Human Conditional Reasoning in Answer Set Programming

Chiaki Sakama

Wakayama University, Japan

NMR 2023@Rhodes, Greece
September 2023

Background

- People use **conditional sentences** and reason with them in everyday life.
- However, **human conditional reasoning** is **not** always logically valid.
- In psychology and cognitive science, it is well known that humans are more likely to perform **logically invalid but pragmatic inference**.

Example

S: If the team wins the first round tournament, then it advances to the final round.

P: The team wins the first round tournament.

C: The team advances to the final round.

- **Affirming the antecedent (AA)** (**Modus Ponens**) concludes **C** from **S** and **P**.

Example

S: If the team wins the first round tournament, then it advances to the final round.

P: The team wins the first round tournament.

C: The team advances to the final round.

- **Affirming the antecedent (AA)** (**Modus Ponens**) concludes **C** from **S** and **P**.
- **Denying the consequent (DC)** (**Modus Tollens**) concludes $\neg$ **P** from **S** and $\neg$ **C** .

Example

S: If the team wins the first round tournament,
then it advances to the final round.

P: The team wins the first round tournament.

C: The team advances to the final round.

- **Affirming the antecedent (AA)** (**Modus Ponens**) concludes **C** from **S** and **P**.
- **Denying the consequent (DC)** (**Modus Tollens**) concludes $\neg$ **P** from **S** and $\neg$ **C** .
- **Affirming the consequent (AC)** concludes **P** from **S** and **C**.

Example

S: If the team wins the first round tournament, then it advances to the final round.

P: The team wins the first round tournament.

C: The team advances to the final round.

- **Affirming the antecedent (AA)** (**Modus Ponens**) concludes **C** from **S** and **P**.
- **Denying the consequent (DC)** (**Modus Tollens**) concludes $\neg$ **P** from **S** and $\neg$ **C** .
- **Affirming the consequent (AC)** concludes **P** from **S** and **C**.
- **Denying the antecedent (DA)** concludes $\neg$ **C** from **S** and $\neg$ **P**.

AA and **DC** are **logically valid**, while **AC** and **DA** are **logically invalid** and often called **logical fallacies**.

Purpose

- The need of considering the **pragmatics** of conditional reasoning has been recognized in cognitive psychology, while relatively little attention has been paid for realizing it in **logic programming**.
- We formulate **human conditional reasoning** in **answer set programming (ASP)**, and realize pragmatic **AC** and **DA** inferences as well as **DC** inference in a uniform manner.
- We characterize **human reasoning tasks** in cognitive psychology, and address applications to **commonsense reasoning** in AI.

Program

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

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

where L_i's are (positive or negative) literals, and not is default negation (NAF). Given a rule r of the above form, $head^+(r)=\{L_1,\ldots,L_k\}$, $head^-(r)=\{L_{k+1},\ldots,L_l\}$, $body^+(r)=\{L_{l+1},\ldots,L_m\}$,and $body^-(r)=\{L_{m+1},\ldots,L_n\}$.
A rule is called a **constraint** if $head^+(r)=head^-(r)=\varnothing$.
A rule is called a **fact** if $body^+(r)=body^-(r)=\varnothing$.

- A program is **consistent** if it has a consistent answer set.
- A program is **contradictory** if it has the answer set Lit (the set of all literals).
- A program is **incoherent** if it has no answer set.

Example

Let Π be the progam:

$$p\,;\, not\, q \leftarrow,$$
$$q\,;\, not\, p \leftarrow\ .$$

Then Π has two answer sets $\varnothing$ and $\{p, q\}$.

An answer set of a GEDP is not always minimal.

Example

Let Π be the progam:

$$p \,;\, not\, q \leftarrow,$$
$$q \,;\, not\, p \leftarrow \;.$$

Then Π has two answer sets $\emptyset$ and $\{p, q\}$.

An answer set of a GEDP is not always minimal.

The rule

$$not\, p \,;\, not\, q \leftarrow$$

is semantically equivalent to the constraint "$\leftarrow p, q$".

When $head^+(r) = \emptyset$, NAF-literals in the head are shifted to literals in the body.

