Logic Programs with Consistency-Restoring Rules

by
Marcello Balduccini and Michael Gelfond

2003 AAAI Spring Symposium
International Symposium on Logical Formalization of Commonsense Reasoning

February 28, 2003
Syntax of CR-Prolog

A **regular rule** is a statement of the form:
\[
  r : h_1 \text{ or } \ldots \text{ or } h_k \leftarrow l_1, \ldots, l_m, \\
  \text{not } l_{m+1}, \ldots, \text{not } l_n
\]
where \( r \) is the name of the rule, and \( h_i, l_i \) are literals.

A **cr-rule** is a statement of the form:
\[
  r : h_1 \text{ or } \ldots \text{ or } h_k \leftarrow l_1, \ldots, l_m, \\
  \text{not } l_{m+1}, \ldots, \text{not } l_n
\]
The rule says that:
if \( l_1, \ldots, l_m \) belong to a set of agent's beliefs and none of \( l_{m+1}, \ldots, l_n \) belongs to it then the agent “may possibly” believe one of the \( h_1, \ldots, h_k \).

This possibility is used only if the agent has no way to obtain a consistent set of beliefs using regular rules only. The extension of A-Prolog by cr-rules is called CR-Prolog.
Syntax of CR-Prolog (cont.)

Example 1

\[ \Pi_0 : \begin{cases} 
  a \leftarrow \text{not } b. \\
  r_1 : b \leftarrow . 
\end{cases} \]

- \( \Pi_0 \) has an answer set \( \{a\} \), computed \textbf{without} the use of cr-rule \( r_1 \).

Now consider

\[ \Pi'_0 = \Pi_0 \cup \{ \neg a. \}. \]

- If \( r_1 \) is not used, \( \Pi'_0 \) in \textbf{inconsistent}.
- The application of \( r_1 \) restores consistency, and leads to the answer set \( \{\neg a, b\} \).
Semantics of Abductive Logic Programs

- Used to define the semantics of CR-Prolog.

- Abductive logic programs are pairs $\langle \Pi, \mathcal{A} \rangle$ where $\Pi$ is a program of A-Prolog and $\mathcal{A}$ is a set of atoms, called *abducibles*.

- The semantics of an abductive program, $\Pi$, is given by the notion of *generalized answer set* – an answer set $M(\Delta)$ of $\Pi \cup \Delta$ where $\Delta \subseteq \mathcal{A}$

- $M(\Delta_1) < M(\Delta_2)$ if $\Delta_1 \subseteq \Delta_2$. We refer to an answer set as *minimal* if it is minimal with respect to this ordering.
Semantics of CR-Prolog

**Definition 1** The hard reduct $hr(\Pi) = \langle H_\Pi, atoms(\{appl\}) \rangle$ transforms CR-Prolog programs into abductive programs. It is defined as follows:

1. Every regular rule of $\Pi$ belongs to $H_\Pi$.

2. For every cr-rule $\rho$ of $\Pi$, with name $r$, the following belongs to $H_\Pi$:
   
   $$\text{head}(\rho) \leftarrow \text{body}(\rho), \text{appl}(r).$$

3. If $preference$ occurs in $\Pi$, $H_\Pi$ contains the following set of rules, denoted by $\Pi_p$:

   $$
   \begin{cases}
   % \text{transitive closure of predicate preference} \\
   \text{is_preferred}(R1, R2) \leftarrow \text{preference}(R1, R2). \\
   \text{is_preferred}(R1, R2) \leftarrow \text{preference}(R1, R3), \text{is_preferred}(R3, R2). \\
   \\
   % \text{no circular preferences} \\
   \leftarrow \text{is_preferred}(R, R). \\
   \\
   % \text{prohibits application of a lesser rule if} \\
   % \text{a better rule is applied} \\
   \leftarrow \text{appl}(R1), \text{appl}(R2), \text{is_preferred}(R1, R2). \\
   \end{cases}
   $$

   ($R_1, R_2, R_3$ are variables for names of rules.)
Semantics of CR-Prolog (cont.)

Definition 2 A set of literals, $C$, is a candidate answer set of $\Pi$ if $C$ is a minimal generalized answer set of $hr(\Pi)$.

