) \leftarrow \neg q, \quad m_2(nr_t) \leftarrow \neg r.$

- $T[P, \Phi, S_2] \Rightarrow S_2 \sqsubseteq S_1$

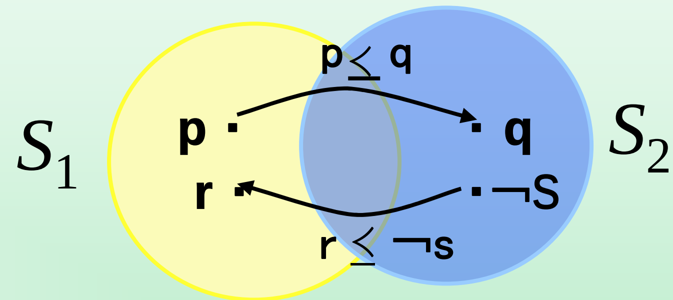
S_1: a preferred answer set

Example 5

P : $p \leftarrow not\ q,$ Φ : $\{ p \preceq q , \neg s \preceq r \}$

$q \leftarrow not\ p.$

$r \leftarrow p, \quad \neg s \leftarrow q.$



answer sets of P : $S_1 = \{ p , r \}$, $S_2 = \{ q, \neg s \}$

$T[P, \Phi, S_1] \Rightarrow S_1 \sqsubseteq S_2$

$T[P, \Phi, S_2] \Rightarrow S_2 \sqsubseteq S_1$

$\Rightarrow \Sigma$

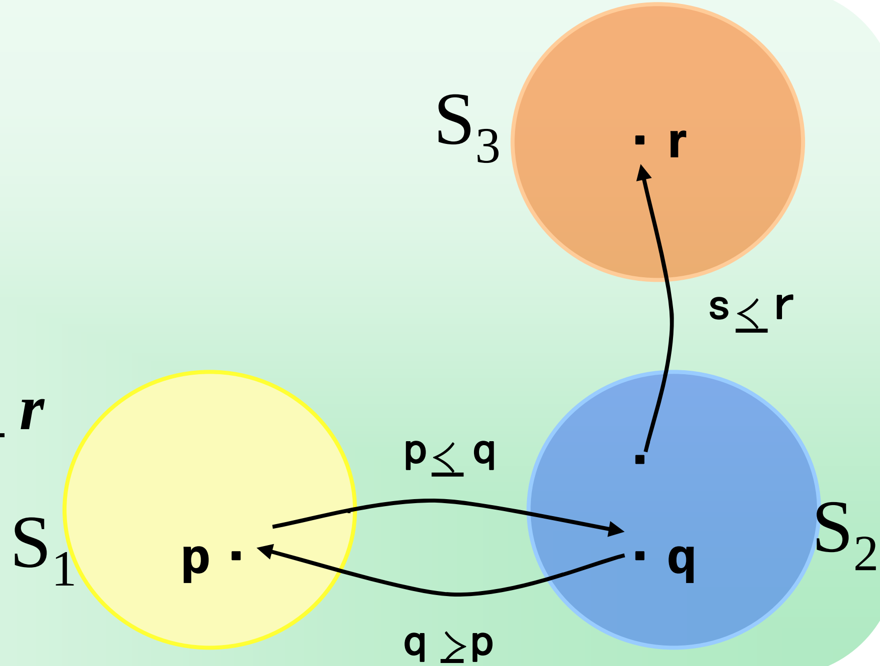
Step3: $\Sigma = \{ \sqsubseteq (s_1, s_2), \sqsubseteq (s_2, s_1)\}$

Step4: $\Psi \cup \Sigma \cup \{as\ (s_1) \leftarrow,\ as$

S_1, S_2 : preferred answer sets

Example 4 ***strictly preferred*** **answer sets**

$$P: \quad p \leftarrow not\ q,\ not\ r,$$
$$q \leftarrow not\ p,\ not\ r,$$
$$r \leftarrow not\ p,\ not\ q,$$
$$s \leftarrow q$$
$$\Phi: \quad p \preceq q\ ,\ q \preceq p\ ,\ s \preceq r$$