AC Completion (1)

Let Π be a program and $r \in \Pi$ a rule of the form:

$$L_1 ; \cdots ; L_k ; \mathit{not}\, L_{k+1} ; \cdots ; \mathit{not}\, L_l$$
$$\leftarrow \; L_{l+1}, \ldots, L_m, \mathit{not}\, L_{m+1}, \ldots, \mathit{not}\, L_n.$$

First, for each disjunct in $\mathit{head}^+(r)$ and $\mathit{head}^-(r)$, converse the implication:

$$L_{l+1}, \ldots, L_m, \mathit{not}\, L_{m+1}, \ldots, \mathit{not}\, L_n \leftarrow L_j \quad (1 \leq j \leq k) \tag{1}$$
$$L_{l+1}, \ldots, L_m, \mathit{not}\, L_{m+1}, \ldots, \mathit{not}\, L_n \leftarrow \mathit{not}\, L_j \; (k+1 \leq j \leq l). \tag{2}$$

In particular, (1) is not produced if $\mathit{head}^+(r) = \varnothing$ or $\mathit{body}^+(r) = \mathit{body}^-(r) = \varnothing$; and (2) is not produced if $\mathit{head}^-(r) = \varnothing$ or $\mathit{body}^+(r) = \mathit{body}^-(r) = \varnothing$.
The set of all (1) and (2) is denoted as $\mathit{conv}(r)$.

AC Completion (2)

Next, define

$$ac(\Pi) = \{\, \Sigma_1\,;\,\cdots\,;\,\Sigma_p \leftarrow \ell_j \;\mid\; \Sigma_i \leftarrow \ell_j\ (1 \le i \le p) \text{ is in } \bigcup_{r \in \Pi} conv(r) \,\}$$

where each Σ_i $(1 \le i \le p)$ is a conjunction of literals and NAF-literals and ℓ_j is either a literal L_j $(1 \le j \le k)$ or an NAF-literal $not\, L_j$ $(k+1 \le j \le l)$.

The **AC completion** of Π is defined as:

$$AC(\Pi) = \Pi \cup ac(\Pi).$$

AC Completion (3)

The set $ac(\Pi)$ contains a rule having a disjunction of conjunctions in its head, while it is transformed to rules of a GEDP. That is, the rule:

$$(\ell_1^1, \ldots, \ell_{m_1}^1) \,;\, \cdots \,;\, (\ell_1^p, \ldots, \ell_{m_p}^p) \leftarrow \ell_j$$

is identified with the set of $m_1 \times \cdots \times m_p$ rules of the form:

$$\ell_{i_1}^1 \,;\, \cdots \,;\, \ell_{i_p}^p \leftarrow \ell_j \qquad (1 \leq i_k \leq m_k;\, 1 \leq k \leq p).$$

By this fact, $AC(\Pi)$ is viewed as a GEDP.
The semantics of $AC(\Pi)$ is defined by its answer sets.

Example

Let Π be the program:

$$p\ ;\ not\, q \leftarrow r,\ not\, s, \qquad p \leftarrow q.$$

Then $ac(\Pi)$ becomes

$$(r,\ not\, s)\ ;\ q \leftarrow p, \qquad r,\ not\, s \leftarrow not\, q$$

where the 1st rule is identified with

$$r\ ;\ q \leftarrow p, \qquad not\, s\ ;\ q \leftarrow p$$

and the 2nd rule is identified with

$$r \leftarrow not\, q, \qquad not\, s \leftarrow not\, q.$$

$\Pi \cup ac(\Pi) \cup \{p \leftarrow\}$ has two answer sets $\{p, q\}$ and $\{p, r\}$.

Formal Properties

A consistent program Π may produce an inconsistent $AC(\Pi)$. In converse, an inconsistent Π may produce a consistent $AC(\Pi)$.

$\Pi_1 = \{\, p \leftarrow \neg\, p, \quad p \leftarrow \,\}$ is **consistent**, but
$AC(\Pi_1) = \Pi_1 \cup \{\, \neg\, p \leftarrow p \,\}$ is **contradictory**.

$\Pi_2 = \{\, \leftarrow \mathit{not}\, p, \quad q \leftarrow p, \quad q \leftarrow \,\}$ is **incoherent**, but
$AC(\Pi_2) = \Pi_2 \cup \{\, p \leftarrow q \,\}$ is **consistent**.

Formal Properties

A consistent program Π may produce an inconsistent $AC(\Pi)$. In converse, an inconsistent Π may produce a consistent $AC(\Pi)$.

$\Pi_1 = \{\, p \leftarrow \neg\, p, \quad p \leftarrow \,\}$ is **consistent**, but $AC(\Pi_1) = \Pi_1 \cup \{\, \neg\, p \leftarrow p \,\}$ is **contradictory**.