Definition 3 Let $C$, $D$ be candidate answer sets of $\Pi$. $C$ is better than $D$ ($C \prec D$) if

$$\exists \text{appl}(r_1) \in C \ \exists \text{appl}(r_2) \in D \ \text{is_preferred}(r_1, r_2) \in C \cap D.$$  \hspace{1cm} (1)

(In the following definition, $\text{atoms}([p, q])$ denotes the set of atoms formed by predicates $p$ and $q$.)

Definition 4 Let $C$ be a candidate answer set of $\Pi$, and $\tilde{C}$ be $C \setminus \text{atoms}([\text{appl}, \text{is_preferred}])$. $\tilde{C}$ is an answer set of $\Pi$ if there exists no candidate answer set, $D$, of $\Pi$ which is better than $C$. 

Semantics of CR-Prolog

— Examples —

Example 2

Let us compute the answer sets of:

$$\Pi_1 \begin{cases} r_1 : p & \leftarrow r, \text{not } q. \\ r_2 : r. \\ r_3 : s & \leftarrow r. \end{cases}$$

(Notice that $\Pi_1 \setminus \{r_3\}$ is consistent.)

The hard reduct of $\Pi_1$ is given by ($\Pi_p$ is omitted):

$$H_{\Pi_1}' \begin{cases} r_1 : p & \leftarrow r, \text{not } q. \\ r_2 : r. \\ r_3' : s & \leftarrow r, \text{appl}(r_3). \end{cases}$$

- $\{p, r, s, \text{appl}(r_3)\}$ is a generalized answer set of $hr(\Pi_1)$, but it is not minimal.

- The only minimal generalized answer set of $hr(\Pi_1)$ is $C' = \{p, r\}$.

- $C'$ is the only answer set of $\Pi_1$. 
Example 3

\[ \begin{align*}
\Pi_2 \quad & \quad \begin{cases}
  r_1 : & p \leftarrow \text{not } q. \\
r_2 : & r \leftarrow \text{not } s. \\
r_3 : & q \leftarrow t. \\
r_4 : & s \leftarrow t. \\
r_5 : & \leftarrow p, r. \\
r_6 : & q \leftarrow \cdot \\
r_7 : & s \leftarrow \cdot \\
r_8 : & t \leftarrow \cdot \\
r_9 : & \text{prefer}(r_6, r_7).
\end{cases}
\end{align*} \]

The hard reduct of \( \Pi_2 \) is given by:

\[ \begin{align*}
H'_{\Pi_2} \quad & \quad \begin{cases}
  r_1 : & p \leftarrow \text{not } q. \\
r_2 : & r \leftarrow \text{not } s. \\
r_3 : & q \leftarrow t. \\
r_4 : & s \leftarrow t. \\
r_5 : & \leftarrow p, r. \\
r'_6 : & q \leftarrow \text{appl}(r_6). \\
r'_7 : & s \leftarrow \text{appl}(r_7). \\
r'_8 : & t \leftarrow \text{appl}(r_8). \\
r_9 : & \text{prefer}(r_6, r_7).
\end{cases}
\end{align*} \]

- The candidate answer sets of \( \Pi_2 \) are (\textit{is-preferred} is omitted):

  \[ \begin{align*}
  C_1 &= \{\text{prefer}(r_6, r_7), \text{appl}(r_6), q, r\} \\
  C_2 &= \{\text{prefer}(r_6, r_7), \text{appl}(r_7), s, p\} \\
  C_3 &= \{\text{prefer}(r_6, r_7), \text{appl}(r_8), t, q, s\}
  \end{align*} \]

- Since \( C_1 < C_2 \), \( \hat{C}_2 \) is not an answer set of \( \Pi_2 \), while \( \hat{C}_1 \) and \( \hat{C}_3 \) are.
Example 4

\[
\begin{cases}
  r_1 : a \leftarrow p. \\
r_2 : a \leftarrow r. \\
r_3 : b \leftarrow q. \\
r_4 : b \leftarrow s. \\
r_{5a} : \leftarrow \text{not } a. \\
r_{5b} : \leftarrow \text{not } b. \\
r_6 : p \uparrow. \\
r_7 : q \uparrow. \\
r_8 : r \uparrow. \\
r_9 : s \uparrow. \\
r_{10} : \text{prefer}(r_6, r_7). \\
r_{11} : \text{prefer}(r_8, r_9).
\end{cases}
\]

- The candidate answer sets of \( \Pi_3 \) are:

\[
C_1 = \{ \text{prefer}(r_6, r_7), \text{prefer}(r_8, r_9), \text{appl}(r_6), \text{appl}(r_9), p, s, a, b \}
\]

\[
C_2 = \{ \text{prefer}(r_6, r_7), \text{prefer}(r_8, r_9), \text{appl}(r_8), \text{appl}(r_7), r, q, a, b \}
\]

- Since \( C_1 \preceq C_2 \) and \( C_2 \preceq C_1 \), \( \Pi_3 \) has no answer set.
Motivating Example

System: an electrical circuit connecting a switch to a light bulb.