Step3 : $T[P, \Phi, S_1] \Rightarrow S_1 \sqsubseteq S_2$

$T[P, \Phi, S_2] \Rightarrow S_2 \sqsubseteq S_1 \quad S_2 \sqsubseteq S_3$

$T[P, \Phi, S_3]$: *inconsistent*

$\Sigma = \{ \sqsubseteq(s_1, s_2) \leftarrow, \sqsubseteq(s_2, s_1) \leftarrow, \sqsubseteq(s_2, s_3) \}$

Step4 : $\Psi \cup \Sigma \cup \{as(s_1) \leftarrow, as(s_2) \leftarrow,$

4. Application to Legal Reasoning

《Legal problem (Gorden, 1993)》

P: $\quad posses \leftarrow, \; ship \leftarrow, \; \neg filstate \leftarrow,$

$perfected \leftarrow posses, \; not \; ab1,$ (*UCC*)

$\neg perfected \leftarrow ship, \; \neg filstate, \; not \; ab2,$ (*SMA*)

$ab1 \mid not \; ab1, \quad ab2 \mid not \; ab2, \quad \leftarrow ab1, ab2,$

$ucc \leftarrow not \; ab1, \quad sma \leftarrow not \; ab2.$

Answer sets of *P*:

$S_1 = \{ \; perfected, \; posses, \; ship, \neg filstate, \; ab2, \; ucc \; \}$

$S_2 = \{ \; \neg perfected, \; posses, \; ship, \neg filstate, \; ab1, \; sma \; \}$

Conflict between ***UCC*** and ***SMA !!***

The principle of *Lex Posterior* gives precedence ***newer*** laws.

The principle of *Lex Superior* gives precedence to laws supported by the ***higher authority***.

UCC is newer than ***SMA***, and ***SMA*** has higher authority than ***UCC***.

《Extended **rule 2** of Γ》

$$\Phi_1: \quad moreRecent(ucc_t, sma_t) \leftarrow,$$

$$fed\,(sma_t) \leftarrow, \qquad state\,(ucc_t) \leftarrow,$$

$$lp(Y,X) \leftarrow moreRecent(X,Y),$$

$$ls(Y,X) \leftarrow fed(X),\ state(Y),$$

$$\preceq(Y,X) \leftarrow lp(Y,X),\ not\ conf_1(X,Y), \qquad (LP)$$

$$\preceq(Y,X) \leftarrow ls(Y,X),\ not\ conf_1(X,Y), \qquad (LS)$$

$$T[P, \Phi_1, S_1] \Rightarrow S_1 \sqsubseteq S_2$$

$$T[P, \Phi_1, S_2] \Rightarrow S_2 \sqsubseteq S_1$$

S_1, S_2: preferred answer sets

Conflict between ***LP*** and ***LS !!***

Meta-priority:

LexPosterior(X,Y) $\preceq$ *LexSuperior(U,V)*

PLP (*P*, Φ_2):

$$\Phi_2 = \Phi_1 \cup \{conf_1(Y, X) \leftarrow lp(X, Y), ls(Y, X), not\ conf_2(X, Y)\}$$

$T[P, \Phi_1, S_2] \Rightarrow$ **inconsistent !**

Only S_2 is a preferred answer set of a PLP (*P*, Φ_2).

S_1={ *perfected*, *posses*, *ship*, ¬*filstate*, *ab2*, *ucc* }

S_2={ ¬*perfected*, *posses*, *ship*, ¬*filstate*, *ab1*, *sma* }

5. Related works and conclusion

- propose a procedure to compute preferred answer sets of a PLP (P, Φ) in answer set programming.
- prove the *soundness* and c*ompleteness* theorems for the procedure.
- The procedure enables PLPs to handle dynamic preferences in addition to the original static ones.
- Example 5 is a counter example for the soundness

  of the Sakama and Inoue's naïve procedure.

  Their procedure becomes sound by using pre-order

  priority relation $\preceq$ instead of strict priority relattion $\prec$

  from Φ.

《Future works》

- We are now implementing our procedure by using the ASP solver ***dlv*** and ***C*++**.
- More precise formalization to accomodate dynamic prefernces in the framework of PLPs will be shown in our subseqent paper.

Thank you

for your Attention