$\Pi_2 = \{\, \leftarrow \mathit{not}\, p, \quad q \leftarrow p, \quad q \leftarrow \,\}$ is **incoherent**, but $AC(\Pi_2) = \Pi_2 \cup \{\, p \leftarrow q \,\}$ is **consistent**.

If a program Π contains neither NAF nor constraint, then $AC(\Pi)$ is consistent. Moreover, for any answer set S of Π, there is an answer set T of $AC(\Pi)$ such that $S \subseteq T$.

If a program Π is contradictory, then $AC(\Pi)$ is contradictory.

DC completion (1)

Let Π be a program. For each rule $r \in \Pi$ of the form:

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

define $wdc(r)$ as the rule:

$$not\, L_{l+1} ; \cdots ; not\, L_m \,;\, L_{m+1} ; \cdots ; L_n \leftarrow not\, L_1, \ldots, not\, L_k, L_{k+1}, \ldots, L_l \qquad (3)$$

and define $sdc(r)$ as the rule:

$$\neg L_{l+1} ; \cdots ; \neg L_m \,;\, L_{m+1} ; \cdots ; L_n \leftarrow \neg L_1, \ldots, \neg L_k, L_{k+1}, \ldots, L_l . \qquad (4)$$

(3) or (4) becomes a fact if $head^+(r) = head^-(r) = \varnothing$; and it becomes a constraint if $body^+(r) = body^-(r) = \varnothing$.

DC completion (2)

The **weak DC completion** of Π is defined by

$$WDC(\Pi) = \Pi \;\cup\; \{\, wdc(r) \;\mid\; r \in \Pi \,\},$$

the **strong DC completion** of Π is defined by

$$SDC(\Pi) = \Pi \;\cup\; \{\, sdc(r) \;\mid\; r \in \Pi \,\}.$$

Given $\Pi = \{\, p \leftarrow not\, q \,\}$, it becomes

$$\begin{aligned} WDC(\Pi) &= \{\, p \leftarrow not\, q, \quad q \leftarrow not\, p \,\}, \\ SDC(\Pi) &= \{\, p \leftarrow not\, q, \quad q \leftarrow \neg\, p \,\}. \end{aligned}$$

Then $WDC(\Pi)$ has two answer sets $\{p\}$ and $\{q\}$, while $SDC(\Pi)$ has the single answer set $\{p\}$.

Formal Properties

If a program Π has a consistent answer set S, then S is an answer set of $WDC(\Pi)$.

The converse does not hold in general.

The program $\Pi = \{ \leftarrow not\, p \}$ has no answer set, while $WDC(\Pi) = \{ \leftarrow \; not\, p, \;\; p \leftarrow \}$ has the answer set $\{p\}$.

Formal Properties

If a program Π has a consistent answer set S, then S is an answer set of $WDC(\Pi)$.

The converse does not hold in general.

The program $\Pi = \{ \leftarrow not\, p \}$ has no answer set, while $WDC(\Pi) = \{ \leftarrow\ not\, p,\ \ p \leftarrow \}$ has the answer set $\{p\}$.

Let Π be a consistent program s.t. every constraint in Π is not-free. Then, $SDC(\Pi)$ is not contradictory.

If a program Π is contradictory, then both $WDC(\Pi)$ and $SDC(\Pi)$ are contradictory.

Weak DA Completion (1)

Let Π be a program and $r \in \Pi$ a rule of the form:

$$L_1 \,;\, \cdots \,;\, L_k \,;\, \mathit{not}\, L_{k+1} \,;\, \cdots \,;\, \mathit{not}\, L_l \\ \leftarrow \; L_{l+1}, \, \ldots, \, L_m, \, \mathit{not}\, L_{m+1}, \, \ldots, \, \mathit{not}\, L_n.$$

First, inverse the implication:

$$\mathit{not}\, L_i \leftarrow \mathit{not}\, L_{l+1};\, \cdots \,; \mathit{not}\, L_m \,;\, L_{m+1}; \cdots \,;\, L_n \quad (1 \leq i \leq k) \tag{5}$$

$$L_i \leftarrow \mathit{not}\, L_{l+1};\, \cdots \,; \mathit{not}\, L_m \,;\, L_{m+1}; \cdots \,;\, L_n \quad (k+1 \leq i \leq l) \tag{6}$$

(5) is not produced if $\mathit{head}^+(r) = \varnothing$ or $\mathit{body}^+(r) = \mathit{body}^-(r) = \varnothing$;
(6) is not produced if $\mathit{head}^-(r) = \varnothing$ or $\mathit{body}^+(r) = \mathit{body}^-(r) = \varnothing$.
The set of rules (5)–(6) is denoted as $\mathit{winv}(r)$.