Exogenous actions: action “brks” breaks the bulb; action “surge” damages the whole circuit, but leaves the bulb intact if protected.

To model the system we introduce fluents: 
\textit{closed}(SW) – switch \textit{SW} is closed;
\textit{ab}(C) – component \textit{C} is malfunctioning;
\textit{prot}(b) – bulb \textit{b} is protected from power surges;
\textit{active}(r) – relay \textit{r} is active;
\textit{on}(b) – bulb \textit{b} is on.

The action description of the system consists of the rules in the first three sections of the following program, $\Pi_d$. 
\[
\begin{align*}
\text{% DYNAMIC CAUSAL LAWS} \\
h(\text{closed}(s_1), T + 1) &\leftarrow o(\text{close}(s_1), T). \\
h(ab(b), T + 1) &\leftarrow o(\text{brks}, T). \\
h(ab(r), T + 1) &\leftarrow o(\text{srg}, T). \\
h(ab(b), T + 1) &\leftarrow \neg h(\text{prot}(b), T), o(\text{srg}, T).
\end{align*}
\]

\[
\begin{align*}
\text{% DOMAIN CONSTRAINTS} \\
h(\text{active}(r), T) &\leftarrow h(\text{closed}(s_1), T), \neg h(ab(r), T). \\
\neg h(\text{active}(r), T) &\leftarrow h(ab(r), T). \\
\neg h(\text{active}(r), T) &\leftarrow \neg h(\text{closed}(s_1), T). \\
h(\text{closed}(s_2), T) &\leftarrow h(\text{active}(r), T). \\
h(\text{on}(b), T) &\leftarrow h(\text{closed}(s_2), T), \neg h(ab(b), T). \\
\neg h(\text{on}(b), T) &\leftarrow h(ab(b), T). \\
\neg h(\text{on}(b), T) &\leftarrow \neg h(\text{closed}(s_2), T).
\end{align*}
\]

\[
\Pi_d
\begin{align*}
\text{% EXECUTABILITY CONDITION} &\leftarrow o(\text{close}(s_1), T), h(\text{closed}(s_1), T).
\end{align*}
\]

\[
\begin{align*}
\text{% INERTIA} &\leftarrow h(F, T), \text{not } \neg h(F, T + 1). \\
\neg h(F, T + 1) &\leftarrow \neg h(F, T), \text{not } h(F, T + 1).
\end{align*}
\]

\[
\begin{align*}
\text{% REALITY CHECKS} &\leftarrow \text{obs}(F, T), \text{not } h(F, T). \\
&\leftarrow \text{obs}(\neg F, T), \text{not } \neg h(F, T).
\end{align*}
\]

\[
\begin{align*}
\text{% AUXILIARY AXIOMS} \\
o(A, T) &\leftarrow hpd(A, T). \\
h(F, 0) &\leftarrow \text{obs}(F, 0). \\
\neg h(F, 0) &\leftarrow \text{obs}(\neg F, 0).
\end{align*}
\]
Motivating Example (cont.)

— Specifying a history —

Recorded history $\Gamma_n$ (where $n$ is the current time step) is given by a collection of statements of the form:

- $\text{obs}(l,t)$ — ‘fluent literal $l$ was observed to be true at moment $t$’;
- $\text{hpd}(a,t)$ — ‘action $a$ was observed to happen at moment $t$’

where $t$ is an integer from the interval $[0, n)$.

The axioms in the last two sections of $\Pi_d$ establish the relationship between relations $\text{obs}$, $\text{hpd}$ and $h, o$.

- $\text{obs, hpd}$: undoubtedly correct observations;
- $h, o$: predictions made by the agent — may be defeated by further observations.

The reality checks axioms ensure that the agent’s predictions do not contradict his observations.

