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:

$$r : h_1 \text{ or } \dots \text{ or } h_k \leftarrow l_1, \dots, l_m, \text{not } l_{m+1}, \dots, \text{not } l_n$$

where r is the name of the rule, and h_i, l_i are literals.

A **cr-rule** is a statement of the form:

$$r : h_1 \text{ or } \dots \text{ or } h_k \stackrel{+}{\leftarrow} l_1, \dots, l_m, \text{not } l_{m+1}, \dots, \text{not } l_n$$

The rule says that:
if $l_1, \dots, l_m$ belong to a set of agent's beliefs and none of $l_{m+1}, \dots, l_n$ belongs to it then the agent "may possibly" believe one of the $h_1, \dots, h_k$.

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

$$\Pi_0 : \begin{cases} a \leftarrow \text{not } b. \\ r_1 : b \stackrel{+}{\leftarrow} . \end{cases}$$

• Π_0 has an answer set $\{a\}$, computed **without** the use of cr-rule r_1.

Now consider

$$\Pi_0' = \Pi_0 \cup \{\neg a.\}.$$

• If r_1 is not used, Π_0' in **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 Π is a program of A-Prolog and $\mathcal{A}$ is a set of atoms, called *abducibles*.

- The semantics of an abductive program, Π, 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 $\Delta_1 \subset \Delta_2$. 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 Π belongs to H_Π.

2. For every cr-rule ρ of Π, with name r, the following belongs to H_Π:

$$head(\rho) \leftarrow \ body(\rho), appl(r).$$

3. If $prefer$ occurs in Π, H_Π contains the following set of rules, denoted by Π_p:

$$\left\{ \begin{array}{l} \text{\% transitive closure of predicate prefer} \\ \ is\_preferred(R1, R2) \leftarrow \ prefer(R1, R2). \\ \ is\_preferred(R1, R2) \leftarrow \ prefer(R1, R3), \\ \qquad\qquad\qquad\qquad\qquad\quad is\_preferred(R3, R2). \\ \text{\% no circular preferences} \\ \ \leftarrow is\_preferred(R, R). \\ \text{\% prohibits application of a lesser rule if} \\ \text{\% a better rule is applied} \\ \ \leftarrow appl(R1), appl(R2), is\_preferred(R1, R2). \end{array} \right.$$

(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 Π if C is a minimal generalized answer set of $hr(\Pi)$.

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

$$\begin{array}{l} \exists appl(r_1) \in C \;\; \exists appl(r_2) \in D \\ is\_preferred(r_1, r_2) \in C \cap D. \end{array} \tag{1}$$

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

Definition 4 Let C be a candidate answer set of Π, and $\widehat{C}$ be $C \setminus atoms(\{appl, is\_preferred\})$. $\widehat{C}$ is an answer set of Π if there exists no candidate answer set, D, of Π which is better than C.

Semantics of CR-Prolog – Examples –

Example 2

Let us compute the answer sets of:

$$\Pi_1 \left\{ \begin{array}{llll} r_1: & p & \leftarrow & r, \text{not } q. \\ r_2: & r. & & \\ & & & \\ r_3: & s & \stackrel{+}{\leftarrow} & r. \end{array} \right.$$

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

The hard reduct of Π_1 is given by (Π_p is omitted):

$$H'_{\Pi_1} \left\{ \begin{array}{llll} r_1: & p & \leftarrow & r, \text{not } q. \\ r_2: & r. & & \\ & & & \\ r'_3: & s & \leftarrow & r, appl(r_3). \end{array} \right.$$

- $\{p, r, s, appl(r_3)\}$ 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 Π_1.