Weak DA Completion (2)

Next, define

$$wda(\Pi) \;=\; \{\, \ell_i \leftarrow \Gamma_1, \ldots, \Gamma_p \;\mid\; \ell_i \leftarrow \Gamma_j \; (1 \leq j \leq p) \;\text{ is in }\; \bigcup_{r \in \Pi} winv(r) \,\}$$

where ℓ_i is either a literal L_i $(k+1 \leq i \leq l)$ or an NAF literal $not\, L_i$ $(1 \leq i \leq k)$, and each Γ_j is a disjunction of literals and NAF literals.

The **weak DA completion** of Π is defined as:

$$WDA(\Pi) = \Pi \;\cup\; wda(\Pi).$$

Weak DA Completion (3)

The set $wda(\Pi)$ contains a rule having a conjuction of disjunctions in its body, while it is transformed to rules of a GEDP. That is, the rule:

$$\ell_i \leftarrow (\ell_1^1; \cdots; \ell_{m_1}^1), \ldots, (\ell_1^p; \cdots; \ell_{m_p}^p)$$

is identified with the set of $m_1 \times \cdots \times m_p$ rules of the form:

$$\ell_i \leftarrow \ell_{j_1}^1, \ldots, \ell_{j_p}^p \;\; (1 \leq j_k \leq m_k;\ 1 \leq k \leq p).$$

By this fact, $WDA(\Pi)$ is viewed as a GEDP.
The semantics of $WDA(\Pi)$ is defined by its answer sets.

Example

Let Π be the program:

$$p\,;\,q \leftarrow r, \mathit{not}\,s, \quad q\,;\,\mathit{not}\,r \leftarrow t, \quad s \leftarrow .$$

Then $\mathit{wda}(\Pi)$ becomes

$$\mathit{not}\,p \leftarrow \mathit{not}\,r; s, \quad \mathit{not}\,q \leftarrow (\mathit{not}\,r; s), \mathit{not}\,t, \quad r \leftarrow \mathit{not}\,t$$

where the 1st rule is identified with

$$\mathit{not}\,p \leftarrow \mathit{not}\,r, \quad \mathit{not}\,p \leftarrow s,$$

and the 2nd rule is identified with

$$\mathit{not}\,q \leftarrow \mathit{not}\,r, \mathit{not}\,t, \quad \mathit{not}\,q \leftarrow s, \mathit{not}\,t.$$

Then, $\Pi \cup \mathit{wda}(\Pi)$ has the answer set $\{s, r\}$.

Strong DA Completion (1)

Let Π be a program and $r \in \Pi$ a rule of the form:

$$L_1\,;\,\cdots\,;\,L_k\,;\,not\,L_{k+1}\,;\cdots\,;\,not\,L_l$$
$$\leftarrow\; L_{l+1},\,\ldots,\,L_m,\,not\,L_{m+1},\,\ldots,\,not\,L_n.$$

First, inverse the implication:

$$\neg\,L_i \leftarrow \neg\,L_{l+1};\cdots;\neg L_m; L_{m+1};\cdots;L_n \quad (1 \leq i \leq k) \tag{7}$$
$$L_i \leftarrow \neg\,L_{l+1};\cdots;\neg L_m; L_{m+1};\cdots;L_n \quad (k+1 \leq i \leq l) \tag{8}$$

As in the case of WDA, the rules (7)–(8) are not produced when their heads or bodies are empty. The set of rules (7)–(8) is denoted as $sinv(r)$.

Strong DA Completion (2)

Next, define

$$sda(\Pi) = \{\, \ell_i \leftarrow \Gamma_1, \ldots, \Gamma_p \mid \ell_i \leftarrow \Gamma_j \ (1 \le j \le p) \text{ is in } \bigcup_{r \in \Pi} sinv(r) \,\}$$

where ℓ_i is either a literal L_i $(k+1 \le i \le l)$ or an NAF literal $not\, L_i$ $(1 \le i \le k)$, and each Γ_j is a disjunction of positive/negative literals.