The trajectories $\langle \sigma_0, a_0, \sigma_1, \ldots, a_{n-1}, \sigma_n \rangle$ defined by $\Gamma_n$ can be extracted from the answer sets of $\Pi_d \cup \Gamma_n$. 
Motivating Example (cont.)
— Specifying a history —

Example 5
(only positive observations are shown for brevity)

History \( \Gamma_1 = \{ \text{obs}(\text{prot}(b), 0), \text{hpd}(\text{close}(s_1), 0) \} \)
defines the trajectory
\[
\langle \{ \text{prot}(b) \}, \{ \text{close}(s_1) \}, \\
\{ \text{closed}(s_1), \text{closed}(s_2), \text{on}(b), \text{prot}(b) \} \rangle.
\]

Example 6

History \( \Gamma_2 = \{ \text{obs}(\text{prot}(b), 0), \text{hpd}(\text{close}(s_1), 0), \text{obs}(\neg\text{closed}(s_1), 1) \} \)
is inconsistent (thanks to the reality checks of \( \Pi_d \)), and hence \( \Gamma_2 \) defines no trajectories.
Motivating Example (cont.)
— Diagnostic Component —

Diagnostic module $DM_0$
A diagnostic module is used to find explanations of a given set of observations $O$.

$$DM_0 : \begin{cases} 
o(A, T) \text{ or } \neg o(A, T) \leftarrow 0 \leq T < n, \\
x_{\text{act}}(A). \end{cases}$$

($x_{\text{act}}(A)$ is satisfied by exogenous actions.)

- If $\Pi_d \cup O$ is consistent, no diagnosis is necessary.

- Otherwise, explanations of $O$ are computed by finding the answer sets of

$$\Pi_d \cup O \cup DM_0$$
Motivating Example (cont.)

– Conclusions –

• Checking consistency and finding a diagnosis in the previous algorithm is achieved by **two calls** to \( l \)-satisfiability checkers — inference engines computing answer sets of logic programs.

• Such multiple calls require the **repetition of a substantial amount of computation** (including grounding of the whole program).

• We have **no way to declaratively specify preferences between possible diagnoses**, and hence may be forced to eliminate unlikely diagnoses by performing extra observations.

\[\Rightarrow\] These problems can be avoided by **introducing cr-rules** in the diagnostic module.
A New Diagnostic Module

Diagnostic Module $DM^{cr}_0$

$$DM^{cr}_0 \begin{cases} r(A,T) : o(A,T) \iff T < n, x_{\text{-act}}(A). \end{cases}$$

(the rule says that some (unobserved) exogenous actions may possibly have occurred in the past.)

Example 7

$$O_1 : \begin{cases} \text{hpdl}(\text{close}(s_1),0). \\ \text{obs}(\text{prot}(b),0). \\ \text{obs}(\text{on}(b),1). \end{cases}$$

- The answer set of $\Pi_d \cup O_1 \cup DM^{cr}_0$ contains no occurrences of exogenous actions – cr-rules are not used.

$$O_2 : \begin{cases} \text{hpdl}(\text{close}(s_1),0). \\ \text{obs}(\text{prot}(b),0). \\ \text{obs}(\neg\text{on}(b),1). \end{cases}$$

- Consistency of the “regular part” of $\Pi_d \cup O_2 \cup DM^{cr}_0$ can be restored only by rule $r(brks,0)$. The observation is explained by the occurrence of $brks$.

$$O_3 : \begin{cases} \text{hpdl}(\text{close}(s_1),0). \\ \text{obs}(\neg\text{on}(b),1). \end{cases}$$

- $\Pi_d \cup O_3 \cup DM^{cr}_0$ has two answer sets, one obtained using $r(brks,0)$, and the other obtained using $r(srg,0)$. The agent concludes that either $brks$ or $srg$ occurred at time 0.
Preferred Explanations

Recall that selection of cr-rules is guided by preference relation \( \text{prefer}(r_1, r_2) \), which says that sets of beliefs obtained by applying \( r_1 \) are preferred over those obtained by applying \( r_2 \).

**Problem:** representing that "brks occurs more often than srg" (hence an explanation based on brks is preferred to one based on srg.)

**Solution:**

\[
\Pi_d^p : \{ \text{prefer}(r(\text{brks}, T), r(\text{srg}, T)) \}.
\]

**Example 8**

\[
\Omega_3 : \begin{cases} 
\text{hpd}(\text{close}(s_1), 0). \\
\text{obs}(\neg \text{on}(b), 1).
\end{cases}
\]

- Given \( \Pi_d \cup \Omega_3 \cup \Pi_d^p \cup DM_0 \), cr-rules are used to conclude that brks occurred at 0.