Example 3

$$\Pi_2 \left\{ \begin{array}{lccl} r_1: & p & \leftarrow & \text{not } q. \\ r_2: & r & \leftarrow & \text{not } s. \\ r_3: & q & \leftarrow & t. \\ r_4: & s & \leftarrow & t. \\ & & & \\ r_5: & & \leftarrow & p, r. \\ & & & \\ r_6: & q \stackrel{+}{\leftarrow} . & & \\ r_7: & s \stackrel{+}{\leftarrow} . & & \\ r_8: & t \stackrel{+}{\leftarrow} . & & \\ & & & \\ r_9: & prefer(r_6, r_7). & & \end{array} \right.$$

The hard reduct of Π_2 is given by:

$$H'_{\Pi_2} \left\{ \begin{array}{lccl} r_1: & p & \leftarrow & \text{not } q. \\ r_2: & r & \leftarrow & \text{not } s. \\ r_3: & q & \leftarrow & t. \\ r_4: & s & \leftarrow & t. \\ & & & \\ r_5: & & \leftarrow & p, r. \\ & & & \\ r'_6: & q & \leftarrow & appl(r_6). \\ r'_7: & s & \leftarrow & appl(r_7). \\ r'_8: & t & \leftarrow & appl(r_8). \\ & & & \\ r_9: & prefer(r_6, r_7). & & \end{array} \right.$$

- The candidate answer sets of Π_2 are ($is\_preferred$ is omitted):

$$C_1 = \{prefer(r_6, r_7), appl(r_6), q, r\}$$
$$C_2 = \{prefer(r_6, r_7), appl(r_7), s, p\}$$
$$C_3 = \{prefer(r_6, r_7), appl(r_8), t, q, s\}$$

- Since $C_1 \prec C_2$, $\widehat{C}_2$ is not an answer set of Π_2, while $\widehat{C}_1$ and $\widehat{C}_3$ are.

Example 4

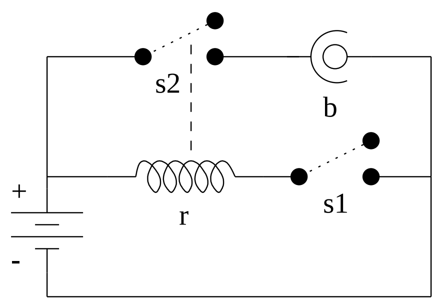
$$\Pi_3 \begin{cases} r_1: & a \leftarrow p. \\ r_2: & a \leftarrow r. \\ r_3: & b \leftarrow q. \\ r_4: & b \leftarrow s. \\ \\ r_{5a}: & \leftarrow \text{not } a. \\ r_{5b}: & \leftarrow \text{not } b. \\ \\ r_6: & p \stackrel{+}{\leftarrow} . \\ r_7: & q \stackrel{+}{\leftarrow} . \\ r_8: & r \stackrel{+}{\leftarrow} . \\ r_9: & s \stackrel{+}{\leftarrow} . \\ \\ r_{10}: & prefer(r_6, r_7). \\ r_{11}: & prefer(r_8, r_9). \end{cases}$$

- The candidate answer sets of Π_3 are:

$$C_1 = \{\, prefer(r_6, r_7), prefer(r_8, r_9), appl(r_6), appl(r_9), p, s, a, b\}$$
$$C_2 = \{\, prefer(r_6, r_7), prefer(r_8, r_9), appl(r_8), appl(r_7), r, q, a, b\}$$

- Since $C_1 \prec C_2$ and $C_2 \prec C_1$, Π_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:
$closed(SW)$ – switch SW is closed;
$ab(C)$ – component C is malfunctioning;
$prot(b)$ – bulb b is protected from power surges;
$active(r)$ – relay r is active;
$on(b)$ – bulb b is on.

The **action description** of the system consists of the rules in the first three sections of the following program, Π_d.