The **strong DA completion** of Π is defined as:

$$SDA(\Pi) = \Pi \ \cup \ sda(\Pi).$$

As before, rules in $sda(\Pi)$ are converted into a GEDP, so that $SDA(\Pi)$ is viewed as a GEDP. The semantics of $SDA(\Pi)$ is defined by its answer sets.

Formal Properties

Let Π be an EDP. If S is a consistent answer set of $WDA(\Pi)$, then S is an answer set of Π.

If a program Π is contradictory, then both $WDA(\Pi)$ and $SDA(\Pi)$ are contradictory.

A consistent program Π may produce an inconsistent $WDA(\Pi)$ or $SDA(\Pi)$. In converse, an incoherent Π may produce a consistent $WDA(\Pi)$ or $SDA(\Pi)$.

AC and DA as Default Reasoning

AC and DA often make a program inconsistent. We relax the effects of the AC or DA completion by introducing additional rules as **default rules**.

$$\textbf{(default AC)} \qquad \frac{(\varphi \Rightarrow \psi) \wedge \psi : \varphi}{\varphi}$$

$$\textbf{(default DA)} \qquad \frac{(\varphi \Rightarrow \psi) \wedge \neg\varphi : \neg\psi}{\neg\psi}$$

The **default AC** rule says: given the conditional $\varphi \Rightarrow \psi$ and the fact ψ, conclude φ as a default consequence. The **default DA** rule is read in a similar manner.

Default AC Completion (1)

Let Π be a program. For each rule $r \in \Pi$ of the form:

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

define $dac(r)$ as the set of rules:

$$L_{l+1}, \ldots, L_m, not\, L_{m+1}, \ldots, not\, L_n \leftarrow L_i, \, \Delta \qquad (1 \leq i \leq k), \tag{9}$$

$$L_{l+1}, \ldots, L_m, not\, L_{m+1}, \ldots, not\, L_n \leftarrow not\, L_i, \, \Delta \qquad (k+1 \leq i \leq l) \tag{10}$$

where
$\Delta = not\, \neg L_{l+1}, \ldots, not\, \neg L_m, not\, L_{m+1}, \ldots, not\, L_n.$

Default AC Completion (2)

The **default AC completion** of Π is defined as:

$$DAC(\Pi) = \Pi \ \cup \ dac(\Pi)$$

in which

$$dac(\Pi) \ = \ \{ \, \Sigma_1 \, ; \, \cdots \, ; \, \Sigma_p \ \leftarrow \ \ell_j, \ \Delta_i \ \mid \ \Sigma_i \leftarrow \ell_j, \ \Delta_i \ (1 \leq i \leq p) \ \text{ is in } \bigcup_{r \in \Pi} dac(r) \, \}$$

where each Σ_i $(1 \leq i \leq p)$ is a conjunction of literals and NAF-literals and ℓ_j is either a literal L_j $(1 \leq j \leq k)$ or an NAF-literal $not\, L_j$ $(k+1 \leq j \leq l)$.

Rules in $dac(\Pi)$ are converted into the form of a GEDP.

Formal Properties

$DAC(\Pi)$ turns a contradictory $AC(\Pi)$ into a consistent program.

Let $\Pi = \{p \leftarrow \neg p, \;\; p \leftarrow\}$. Then $AC(\Pi) = \Pi \cup \{\neg p \leftarrow p\}$ is contradictory, while $DAC(\Pi) = \Pi \cup \{\neg p \leftarrow p, \, not\, p\}$ has the single answer set $\{p\}$.

Let Π be a consistent program. If $DAC(\Pi)$ has an answer S, then $S \neq Lit$.

Let Π be a program. If $AC(\Pi)$ has a consistent answer set S, then S is an answer set of $DAC(\Pi)$.