- The agent does not conclude that srg occurred – this corresponds to a less preferred set of beliefs.

- The agent may derive that srg occurred only if additional information is provided, showing that brks cannot have occurred.
Applications

Dynamic Preferences for $DM^{cr}_0$

Problem: representing the additional information:

"Bulb blow-ups happen more frequently than power surges unless there is a storm in the area."

Solution:

$$DM_p : \begin{cases} \text{prefer}(r(\text{brks}, T), r(\text{srg}, T)) \leftarrow \neg h(\text{storm}, 0). \\
\text{prefer}(r(\text{srg}, T), r(\text{brks}, T)) \leftarrow h(\text{storm}, 0). \end{cases}$$

Example 9

$$O_4 : \begin{cases} hpd(\text{close}(s_1), 0). \\
\text{obs}(\text{storm}, 0). \\
\text{obs}(\neg \text{on}(b), 1). \end{cases}$$

- Obviously $O_4$ requires an explanation. It is storming and therefore the intuitive explanation is $o(\text{srg}, 0)$.
- Program $\Pi_d \cup O_4 \cup DM_p \cup DM^{cr}_0$ has two candidate answer sets. Due to the second rule of $DM_p$ only one of them, containing $\text{srg}$, is the answer set of the program and hence $o(\text{srg}, 0)$ is the explanation of $O_4$. 
Applications (cont.)

Example 10

\[
O_5 : \begin{cases}
    hpd(close(s_1), 0).
    
ob(s(storm, 0) \text{ or } o(s(\neg storm, 0)).
    
ob(\neg on(b), 1).
    
ob(\neg ab(b), 1).
\end{cases}
\]

- Common-sense should tell the agent that there was a power surge. Nothing can be said, however, on whether there has been a storm.
- The answer sets of \( \Pi_d \cup O_5 \cup DM_p \cup DM_{cr}^c \) contain sets of facts:
  
  \[
  \{obs(storm, 0), o(srg, 0)\}
  \]
  
  \[
  \{obs(\neg storm, 0), o(srg, 0)\}
  \]

  which correspond to the intuitive answers.
Applications (cont.)

Generation of shortest plans
Consider the following planning module, $PM_0$:

\[
\begin{align*}
  r_4(T) : & \quad \text{maxtime}(T) \leftarrow n \leq T. \\
  & \quad \text{prefer}(r_4(T), r_4(T + 1)). \\
  r_5(A, T) : & \quad o(A, T) \leftarrow \text{maxtime}(MT), n \leq T < MT.
\end{align*}
\]

(Here $n$ stands for the current time of the agent’s history – in our case 0.)

- Cr-rule $r_4(T)$ says that any time can possibly be the maximum planning time of the agent.
- The second rule gives the preference to shortest plans.
- The last rule allows to the agent the future use of any of his actions.
Applications (cont.)

Example 11: Using $PM_0$
Consider the Yale Shooting Scenario. The agent is given:
- \textit{initial situation:} the turkey is alive and the gun is unloaded;
- \textit{goal:} killing the turkey, represented as:

\[
\begin{align*}
\text{goal} & \leftarrow h(\text{dead}, T). \\
\text{not goal} & \leftarrow. 
\end{align*}
\]

- The goal does not hold at current moment 0, which causes inconsistency.
- Rules $r_5(A, 0), r_5(A, 1), \ldots, r_5(A, MT)$ allow to restore consistency.
- Without the preference relation, $MT$, can be determined by any rule from $r_4(0), r_4(1) \ldots$.
- The preference forces the agent to select the shortest plan – in our case

\[\{o(\text{load}, 0), h(\text{shoot}, 1)\}.\]
Related Work

DLV’s weak constraints

- **Weak constraint**: a constraint that can be violated, to obtain an answer set.
- **Weight**: the cost of violating the weak constraint.
- **Preferred answer set**: minimizes the sum of the weights of the constraints that the answer set violates.

- Weights induce a **total order** on the weak constraints of the program, as opposed to the **partial order** that can be specified on cr-rules.