Π_d

%% **DYNAMIC CAUSAL LAWS**

$h(closed(s_1), T+1) \leftarrow o(close(s_1), T).$
$h(ab(b), T+1) \leftarrow o(brks, T).$
$h(ab(r), T+1) \leftarrow o(srg, T).$
$h(ab(b), T+1) \leftarrow \neg h(prot(b), T), o(srg, T).$

%% **DOMAIN CONSTRAINTS**

$h(active(r), T) \leftarrow h(closed(s_1), T), \neg h(ab(r), T).$
$\neg h(active(r), T) \leftarrow h(ab(r), T).$
$\neg h(active(r), T) \leftarrow \neg h(closed(s_1), T).$
$h(closed(s_2), T) \leftarrow h(active(r), T).$
$h(on(b), T) \leftarrow h(closed(s_2), T), \neg h(ab(b), T).$
$\neg h(on(b), T) \leftarrow h(ab(b), T).$
$\neg h(on(b), T) \leftarrow \neg h(closed(s_2), T).$

%% **EXECUTABILITY CONDITION**

$\leftarrow o(close(s_1), T), h(closed(s_1), T).$

%% **INERTIA**

$h(F, T+1) \leftarrow h(F, T), \text{not } \neg h(F, T+1).$
$\neg h(F, T+1) \leftarrow \neg h(F, T), \text{not } h(F, T+1).$

%% **REALITY CHECKS**

$\leftarrow obs(F, T), \text{not } h(F, T).$
$\leftarrow obs(\neg F, T), \text{not } \neg h(F, T).$

%% **AUXILIARY AXIOMS**

$o(A, T) \leftarrow hpd(A, T).$
$h(F, 0) \leftarrow obs(F, 0).$
$\neg h(F, 0) \leftarrow obs(\neg F, 0).$

Motivating Example (cont.)
– Specifying a history –

Recorded history Γ_n (where n is the current time step) is given by a collection of statements of the form:
• $obs(l, t)$ – 'fluent literal l was observed to be true at moment t';
• $hpd(a, t)$ – '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 Π_d establish the **relationship between relations** obs**,** hpd **and** h**,**o.
• 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 Γ_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 $\Gamma_1 = \{obs(prot(b), 0), hpd(close(s_1), 0)\}$

defines the trajectory

$$\langle \{prot(b)\}, \{close(s_1)\}, \{closed(s_1), closed(s_2), on(b), prot(b)\} \rangle.$$

Example 6

History $\Gamma_2 = \{obs(prot(b), 0), hpd(close(s_1), 0), obs(\neg closed(s_1), 1)\}$

is inconsistent (thanks to the reality checks of Π_d), and hence Γ_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.

$$DM_0 : \left\{ \begin{array}{l} o(A,T) \text{ or } \neg o(A,T) \leftarrow 0 \leq T < n, \\ \qquad\qquad\qquad\qquad\qquad\qquad x\_act(A). \end{array} \right.$$

($x\_act(A)$ 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 lp-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.

⇒ These problems can be avoided by **introducing cr-rules** in the diagnostic module.

A New Diagnostic Module

Diagnostic Module DM_0^{cr}

$$DM_0^{cr} \left\{ \begin{array}{ll} r(A,T): & o(A,T) \stackrel{+}{\leftarrow} T < n, \\ & \qquad x\_act(A). \end{array} \right.$$

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

Example 7

$$O_1 : \left\{ \begin{array}{l} hpd(close(s_1), 0). \\ obs(prot(b), 0). \\ obs(on(b), 1). \end{array} \right.$$

• The answer set of $\Pi_d \cup O_1 \cup DM_0^{cr}$ contains no occurrences of exogenous actions – cr-rules are not used.

$$O_2 : \left\{ \begin{array}{l} hpd(close(s_1), 0). \\ obs(prot(b), 0). \\ obs(\neg on(b), 1). \end{array} \right.$$

• Consistency of the "regular part" of $\Pi_d \cup O_2 \cup DM_0^{cr}$ can be restored only by rule $r(brks, 0)$. The observation is explained by the occurrence of $brks$.

$$O_3 : \left\{ \begin{array}{l} hpd(close(s_1), 0). \\ obs(\neg on(b), 1). \end{array} \right.$$

• $\Pi_d \cup O_3 \cup DM_0^{cr}$ 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 $prefer(r_1, r_2)$, 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:

$$\Pi_d^p : \{ \; prefer(r(brks, T), r(srg, T)).$$

Example 8

$$O_3 : \left\{ \begin{array}{l} hpd(close(s_1), 0). \\ obs(\neg on(b), 1). \end{array} \right.$$