Default DA Completion

Let Π be a program. Define

$$\begin{aligned} wdda(\Pi) &= \{\, \ell_i \leftarrow \Gamma_1, \ldots, \Gamma_p,\, \delta_i^w \mid \\ &\qquad \ell_i \leftarrow \Gamma_j \ (1 \leq j \leq p) \ \text{is in} \bigcup_{r \in \Pi} winv(r) \,\}, \\ sdda(\Pi) &= \{\, \ell_i \leftarrow \Gamma_1, \ldots, \Gamma_p,\, \delta_i^s \mid \\ &\qquad \ell_i \leftarrow \Gamma_j \ (1 \leq j \leq p) \ \text{is in} \bigcup_{r \in \Pi} sinv(r) \,\} \end{aligned}$$

where ℓ_i and Γ_j are the same as those in WDA and SDA. $\delta_i^w = not \neg L_i$ if $\ell_i = L_i$, and $\delta_i^w = not\, L_i$ if $\ell_i = not\, L_i$; $\delta_i^s = not \neg L_i$ if $\ell_i = L_i$, and $\delta_i^s = not\, L_i$ if $\ell_i = \neg L_i$. The **weak default DA completion** and the **strong default DA completion** of Π are respectively defined as:

$$\begin{aligned} WDDA(\Pi) &= \Pi \cup wdda(\Pi), \\ SDDA(\Pi) &= \Pi \cup sdda(\Pi). \end{aligned}$$

Formal Properties

The WDDA/SDDA eliminates contradictory.

Let $\Pi_1 = \{\neg p \leftarrow p, \;\; \neg p \leftarrow\}$ where
$SDA(\Pi_1) = \Pi_1 \cup \{p \leftarrow \neg p\}$ is contradictory.
$SDDA(\Pi_1) = \Pi_1 \cup \{p \leftarrow \neg p, \mathit{not}\, \neg p\}$ has the answer set $\{\neg p\}$.

Let Π be a consistent program. If $WDDA(\Pi)$ (or $SDDA(\Pi)$) has an answer set S, then $S \neq Lit$.

Let Π be a program. If $WDA(\Pi)$ (resp. $SDA(\Pi)$) has a consistent answer set S, then S is an answer set of $WDDA(\Pi)$ (resp. $SDDA(\Pi)$).

Comparison (1)

The proposed completion is **different** from **Clark completion** or **weak completion** in normal logic programs.

Let $\Pi_1 = \{\, p \leftarrow q,\ \ p \leftarrow \,\}$.

- **Clark completion** becomes
  $Comp(\Pi_1) = \{\, p \leftrightarrow q \vee \top,\ \ q \leftrightarrow \bot \,\}$
  that has the single **supported model** $\{p\}$.
- **Weak completion** becomes $wcomp(\Pi_1) = \{\, p \leftrightarrow \top \,\}$
  then p is true but q is unknown.
- **AC completion** becomes $AC(\Pi_1) = \Pi_1 \cup \{\, q \leftarrow p \,\}$ that has the answer set $\{p, q\}$.

As such, Clark completion and weak completion **do not** realize AC inference in general.

Comparison (2)

Let $\Pi_2 = \{\, p \leftarrow \mathit{not}\, q \,\}$.

- $\mathit{Comp}(\Pi_2) = \{\, p \leftrightarrow \neg q, \;\; q \leftrightarrow \bot \,\}$ has the single supported model $\{p\}$.
- $\mathit{wcomp}(\Pi_1) = \{\, p \leftrightarrow \neg q \,\}$ then both p and q are unknown.
- $\mathit{WDC}(\Pi_2) = \Pi_2 \cup \{\, q \leftarrow \mathit{not}\, p \,\}$ has two answer sets $\{p\}$ and $\{q\}$.

Let $\Pi_3 = \{\, p \leftarrow \mathit{not}\, q, \;\; p \leftarrow q, \;\; q \leftarrow p \,\}$.

- $\mathit{Comp}(\Pi_3) = \{\, p \leftrightarrow q \vee \neg q, \;\; q \leftrightarrow p \,\}$ has the single supported model $\{p, q\}$.
- $\mathit{wcomp}(\Pi_3) = \mathit{Comp}(\Pi_3)$ then both p and q are true.
- $\mathit{WDA}(\Pi_3) = \Pi_3 \cup \{\, \mathit{not}\, p \leftarrow q, \mathit{not}\, q, \;\; \mathit{not}\, q \leftarrow \mathit{not}\, p \,\}$ has no answer set.

Final Remark

1. In cognitive psychology, empirical studies show people perform AC/DA/DC depending on the context in which a conditional sentence is used. The proposed theory is used for encoding knowledge in a way that people are likely to use it and realizing pragmatic inferences in ASP.
2. Completions are defined in a modular way, so one can apply respective completion to specific rules of a program according to their contexts. Different completions can be mixed in the same program.
3. Since a GEDP is transformed to a semantically equivalent EDP, answer sets of completed programs are computed using existing answer set solvers.