Diagnostic module for DLV, $DM_{wk}$

\[
\begin{align*}
o(A,T) \text{ or } \neg o(A,T) & \leftarrow 0 \leq T < n, \ x_{\text{act}}(A). \\
\sim o(brks,T), h(storm,0). & [4 : ] \\
\sim o(srg,T), h(storm,0). & [1 : ] \\
\sim o(brks,T), \neg h(storm,0). & [1 : ] \\
\sim o(srg,T), \neg h(storm,0). & [4 : ]
\end{align*}
\]

- **First two constraints**: if a storm occurred, assuming that action $brks$ occurred has a cost of 4, while assuming that action $srg$ occurred has a cost of 1.
- **Last two constraints**: if a storm did not occur, assuming that action $brks$ occurred has a cost of 1, while assuming that action $srg$ occurred has a cost of 4.
Related Work (cont.)

Example 12

\[ O_5 : \begin{cases} 
  hpd(close(s_1), 0). \\
  obs(storm, 0) \text{ or } obs(\neg storm, 0). \\
  obs(\neg on(b), 1). \\
  obs(\neg ab(b), 1).
\end{cases} \]

- The only possible explanation of recorded history \( O_5 \) is the occurrence of \( srg \) at time 0.
- \( \Pi_d \cup O_5 \cup DM_{wk} \) has two “candidate” answer sets:
  \[ \{ \text{obs(storm, 0), o(srg, 0)} \} \]
  \[ \{ \text{obs(\neg storm, 0), o(srg, 0)} \} \]

- **Problem**: the second set of facts has a cost of 4, while the first has a cost of 1. This forces the reasoner to assume, without any sufficient reason, the presence of a storm.
Motivating Example (cont.)
— Agent Architecture —

Our agent architecture is based on the following loop:

\[
\text{Observe-think-act loop}
\]

1. observe the world;
2. interpret the observations;
3. select a goal;
4. plan;
5. execute part of the plan.

**Diagnosis** occurs as follows:

During step 1, the agent obtains observations $O$. At step 2, it first needs to check if $\Pi_d \cup O$ is consistent. If it is not, then it must find explanations for $O$ by computing the answer sets of $\Pi_d \cup O \cup DM_0$. 
Pareto Optimality

Let:
• $\Pi$ be a program,
• $r$ be the name of a cr-rule of $\Pi$
• $A$ and $B$ be generalized answer sets of $H_\Pi$.

Definition 5 $A$ is better than or equal to $B$ w.r.t. $r$ ($A \preceq_r B$) iff:

$$appl(r) \in A \land appl(r) \in B,$$
$$\text{or}$$
$$appl(r) \in A \land \exists \ un appl(r') \in B \text{ s.t.}$$
$$\text{is-preferred}(r, r') \in A \cap B$$

Definition 6 $A$ is better than $B$ w.r.t. $r$ ($A <_r B$) iff:

$$A \preceq_r B, \text{ and}$$
$$appl(r) \notin B.$$
Pareto Optimality (cont.)

**Definition 8** $A$ is a Pareto-optimal candidate answer set of $\Pi$ if there exists no generalized answer set of $\Pi_H$ that dominates $A$.

**Definition 9** $A$ is a Pareto-optimal answer set of $\Pi$ if $A$ is set-theoretic minimal among the Pareto-optimal candidate answer sets of $\Pi$.

- This alternative semantics yields the same results in the previous examples.

- Differences arise with programs where *there is no clear reason to prefer one generalized answer set to another.*
Pareto Optimality (cont.)

Example 13

\[
\begin{array}{l}
\Pi_4 = \left\{ \begin{array}{l}
r_1 : q \leftarrow^+ . \\
r_2 : p \leftarrow^+ . \\
r_3 : a \leftarrow^+ . \\
r_4 : b \leftarrow^+ . \\
\text{prefer}(a, b). \\
\text{prefer}(p, q). \\
\text{ok} \leftarrow a, q. \\
\text{ok} \leftarrow b, p. \\
\leftarrow \text{not ok}. \\
\end{array} \right. \\
\end{array}
\]

- *Original semantics:* \( \Pi_4 \) has no answer set—\( \Pi_4 \) has two candidate answer sets, \( A = \{a, q\} \) and \( B = \{b, p\} \), but \( A \not\prec B \) and \( B \not\prec A \).

- *Pareto optimality:* \( A \) and \( B \) are Pareto-optimal answer sets—none dominates the other; both are Pareto optimal candidate answer sets, and they are also minimal.