• Given $\Pi_d \cup O_3 \cup \Pi_d^p \cup DM_0^{cr}$, 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_0^{cr}
Problem: representing the additional information:

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

Solution:

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

Example 9

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

• Obviously O_4 requires an explanation. It is storming and therefore the intuitive explanation is $o(srg, 0)$.
• Program $\Pi_d \cup O_4 \cup DM_p \cup DM_0^{cr}$ has two candidate answer sets. Due to the second rule of DM_p only one of them, containing srg, is the answer set of the program and hence $o(srg, 0)$ is the explanation of O_4.

Applications (cont.)

Example 10

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

• 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 $\Pi_d \cup O_5 \cup DM_p \cup DM_0^{cr}$ 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:

$$
\begin{cases}
r_4(T): & maxtime(T) \stackrel{+}{\leftarrow} n \leq T. \\
 & prefer(r_4(T), r_4(T+1)). \\
 & \\
r_5(A,T): & o(A,T) \stackrel{+}{\leftarrow} maxtime(MT), n \leq T < MT.
\end{cases}
$$

(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:

◇ *initial situation:* the turkey is alive and the gun is unloaded;

◇ *goal:* killing the turkey, represented as:

$$\begin{cases} goal \leftarrow h(dead, T). \\ \leftarrow \text{not } goal. \end{cases}$$

• 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

$$\{o(load, 0), h(shoot, 1)\}.$$

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}

$$\left\{ \begin{array}{l} o(A,T) \text{ or } \neg o(A,T) \leftarrow 0 \leq T < n,\ \ x\_act(A). \\ :\sim o(brks,T), h(storm,0).\ [4:] \\ :\sim o(srg,T), h(storm,0).\ [1:] \\ :\sim o(brks,T), \neg h(storm,0).\ [1:] \\ :\sim o(srg,T), \neg h(storm,0).\ [4:] \end{array} \right.$$

• *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:

$$\{obs(storm, 0), o(srg, 0)\}$$
$$\{obs(\neg storm, 0), o(srg, 0)\}$$

• **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:

> *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:

- Π be a program,
- r be the name of a cr-rule of Π
- A and B be generalized answer sets of H_Π.

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

$$appl(r) \in A \wedge appl(r) \in B, \text{ or}$$
$$appl(r) \in A \wedge \exists\ appl(r') \in B \text{ s.t. } is\_preferred(r, r') \in A \cap B$$

Definition 6 A is **better than** B w.r.t. r ($A \prec_r B$) iff:

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

Definition 7 A **dominates** B iff:

$$\forall\ appl(r) \in A\ \ A \preceq_r B, \text{ and}$$
$$\exists\ appl(r) \in A\ \ A \prec_r B$$

Pareto Optimality (cont.)

Definition 8 A is a **Pareto-optimal candidate answer set** of Π if there exists no generalized answer set of H_Π that dominates A.

Definition 9 A is a **Pareto-optimal answer set** of Π if A is set-theroretic minimal among the Pareto-optimal candidate answer sets of Π.

- 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

$$\Pi_4 \left\{ \begin{array}{llll} r_1 : & q & \stackrel{+}{\leftarrow} & . \\ r_2 : & p & \stackrel{+}{\leftarrow} & . \\ r_3 : & a & \stackrel{+}{\leftarrow} & . \\ r_4 : & b & \stackrel{+}{\leftarrow} & . \\ \\ \multicolumn{4}{l}{prefer(a,b).} \\ \multicolumn{4}{l}{prefer(p,q).} \\ \\ & ok & \leftarrow & a, q. \\ & ok & \leftarrow & b, p. \\ & & \leftarrow & \text{not } ok. \end{array} \right.$$

- *Original semantics:* Π_4 **has no answer set**– Π_4 has two candidate answer sets, $A = \{a, q\}$ and $B = \{b, p\}$, but $A \prec B$ and $B \prec A$.

- *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.